Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Comparação de Redes Neurais Convolucionais para o diagnóstico de melanoma por meio de Transfer Learning e Data Augmentation em imagens dermatoscópicas com classes desbalanceadas

Marcelo G. M. C. Oliveira
Walyson M. D. Leite

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientadora
Prof.a Dr.a Roberta Barbosa Oliveira

Brasília
2021

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Comparação de Redes Neurais Convolucionais para o diagnóstico de melanoma por meio de Transfer Learning e Data Augmentation em imagens dermatoscópicas com classes desbalanceadas

Marcelo G. M. C. Oliveira
Walyson M. D. Leite

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof.a Dr.a Roberta Barbosa Oliveira (Orientadora)
CIC/UnB

Prof. Dr. Jan Mendonça Correa
CIC/UnB

Prof. Dr. Thiago de Paulo Faleiros
CIC/UnB

Prof. Dr. Marcelo Grandi Mandelli
Coordenador do Bacharelado em Ciência da Computação

Brasília, 20 de maio de 2021

Dedicatória

Dedico esse trabalho ao meu futuro eu. Que em alguns anos, ao ler essa dedicatória, perceba que uma longa caminhada já foi percorrida e que ainda existe um longo caminho pela frente.

Agradecimentos

Gostaríamos de agradecer a nossa orientadora Roberta Barbosa Oliveira pela atenção, paciência, pelo tempo dedicado a nos ajudar e por escolher compartilhar um pedaço de seu conhecimento conosco. Também agradecemos aos nossos colegas e amigos com quem compartilhamos as nossas alegrias e frustrações.

Marcelo Giordano: Em primeiro lugar, agradeço a minha linda família, Julia e Jordana, que me deram forças para continuar essa jornada chamada graduação, mesmo em tempos tão difíceis como os que passamos atualmente. Aos meus pais, que me incentivaram a sonhar com um aprendizado digno em uma instituição de qualidade como a Universidade de Brasília. Por fim, ao meu amigo Walyson Maxwel que, sem o seu auxílio e paciência, não seria possível nem sonhar com a finalização deste trabalho.

Walyson Maxwel: Agradeço à Deus em quem eu encontro segurança nos momentos difíceis, à minha família que me deu o suporte necessário na caminhada universitária e ao Marcelo, com quem eu compartilhei as dificuldades desse trabalho e assim como eu, entende os obstáculos que foram superados.

Resumo

Um grande desafio encontrado atualmente na detecção de doenças de pele é a classificação entre lesões benignas e malignas. Algumas lesões podem ter características similares e o diagnóstico correto é custoso, uma vez que são necessários exames clínicos para a correta classificação da lesão. Na tentativa de resolução do problema, a automatização do diagnóstico por meio do aprendizado de máquina torna-se uma alternativa a ser considerada. Nesse contexto, esse trabalho realiza a comparação do desempenho de três arquiteturas já conhecidas: AlexNet, GoogLeNet e ResNet. O objetivo principal é verificar se as técnicas de *data augmentation* e de *transfer learning* influenciam o desempenho das arquiteturas e são suficientes para promover um aumento da capacidade preditiva dos modelos para detecção de melanoma em imagens dermatoscópicas. Cinco etapas experimentais foram realizadas de forma a verificar a influência de cada uma das técnicas no desempenho das arquiteturas considerando uma base de dados desbalanceada. As métricas de avaliação utilizadas foram: acurácia balanceada, sensibilidade, especificidade curva ROC e AUC. Com a combinação das técnicas de *data augmentation* e *transfer learning*, a arquitetura ResNet obteve os melhores resultados com uma acurácia balanceada de 0,752 e AUC de 0,839. Importante destacar que esse desempenho foi obtido quando as técnicas de *data augmentation* e *transfer learning* foram aplicadas simultaneamente, demonstrando, portanto, a eficiência de suas utilizações de forma combinada uma vez que foram responsáveis por proporcionar um aumento na acurácia balanceada em mais de 25% para essa arquitetura.

Palavras-chave: Aprendizado de Máquina, Redes Neurais Convolucionais, Aumento de Dados, Transferência de Aprendizado, Melanoma

Abstract

A major challenge currently encountered in the detection of skin diseases is the classification between benign and malignant lesions. Some lesions may have similar characteristics and the correct diagnosis is costly, since clinical examinations are necessary for the correct classification of the lesion. In an attempt to solve the problem, the automation of diagnosis through machine learning becomes an alternative to be considered. In this context, this work compares the performance of three already known architectures: AlexNet, GoogLeNet and ResNet. The main objective is to verify if the data augmentation and transfer learning techniques influence the performance of the architectures and are sufficient to promote an increase in the predictive capacity of the models for detecting melanoma in dermoscopic images. Five experimental steps were carried out in order to verify the influence of each technique on the performance of the architectures considering an unbalanced database. The evaluation metrics used were: balanced accuracy, sensitivity, specificity, ROC curve and AUC. With the combination of data augmentation and transfer learning techniques, the ResNet architecture achieved the best results, with a balanced accuracy of 0,752 and AUC of 0,839. It is important to highlight that this performance was obtained when the techniques of data augmentation and transfer learning were applied simultaneously, demonstrating, therefore, the efficiency of their uses in a combined way since they were responsible for providing an increase in balanced accuracy of over 25% for this architecture.

Keywords: Machine Learning, Convolutional Neural Networks, Data Augmentation, Transfer Learning, Melanoma

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AUC Área Abaixo da Curva.

BP Retropropagação de erros.

CAD Diagnóstico Auxiliado por Computador.

CNN Rede Neural Convolucional.

CNNs Redes Neurais Convolucionais.

DNN Rede Neural Profunda.

DNNs Redes Neurais Profundas.

FN Falso Negativo.

FP Falso Positivo.

GANs Redes Adversárias Generativas.

ILSVRC Desafio de Reconhecimento da ImageNet.

INCA Instituto Nacional de Câncer.

LTU Unidade de Limite Linear.

LTUs Unidades de Limite Linear.

MLP Perceptron Multicamadas.

ReLU Unidade Linear Retificada.

RNA Rede Neural Artificial.

RNAs Rede Neurais Artificiais.

ROC Curva Característica de Operação do Receptor.

SGD Gradiente Descendente Estocástico.

SLP Perceptron de Camada Única.

VN Verdadeiro Negativo.

VP Verdadeiro Positivo.

Capítulo 1

Introdução

Em 1936, o matemático inglês Alan Turing [1] publicou seu modelo matemático capaz de realizar procedimentos, chamado de Máquina Universal de Turing. Nas décadas seguintes, várias máquinas computacionais foram propostas com base nesse modelo matemático, expandindo a capacidade de processamento das máquinas anteriores ao modelo de Turing. Por volta de 1950, Turing publicou seu artigo *Computing Machinery an Intelligence* [1] que revolucionou a discussão sobre a capacidade de uma máquina aprender, através de seu famoso *Imitation Game*, e propôs que máquinas no futuro poderiam realizar atividades até então somente possíveis pelo cérebro humano. Uma subárea do estudo de inteligência artificial denominada aprendizado de máquina, termo proposto por Samuel [2], vem sendo amplamente estudada por uma imensa gama de pesquisadores até os dias atuais.

Muitos avanços na área de aprendizado de máquina foram propostos, como por exemplo, as Rede Neurais Artificiais (RNAs). Baseado em modelos matemáticos e descrições fisiológicas do cérebro, o primeiro modelo de neurônio artificial foi concebido por Warren McCulloch e Walter Pitts em 1943 [3]. Posteriormente, Rosemblatt [4] elabora o primeiro modelo de Rede Neural conhecido como Perceptron. Naquele momento, sistemas computacionais eram baseados fortemente nas estruturas fisiológicas do cérebro de animais e, portanto, muitos avanços foram possibilitados pela interdisciplinaridade entre as áreas de medicina e computação.

Em 1972, o sistema MYCIN [5] foi desenvolvido utilizando as técnicas de aprendizado de máquina para diagnosticar doenças infecciosas por amostras de sangue. Este foi o primeiro sistema de Diagnóstico Auxiliado por Computador (CAD), do inglês *Computer aided diagnosis*, a ter impacto na área da saúde e, assim, o CAD tornou-se uma realidade. Após a década de 70 e até hoje, vários outros sistemas de CAD foram propostos para as mais diversas áreas da saúde. Focado na detecção de câncer de mama, o primeiro sistema de CAD comercializado foi o ImageChecker [6]. Daí em diante, o reconhecimento de padrões em imagens médicas virou uma realidade e vários sistemas de CAD começaram a

destacar-se no mercado, muitos deles inclusive, possuindo uma taxa de acerto no diagnóstico superior a de médicos. Por exemplo, Haenssle et al. [7] demonstra que os sistemas baseados em aprendizado de máquina obtém um melhor desempenho no reconhecimento de melanoma em imagens dermatoscópicas que seres humanos.

Atualmente, com o desenvolvimento das técnicas de um subconjunto de aprendizado de máquina chamado de aprendizado profundo, do inglês *Deep Learning*, o emprego de sistemas de CAD virou uma tendência em várias áreas da medicina. Uma classificação razoável para diferenciar os tipos de problemas abordados pela área de aprendizado de máquina é:

1. No aprendizado supervisionado busca-se comparar as entradas dos modelos de aprendizado de máquina com saídas pré-estabelecidas a fim de tornar estes modelos autosuficientes na rotulação das entradas propostas;

2. No aprendizado não-supervisionado busca-se encontrar padrões entre as diversas entradas de um modelo de RNA e separá-las de acordo com os padrões detectados;

3. No aprendizado por reforço busca-se melhorar o desempenho de um agente computacional, baseado nas respostas recebidas no ambiente em que ele está inserido.

Sistemas de CAD vêm sendo propostos todos os dias para auxiliar na resolução de problemas médicos. Portanto, o presente trabalho faz a proposta de um sistema de CAD para a detecção precoce do melanoma. Como descrito por Adegun et al. [8], vários sistemas de CAD baseados nas técnicas de *Deep Learning* foram propostos para a classificação de câncer de pele, inclusive o melanoma. Neste trabalho é feita a comparação de técnicas de Deep Learning utilizadas nestes sistemas de CAD a fim de avaliar sua real capacidade em facilitar o diagnóstico precoce de melanoma.

1.1 Definição do problema

Dados do Instituto Nacional de Câncer (INCA) [9] apontam que o câncer de pele é o tipo de câncer com maior incidência no Brasil, 33% dos casos de câncer são relacionados a doenças de pele. Entre os tipos de câncer de pele existentes, o melanoma é o tipo mais grave e o que tem a maior probabilidade de evoluir para metástase. De acordo com pesquisas feitas pelo INCA, para 2020, foi prevista a ocorrência de 176.930 casos de câncer de pele do tipo não-melanoma e 8.450 casos de melanoma, sendo assim, 185.380 casos no total. O prognóstico para câncer de pele é considerado bom quando em fase inicial, evidenciando a necessidade de uma detecção precoce desta doença. Com base nessas informações, a

construção e o aprimoramento de sistemas de CAD que possibilitem a detecção rápida e precisa deste tipo de doença é imprescindível.

Dentre técnicas de *Deep Learning*, a Rede Neural Convolucional (CNN), do inglês Convolutional Neural Network, é a mais eficiente para o reconhecimento de padrões em imagens. Por isso, esta técnica que obtém resultados excelentes em uma vasta gama de problemas vem sendo amplamente utilizada em áreas correlatas a medicina. Com seu uso, é possível aumentar drasticamente a probabilidade de um diagnóstico precoce, resultando em um prognóstico favorável ao paciente. Trabalhos como de Acosta et al. [10], Pham et al. [11], Hosny et al. [12], demonstram a robustez da aplicação destas técnicas que, em conjunto com os conhecimentos médicos, atingem uma precisão na detecção precoce de câncer maior que a somente com seres humanos, como demonstrado por Haenssle et al. [7].

Adentrando o universo das CNNs, duas técnicas propostas na literatura vêm se destacando: *transfer learning* e *data augmentation*. O conjunto destas técnicas aplicadas em CNNs compõe o estado da arte atual para reconhecimento de padrões em imagens. No presente trabalho, será proposto um sistema de CAD que explora os limites deste conjunto de técnicas com o intuito de avaliar se sua utilização consegue prover uma boa solução para os objetivos descritos na próxima seção.

1.2 Objetivos

O presente trabalho tem como objetivo geral realizar experimentos relacionados ao reconhecimento de padrões em imagens dermatoscópicas para a detecção de melanoma utilizando modelos de CNN. Para alcançar o objetivo citado, definiu-se os seguintes objetivos específicos:

1. Verificar se a utilização da técnica de *transfer learning*, para a base de dados definida, promove um aumento no desempenho de CNNs;

2. Verificar se a técnica de *data augmentation*, para a base de dados definida, promove um aumento no desempenho de CNNs;

3. Verificar se a utilização combinada das técnicas de *data augmentation* e *transfer learning*, para a base de dados definida, conseguem promover um aumento da acurácia balanceada de CNNs;

4. Verificar a influência do desbalanceamento das classes, na base de dados definida, na capacidade preditiva de CNNs.

1.3 Estruturação

Este trabalho é constituído dos seguintes capítulos:

- O **Capítulo 2** fornece o arcabouço teórico para a realização dos experimentos, apresentando os conceitos que permeiam as áreas do conhecimento de RNAs e CNNs. Após isso, descreve os modelos de CNNs e métricas de avaliação utilizados durante os experimentos;
- O **Capítulo 3** descreve os artigos científicos que serviram de inspiração e base teórica para a proposta do presente trabalho;
- O **Capítulo 4** apresenta e descreve a formulação dos experimentos propostos pelo presente trabalho;
- O **Capítulo 5** apresenta e descreve os resultados obtidos pelos experimentos propostos a partir das métricas de avaliação pré-definidas;
- O **Capítulo 6** propõe uma discussão sobre os resultados obtidos e trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Esse capítulo discorre sobre os principais conceitos teóricos necessários para um bom entendimento do presente trabalho. Nas Seções 2.1, 2.2 e 2.3 são apresentados os conceitos de aprendizado de máquina, fundamentos de redes neurais artificiais e redes neurais convolucionais respectivamente. Na Seção 2.4 o referencial teórico para o processo de treinamento é apresentado. Na Seção 2.5 as arquiteturas Rede Neural Convolucional (CNN) utilizadas são apresentadas. Nas Seções 2.6 e 2.7 as técnicas a serem analisadas, *transfer learning* e *data augmentation* são discutidas e por fim, na Seção 2.8 são elencadas as métricas de avaliação de desempenho que serão utilizadas.

2.1 Aprendizado de Máquina

A contar do início da teoria da computação, há uma necessidade de se chegar às fronteiras do conhecimento humano sobre Máquinas de Turing e seus poderes computacionais. Desde essa época, compara-se a capacidade de processamento do cérebro humano com a de uma máquina em problemas relacionados ao aprendizado e a tomada de decisões. Uma discussão importante sobre os limites de sistemas computacionais é feita por Turing et al. [1] na qual é discutido se máquinas conseguiriam pensar ou aprender. Com o avanço das técnicas relacionadas ao aprendizado de máquina, o que se pensava impossível antigamente virou realidade em relação a capacidade de aprendizado e reconhecimento de padrões de uma máquina.

De forma similar aos seres humanos, podemos dizer que um sistema computacional aprendeu a realizar uma tarefa se, ao completar uma etapa de treinamento, obtém um melhor desempenho na execução dessa tarefa novamente. Uma definição mais técnica é proposta por Mitchell [13] que relaciona a tarefa, a experiência e o desempenho do sistema computacional, mas a definição aqui citada contém um nível de abstração que fornece arcabouço suficiente para o desenvolvimento do presente trabalho.

2.2 Redes Neurais Artificiais

Rede Neurais Artificiais (RNAs) surgem como modelos que simulam o funcionamento da esquemática de um cérebro humano, ou seja, um conjunto de neurônios que trafegam informação por suas estruturas. No contexto de RNA, o corpo neuronal é abstraído como um nó em um grafo e seu axônio abstraído como uma aresta. Obviamente, essas abstrações não contemplam todas as funções que um neurônio desempenha na passagem de pulsos elétricos, elas são um modelos simplificados do processo. Mitchel [13], afirma que várias complexidades de um cérebro humano não são representadas pelos modelos de RNA, ainda assim esses modelos conseguem resultados excelentes na resolução de problemas de forma similar ao cérebro humano.

O Perceptron, também conhecido como Unidade de Limite Linear (LTU), do inglês *Linear Threshold Unit*, é um modelo inicial de RNA proposto por Rosenblatt [4]. Por ser um modelo simples, é possível descrever facilmente suas características com o objetivo de ilustrar a elaboração de uma rede conforme apresentado na Figura 2.1.

A arquitetura de um Perceptron pode ser caracterizada da seguinte forma:

1. Entradas e saídas são representadas por valores;
2. Os neurônios de entrada, do inglês *input neurons*, apenas retornam os valores passados na própria entrada;
3. Os neurônios de viés (b), do inglês *Bias*, são inicializados com valores independentes da entrada;
4. Há arestas com pesos saindo de todos os neurônios de entrada para a LTU;
5. O Perceptron computa o somatório de pesos (w_i) multiplicados por entradas (x_i), como descrito na Equação (2.1);
6. Em seguida, aplica uma função degrau, do inglês *step function*, e retorna como saída seu resultado.

$$z = b + w_1 * x + 1 + w_2 * x_2 + ... + w_n * x_n \tag{2.1}$$

A função de ativação, na Figura 2.1 mostrada como a função degrau, define qual é a saída após o somatório realizado. Géron [14], afirma que a função de ativação mais comum para Perceptrons é chamada de função degrau Heaviside, do inglês *Heaviside Step Function*. Portanto, esse Perceptron computa a combinação linear das entradas e, se o resultado ultrapassar o limiar, retorna a classe positiva, caso contrário a classe negativa.

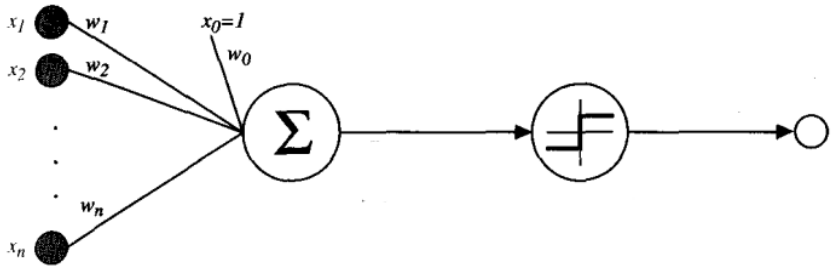

Figura 2.1: Estrutura de um Perceptron mostrando as entradas x_i, os pesos w_i, o neurônio de bias x_0 inicializado com valor 1 e sua aresta w_0 e, por fim, passando pelas etapas de somatório e de ativação (Fonte: [13]).

Assim, torna-se possível a realização de classificações binárias com essa estrutura. **Perceptron de Camada Única (SLP)** é a combinação de Perceptrons em uma só camada, como mostrado na Figura 2.2. Combinando vários Perceptrons dessa forma, é possível classificar uma entrada em k classes distintas.

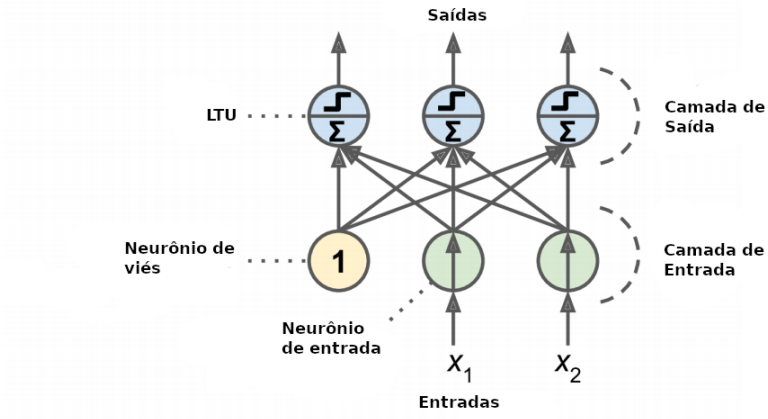

Figura 2.2: Representação da estrutura de um Perceptron de Camada Única mostrando todos os seus componentes (Fonte: [14]).

Para uma RNA ter um processo de aprendizado - nesse contexto, atualizar os pesos de suas arestas - e passar a apontar corretamente um padrão, é necessário um processo de treinamento. O algoritmo proposto por Rosemblatt [4] para treinamento de Perceptrons dessa magnitude é o seguinte: o Perceptron é alimentado com uma instância de treino por vez, para cada instância ele faz suas predições e, por fim, em cada neurônio de saída que produziu uma predição incorreta são reforçados os pesos das conexões que provavelmente contribuíram para uma predição correta. Ainda Rosemblatt [4], provou que se as instâncias de treino são linearmente separáveis, esse algoritmo de treinamento utilizando Perceptrons convergiria para uma solução, o chamado Teorema da Convergência

de Perceptrons, e portanto um perceptron simples é capaz de representar algumas funções lógicas como AND, OR e NOT.

Perceptron Multicamadas (MLP), do inglês *Multi-Layer Perceptron*, é uma RNA formada por múltiplos Perceptrons. Esse tipo de RNA é um bom exemplo para introduzir o uso de camadas escondidas e o conceito de camadas totalmente conectadas, haja vista que uma MLP tem a estrutura de uma rede com essas características. Levando em conta a Figura 2.3, é observado que existem vértices de entrada, vértices de viés, vértices com LTUs e que se tem três camadas: uma camada com as entradas e duas camadas de Perceptrons. Uma das camadas de Perceptrons pode ser chamada de camada escondida, do inglês *hidden layer*, é ela que define a profundidade da rede, já a outra camada, nesse exemplo, é a camada que comporta os Perceptrons responsáveis por retornar a saída dessa rede, ou camada de saída. Outra observação é que todos os vértices de uma camada têm arestas apontando para todos os vértices da próxima.

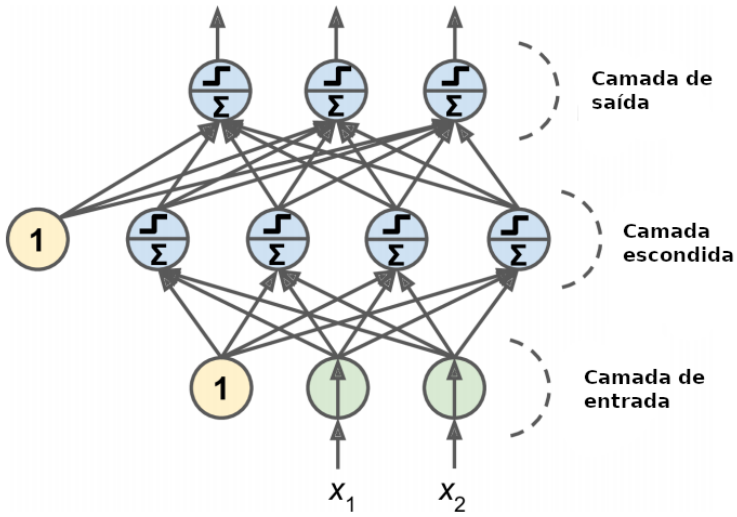

Figura 2.3: Esquemática de um Perceptron Multicamadas. Como descrito no paragrafo anterior, x_1 e x_2 são as entradas, as setas indicam os neurônios de entrada, em amarelo os neurônios de viés, em azul as LTUs. Também é mostrada distribuição das camadas. (Fonte: [14]).

Portanto, a MLP é um grafo direcionado com suas camadas totalmente conectadas, de forma que a informação percorre somente em um sentido. Essa característica dá a essa RNA a classificação de rede neural de alimentação para frente, do inglês *feed-foward neural network*. Uma RNA com mais de duas camadas escondidas é geralmente enquadrada como uma DNN e, portanto, uma MLP pode ser considerada uma DNN se for composta pela característica citada anteriormente.

Aprendizado profundo remete ao treinamento de DNNs para que elas possam, através desse treinamento, obter predições melhores relacionadas a um problema proposto. Contudo, o treinamento de DNNs requer um algoritmo muito bem elaborado, haja vista que quanto maior a quantidade de camadas escondidas e de LTUs, maior a quantidade de pesos a serem ajustados.

Rumelhart et al. [15], propuseram um algoritmo chamado de Retropropagação de erros (BP), do inglês *Backpropagation of errors*, para o treinamento de MLP. Como descrito por Mitchell [13], esse algoritmo permite o treinamento eficaz de RNAs aplicando algum algoritmo de Descida do Gradiente, do inglês *Gradiente Descent*, na tentativa de minimizar a média do quadrado dos erros, do inglês *mean squared error*, entre a saída da MLP e os valores alvo para essas saídas. Esse processo é genericamente dito como o cômputo do gradiente, em relação aos erros, para minimizar a função de custo, no caso anterior a *Mean squared error*.

O cálculo de Descida do Gradiente necessita que as LTUs de uma MLP tenham uma função de ativação não-linear, e, portanto, a função degrau precisa ser modificada para uma alternativa não-linear como as descritas na Seção 2.3.2. Agora, com um algoritmo bem definido para seu treinamento, as MLP podem ser utilizadas para a classificação binária de entradas mais complexas do que funções lógicas simples, haja vista que a função de ativação foi modificada e agora passa a apontar em cada neurônio de saída a probabilidade da entrada estar ou não em uma classe.

Por fim, diversos tipos de arquiteturas foram propostos pela comunidade científica ao passar dos anos para os mais diversos tipos de problemas utilizando modificações de MLP. Por exemplo, Acosta et al. [10] usaram CNN para detecção precoce do melanoma, já Mittal et al. [16] consideram CNN para a classificação da morfologia de galáxias. No presente trabalho, o foco é direcionado para as CNNs na detecção e reconhecimento de padrões. Suas características e aplicações serão descritas nas próximas seções.

2.3 Redes Neurais Convolucionais

Um problema específico em que redes neurais foram aplicadas é o do reconhecimento de padrões em imagens. Para essa finalidade, uma arquitetura bem difundida no campo de aprendizado profundo é a CNN.

A primeira CNN publicada foi a Neocognitron, proposta por Fukushima et al. [17]. Essa rede se diferenciava de uma MLP em premissas básicas e, portanto, obtinha resultados superiores. A primeira premissa é que uma imagem é um arranjo espacial mais do que um vetor de *pixels* e, tratando como tal, a imagem poderia ser representada como matriz. Em consequência, as camadas internas da rede neural foram arquitetadas como

planos compostos de células neuronais. Foram propostas as camadas de convolução e as de redução de taxa de amostragem, do inglês *downsampling*. A primeira é composta por neurônios que escaneiam padrões nas imagens e a segunda por neurônios que calculam operações de máximo nos planos retornados pelas camadas de convolução. A segunda premissa é baseada nos estudos de Hubel et al. [18], que avaliaram as células córtex visual dos gatos chegando a conclusão que no reconhecimento de imagens haviam duas etapas de processamento distintos, células S respondiam ao sinal e células C confirmavam a respostas das células S, gerando assim uma estrutura hierárquica que foi seguida na Neocognitron.

A arquitetura LeNet proposta por LeCun et al. [19] foi o esqueleto inicial para as CNNs existentes hoje em dia. Os autores propuseram o treinamento da rede pelo algoritmo de BP. Nessa rede, além das camadas convolucionais e das camadas de *downsampling*, uma MLP é colocada na saída da última camada de convolução para realizar a classificação das imagens.

Por fim, agora os componentes básicos presentes em uma CNN são: 1. Funções de ativação, 2. Camadas Convolucionais, 3. Camadas de *downsampling* mais especificamente *pooling*, 4. Camadas totalmente conectadas compostas por MLP. Estes componentes serão descritos nas seções seguintes.

2.3.1 Camadas Convolucionais

Camadas convolucionais são assim chamadas pois executam uma operação matemática chamada de convolução sobre uma imagem de entrada. Convolução em si, considerando o domínio discreto, é uma transformação linear descrita pela Equação (2.2).

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n). \tag{2.2}$$

Na terminologia de CNN, a matriz I é chamada de entrada, a matriz K é chamada de núcleo ou filtro, do inglês *Kernel*, a saída *S(i, j)* as vezes referenciada como mapa de características, do inglês *feature map*, i e j as coordenadas do elemento de entrada e m e n as coordenadas do elemento do filtro, como descrito por Goodfellow et al. [20].

Relembrando a Seção 2.3, essas camadas são formadas por planos. Esse planos doravante serão chamados de filtros. Um filtro simples para detectar linhas horizontais pode ser visto como uma matriz 3×3, mostrado na Figura 2.4, tratando a convolução aplicada em funções de domínio discreto. Como mostra a imagem da Figura 2.5 adaptada de Goodfellow et al. [20], esse filtro é aplicado nas submatrizes de uma matriz de entrada resultando em uma outra matriz de saída com menor dimensão e pesos reforçados.

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Figura 2.4: Proposta de um filtro para a camada convolucional em domínios discretos.

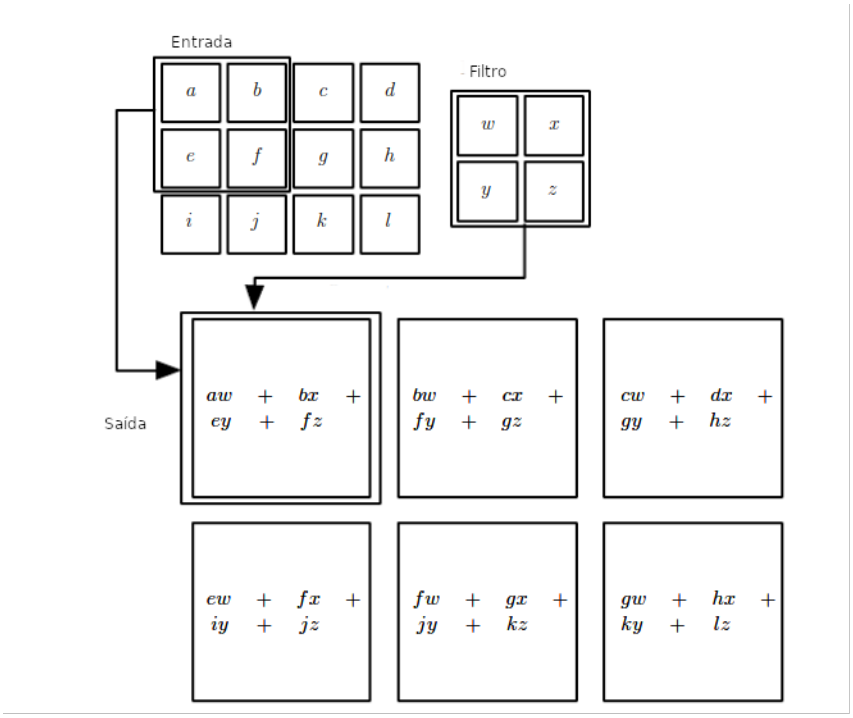

Figura 2.5: Convolução entre matrizes detalhada: a até l são elementos da entrada I, w até z são elementos do filtro K e os elementos da matriz de saída S são resultado da soma da multiplicação entre elementos de entrada e do filtro. (Fonte: [20]).

Levando em conta que podem ser propostas várias camadas de convolução, para manter o tamanho das saídas pode ser empregada uma técnica de preenchimento com zeros, do inglês *Zero Padding*. Essa técnica é definida, no contexto de CNNs, como o preenchimento das bordas de uma matriz de saída da convolução com zeros até que essa matriz tenha as mesmas dimensões que a matriz de entrada.

2.3.2 Funções de Ativação

Seguindo a lógica do Perceptron, após aplicada a convolução na imagem de entrada, aplica-se uma função de ativação não-linear. Existem vários tipos de funções de ativação

utilizadas na literatura de RNA. Nesta subseção, dois tipos de função de ativação serão descritas, uma função utilizada nas camadas escondidas de convolução e outra utilizada nas camadas de saída da rede, pois são os tipos utilizados no presente trabalho.

Unidade Linear Retificada (ReLU), do inglês *Rectfied Linear Unit*, é um tipo de função de ativação não-linear muito utilizada nas camadas escondidas de uma RNA. Ela pode ser descrita pela Equação (2.3) e seu gráfico, adaptado de Mittal et al. [16], é mostrado na Figura 2.6.

$$ReLU(x) = max(0, x). \tag{2.3}$$

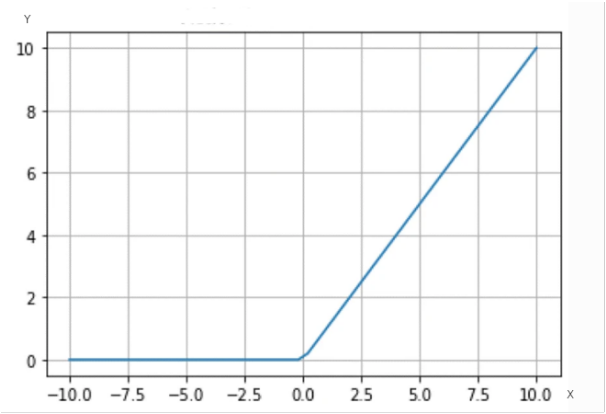

Figura 2.6: ReLU: eixo x são os valores das entradas e y são as saídas retornadas (Fonte: [16]).

A função ReLU recebe o maior valor entre os valores 0 e a entrada x, sendo a operação *max(0,x)* a responsável pelo retorno desse maior valor. Esta função de ativação possui uma não-linearidade bem limitada e é idêntica a função identidade para valores positivos.

A derivada da ReLU, Equação (2.4), demonstra que seu gradiente sempre terá uma direção não nula, ao contrário das funções de ativação sigmoidais, e, contando que funções de ativação sigmoidais saturam a partir de um ponto, a ReLU é uma melhor escolha no cálculo de descida do gradiente. Com isso, Géron [14] afirma que ReLU é a mais amplamente utilizada função de ativação nas camadas internas das diversas RNAs já propostas.

$$ReLU'(x) = \begin{cases} 1, & \text{se } x \geq 0 \\ 0, & \text{c.c.} \end{cases}. \tag{2.4}$$

Função Softmax é outro tipo função de ativação normalmente utilizada na camada de saída de uma MLP em problemas de classificação. Goodfellow et al. [20] apontam como característica principal do uso dessa função a sua capacidade de retornar a distribuição

de probabilidade sobre k classes. Por essa característica, ela é amplamente utilizada em classificação multiclasses:

$$s_k(x) = \theta_k^T * x. \tag{2.5}$$

Primeiramente, como descrito por Géron [14], é necessário calcular a pontuação para cada classe k em uma instância x. O gráfico desta função, adaptado de Mittal et al. [16], é mostrado na Figura 2.7.

Portanto, k é uma classe, x um instância da rede, θ_k^T o vetor de parâmetros dedicado para a classe k e T a operação vetorial de transposição. Agora que cada classe tem uma pontuação, a função Softmax é aplicada, Equação (2.6), para calcular a probabilidade estimada p_k de uma instância x estar em uma classe de pontuação $s_k(x)$.

$$p_k = \sigma(\mathbf{s}(\mathbf{x}))_k = e^{s_k(\mathbf{x})} \div \sum_{j=1}^{K} e^{s_j(\mathbf{x})}. \tag{2.6}$$

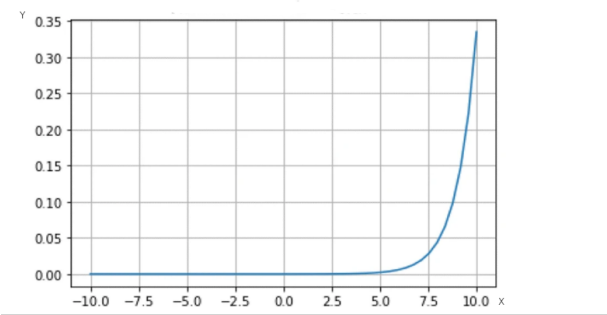

Figura 2.7: Ativação Softmax: eixo x representa os valores das entradas e eixo y as probabilidades de saída retornadas (Fonte: [16]).

2.3.3 Camadas de *Pooling*

Camadas de *downsampling* são muito utilizadas nas arquiteturas de CNNs. A técnica de *Pooling* é uma das técnicas mais usadas para realizar o *downsampling*. Quando aplicada essa técnica, as camadas passam a se chamar camadas de *Pooling*. Goodfellow et al. [20] afirmam que essas camadas tem como objetivo reduzir as *feature maps* de forma a gerar uma matriz das dimensões desejadas para a próxima camada. A aplicação dessa técnica é essencial na redução do custo computacional das CNNs, afinal, ela reduz o número de neurônios e, portanto, o de parâmetros a serem ajustados no treinamento dessas redes. Camadas de *Pooling* são utilizada para realizar o *downsample* de *feature maps*.

Camadas de *Pooling* são camadas compostas por filtros que realizam operações matriciais sobre as *feature maps* com o objetivo gerar matrizes de saída com dimensões

reduzidas. Esses filtros são matrizes de neurônios como nas camadas convolucionais, exceto que os neurônios de *Pooling* não têm pesos, sua única função é agregar as entradas aplicando técnicas como *Max Pooling*, *Average Pooling*, entre outras.

Max Pooling* e *Average Pooling são duas funções utilizadas para realizar o *pooling*, como mostra a Figura 2.8. Ambas recebem a matriz de entrada, aplicam um filtro e reduzem as suas dimensões para a dimensão desejada. A primeira, ao aplicar esse filtro, retorna na matriz de saída o maior valor dentre os neurônios na região em que o filtro é aplicado. A segunda, retorna a média entre os valores da região onde é aplicado o filtro na matriz de saída.

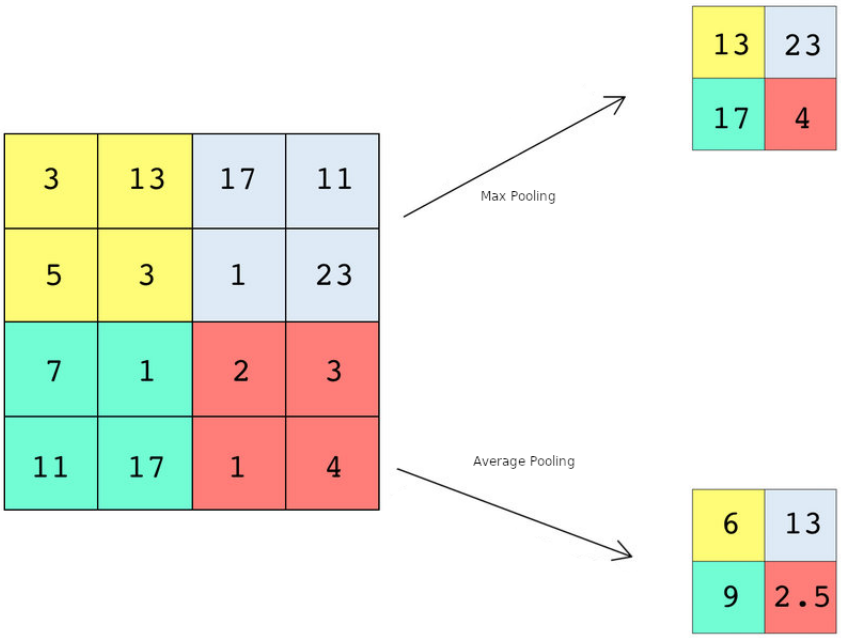

Figura 2.8: *Max Pooling* e *Average Pooling*: a primeira retorna o maior valor das submatrizes destacadas e a segunda retorna a soma dos valores sobre a quantidade de elementos das submatrizes destacadas (Fonte: [21]).

2.3.4 Camadas Totalmente Conectadas

Como citado na Seção 2.3, a MLP pode ser combinada com camadas convolucionais para gerar uma estrutura atual de CNN. Quando utilizadas na construção de CNNs, essas MLP são chamadas de camadas totalmente conectadas.

O objetivo é simples, fazer a classificação da entrada após uma série de camadas de detecção de padrões. Algumas arquiteturas atuais podem utilizar camadas totalmente conectadas com outros objetivos, mas são casos particulares que serão abordados pelo presente trabalho, se ocorrerem, nas descrições de suas respectivas estruturas. Como as

camadas convolucionais tratam as entradas como matrizes, é preciso vetorizar a saída da camada anterior à camada totalmente conectada para que ela possa realizar a classificação da entrada.

2.4 Treinamento

Como descrito na Seção 2.3, o treinamento de CNNs é normalmente feito utilizando o algoritmo de BP. A lógica por trás da Descida do Gradiente utilizada nesse algoritmo é reajustar os pesos para minimizar a função de custo, do inglês *cost function*. No presente trabalho, a demonstração matemática dos cálculos dos vetores de gradiente e derivadas direcionais não são parte dos objetivos propostos, portanto, definições intuitivas das operações do Gradiente Descendente fornecem a abstração necessária.

Na Seção 2.4.1, serão descritas as variações do algoritmo de Gradiente Descendente e o otimizador *Adam* para essas variações. Logo após, na Seção 2.4.2, será discutida a função de custo utilizada em CNNs para classificação binária e multiclasses.

2.4.1 Otimizadores

De forma intuitiva, Géron [14] descreve as iterações do Gradiente Descendente da seguinte forma:

1. Inicializa-se o vetor de parâmetros v com valores aleatórios, técnica conhecida como inicialização randômica, do inglês *random initialization*;

2. É feito o cálculo do gradiente local da função de erro com relação ao vetor de parâmetros v;

3. Desloca-se, passo por passo, em direção descendente até que seja encontrado um mínimo local, do inglês *local minimum*.

Com relação as técnicas de RNAs, cada passo executado pelo algoritmo da Descida do Gradiente tem um tamanho definido por um parâmetro da rede chamado taxa de aprendizagem, do inglês *learning rate*. É necessário ressaltar, também, que o melhor caso para a otimização proposta pelo algoritmo de Gradiente Descendente é que ele vá a convergir para o mínimo global, mais do que o mínimo local, e, portanto, é necessária uma escolha criteriosa da função de custo. As funções de custo mais utilizadas tem um gráfico convexo, justamente por facilitar a convergência desse algoritmo para um mínimo global, portanto vários otimizadores são baseados nessa característica.

Ruder [22], afirma que considerando uma RNA com milhões de parâmetros podendo consumir uma quantidade de memória maior que o disponível em um sistema computacional, o Gradiente Descendente pode se tornar muito lento, haja a vista que esse algoritmo para uma única mudança de parâmetros tem que calcular o gradiente de toda a base de dados. Por esse motivo, foram propostos algumas modificações, na literatura de RNAs, para o Gradiente Descendente com o intuito de melhorar o desempenho desse algoritmo quando um vetor de parâmetros grande o suficiente é utilizado.

Gradiente Descendente Estocástico (SGD), do inglês *Stochastic Gradient Descent*, é um algoritmo proposto para a otimização do Gradiente Descendente que aplica uma atualização de parâmetros, aleatoriamente, em somente uma instância de treino a cada passo, melhorando o desempenho do algoritmo de Gradiente Descendente e reduzindo seu custo computacional. Géron [14] afirma que o SGD tem esse melhor desempenho por manter em memória somente essa única instância de treinamento, facilitando o treinamento de RNAs com vetores de peso muito grandes.

Gradiente Descendente Mini-Batch é uma adaptação ao SGD. Ao invés de levar em conta somente uma instância de treino por passo, o algoritmo seleciona um conjunto de instâncias de treino, diminuindo assim a aleatoriedade do SGD e melhorando a sua convergência para um mínimo, dado que, por sua aleatoriedade, o SGD tem dificuldades em convergir para um mínimo local como o Gradiente Descendente. O otimizador citado aqui, é uma proposta de adaptação para algoritmos estocásticos relacionados ao SGD a fim de acelerar a convergência desses algoritmos a um mínimo local ou global.

Adam, do inglês *adaptive moment estimation*, é o estado-da-arte para algoritmos de otimização do SGD. Géron [14], descreve o algoritmo como a junção de duas características baseadas nos algoritmos de otimização de *Momentum* e *RMSProp*. Ainda Géron, descreve o conjunto desses algoritmos da seguinte forma:

1. *Momentum Optimization* mantém um registro da média do decaimento exponencial dos gradientes passados;

2. *RMSProp* mantém um registro da média do decaimento exponencial das raízes dos gradientes passados.

Com essa combinação de algoritmos, o *Adam* calcula os *learning rates* adaptativos para cada parâmetro. Ruder [22], afirma que a otimização de *Momentum* acelera o SGD em direções relevantes, ou seja, para o mínimo local ou global, e diminui as oscilações no caminho. Géron [14], afirma que o *RMSProp*, melhora o desempenho da técnica de *learning rate* adaptativo acelerando o vetor gradiente quando as curvas dos gradientes são íngremes e, também, quando elas são suaves, levando o SGD a uma convergência mais rápida.

2.4.2 Função de custo *Cross-entropy*

Ainda sobre o algoritmo de Gradiente Descendente, como citado na Seção 2.4.1, é necessário que a escolha de uma função de custo seja bem planejada para que ela possa comparar as saídas das RNAs com os valores alvo e penalizar as instâncias de treino que retornaram uma predição incorreta.

Entropia Cruzada, do inglês *Cross-entropy*, é uma das funções de custo utilizadas em CNNs para classificação binária e multiclasses. Ela é bastante utilizada em modelos de RNAs que tem como saída probabilidades de uma entrada pertencer ou não à uma determinada classe, podendo assim ser chamada também de função de custo logística, do inglês *logistic loss*.

Quando a probabilidade retornada por uma instância de treinamento diverge da probabilidade alvo para uma classe, essa instância é penalizada pela função de custo. A *logistic loss* retorna valores altos para predições incorretas e baixas para predições corretas, como mostra a Figura 2.9.

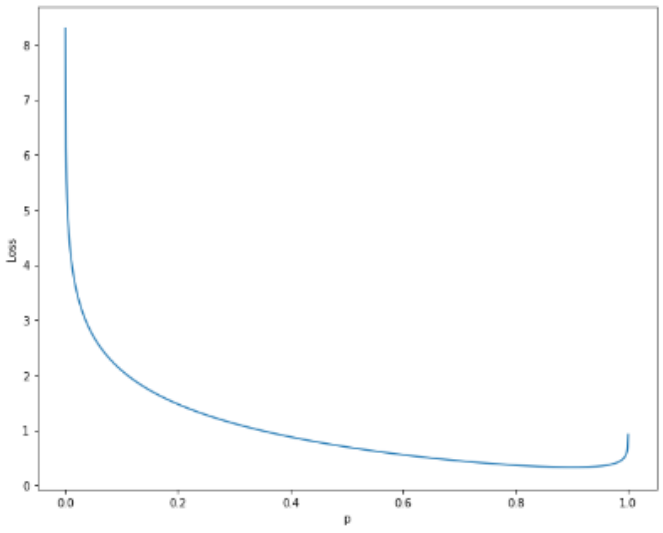

Figura 2.9: Gráfico de *Loss* por probabilidade p sendo que o valor alvo para a classe positiva é 1 (Fonte: [23]).

2.5 Arquiteturas CNN

Como discutido na Seção 2.3, redes neurais convolucionais contêm diferentes mecanismos para manipular e realizar cálculos sobre um dado fornecido como entrada com o objetivo de gerar uma saída. Estratégias como a utilização de camadas convolucionais, funções de ativação, técnicas de *pooling* e camadas totalmente conectadas são combinadas de

forma a gerar um conjunto de transformações nos dados de entrada. A combinação dessas diversas técnicas de manipulação dos dados, em uma ordem específica, seguindo uma estrutura definida, leva o nome de arquitetura CNN [24]. Diversas arquiteturas foram propostas no decorrer dos anos e novas continuam sendo criadas [24]. Apenas três arquiteturas CNNs serão utilizadas nesse trabalho: Alexnet, GoogLeNet e ResNet. Todas elas possuem uma característica em comum, foram vencedoras, em edições diferentes, do Desafio de Reconhecimento da ImageNet (ILSVRC) [25], demonstrando a sua eficiência em problemas de reconhecimento de padrões em imagens por meio de aprendizado de máquina. Nas seções seguintes, é fornecido um breve contexto e uma breve descrição da estrutura básica de cada uma dessas três arquiteturas escolhidas.

2.5.1 AlexNet

A AlexNet foi proposta por Krizhevesky et al. [26] e com ela ganharam o ILSVRC em 2012, tornando-se o novo estado da arte e revolucionando a área de aprendizado profundo em um momento que as técnicas tradicionais de aprendizado de máquina estavam em evidência [27]. Os autores obtiveram na competição uma taxa de erro top-5 de 15,3%, valor significativamente menor que o segundo colocado com 26,2% [25].

Como mostrado na Figura 2.10, a AlexNet é composta por 8 camadas sendo 5 *convolutional layers* que realizam o processo de convolução seguido de *max pooling* e 3 *fully connected layers*, por fim, a saída da última camada é fornecida como *input* para uma função Softmax que fornece as probabilidades das classes. Os dados de entrada devem possuir dimensões de 227×227. A arquitetura também utiliza *data augmentation* e *dropout* para redução do *overfitting* [26].

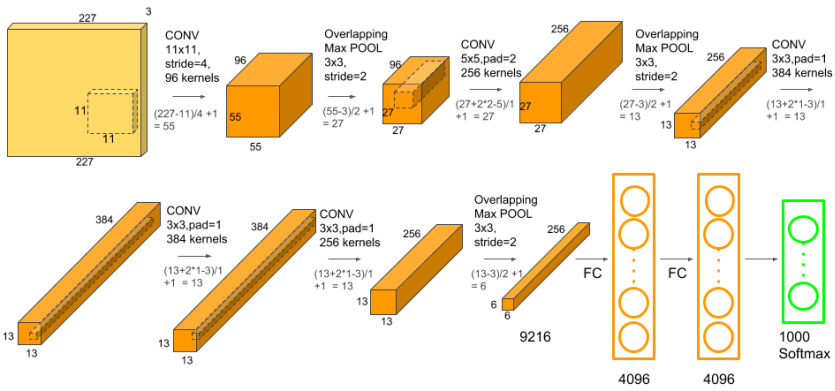

Figura 2.10: Estrutura da AlexNet (Fonte: [28]).

Um fator importante dessa arquitetura é a quantidade imensa de parâmetros, mais de 60 milhões [27] que influenciam diretamente no tempo de treinamento e utilização de recursos computacionais.

2.5.2 GoogLeNet

A arquitetura GoogLeNet foi criada por Szegedy et al. [29] e assim como a AlexNet também ganhou um desafio ILSVRC, mas na edição de 2014 [25]. A proposta da arquitetura era o desenvolvimento de um modelo que possuísse uma menor complexidade computacional em comparação com as demais CNNs existentes. A arquitetura obteve nessa edição uma taxa de erro top-5 de 6,7%, desempenhando melhor que outras arquiteturas conhecidas como AlexNet e VGG [30], apresentando, inclusive, uma melhora significativa em relação a arquitetura vencedora da edição anterior [25].

Para obter os resultados esperados, a arquitetura foi proposta como um conjunto de 22 camadas, dentre elas, camadas tradicionais de convolução e de *max pooling* e além disso, utilizou um conceito inovador de *Inceptions Layers*, como pode ser observado na Figura 2.11 que consiste na realização simultânea de convoluções, *max pooling* e aplicação de filtros dentro de uma mesma camada [25]. Para evitar um número muito grande de operações nessa etapa, dentro do módulo *inception* são utilizadas convoluções 1×1 para promover uma redução dimensional.

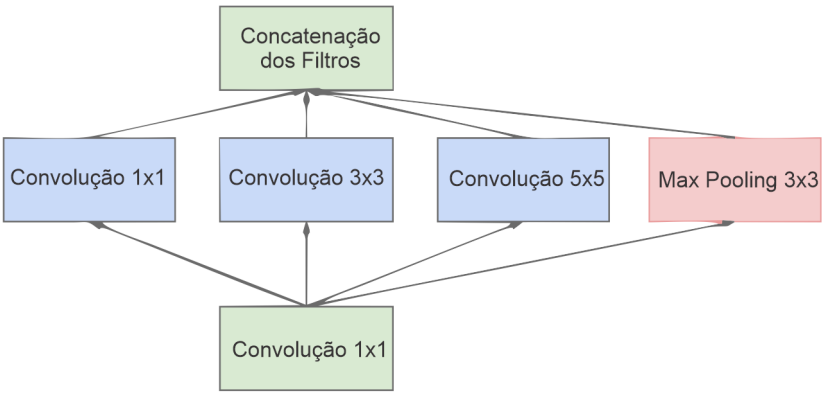

Figura 2.11: Estrutura do módulo *Inception* (Fonte: [29]).

Apesar do número expressivo de camadas, 22, a quantidade de parâmetros da arquitetura se manteve baixa em comparação, por exemplo,com a AlexNet com 60 milhões e com a VGG com 128 milhões [27].

2.5.3 ResNet

A última arquitetura que será utilizada no presente trabalho é a *ResNet*, assim como as demais citadas, também foi vencedora do desafio ILSVRC na edição de 2015, obtendo na competição uma taxa de erro top-5 de 3.57%. Importante ressaltar a variação considerável na taxa de erro top-5 para esse competição no período 2012 - 2015, o melhor resultado na edição de 2012 foi de 15,3% e em 2015 esse valor diminuiu para 3,57%, uma variação de mais de 10%, quantidade considerável considerando o tamanho da base de dados ImagetNet [31].

O modelo foi apresentado por He et al. [32] e tinha como proposta o desenvolvimento de uma rede ultra profunda que não sofresse do problema de *vanishing gradient*, característica de redes treinadas com técnicas de gradiente e *backpropagation* que, em algumas situações, devido ao tamanho muito pequeno do valor do gradiente, não é capaz de modificar os pesos da rede. A ResNet possui diversas variantes com diferentes quantidades de camadas, entretanto, a versão citada aqui possui 152 camadas e possui como característica especial a técnica de *skip connections* que consiste, de forma simplificada, em fornecer, para um grupo de camadas convolucionais a entrada da camada inicial como entrada da camada final, pulando, portanto, conexões [32]. A Figura 2.12 demonstra o procedimento da técnica de *skip connection*.

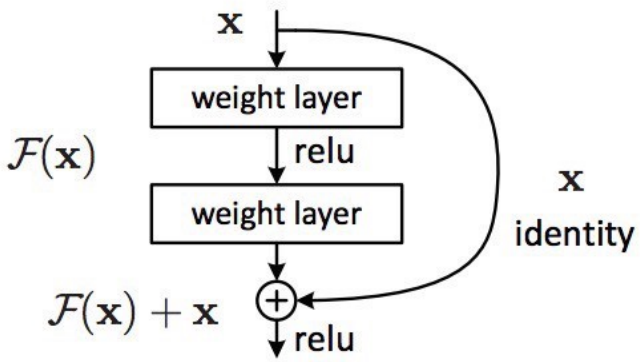

Figura 2.12: Técnica de *skip sonnection* (Fonte: [33]).

A variante Resnet-50, com 49 camadas, possui 25.5 milhões de parâmetros [32], em comparação com as arquiteturas anteriores, apesar de ser proposta como um modelo profundo, apresenta uma quantidade de parâmetros dentro da média.

2.6 Transfer Learning

O *transfer learning* surge da ideia de que os seres humanos dificilmente aprendem uma nova técnica ou habilidade do zero. Dado que um indivíduo já possui conhecimento para uma tarefa qualquer, é mais fácil para ele desempenhar uma tarefa similar, por exemplo, uma pessoa que já sabe tocar um instrumento musical tem facilidade no aprendizado de outros instrumentos, ou seja, os seres humanos conseguem reaproveitar conhecimentos de outros domínios quando tentam aprender tarefas de novos domínios.

O estudo de *transfer learning* vem se intensificando desde 1995. Durante o estágio inicial da técnica, o objetivo principal era fazer com que o modelo aprendesse as tarefas do domínio de origem e destino por meio de um treinamento simultâneo. Após 2005, o objetivo principal mudou para a transferência de conhecimento de um domínio para outro em etapas de treinamento separadas [34]. A Figura 2.13 sumariza o procedimento da técnica de *transfer learning*.

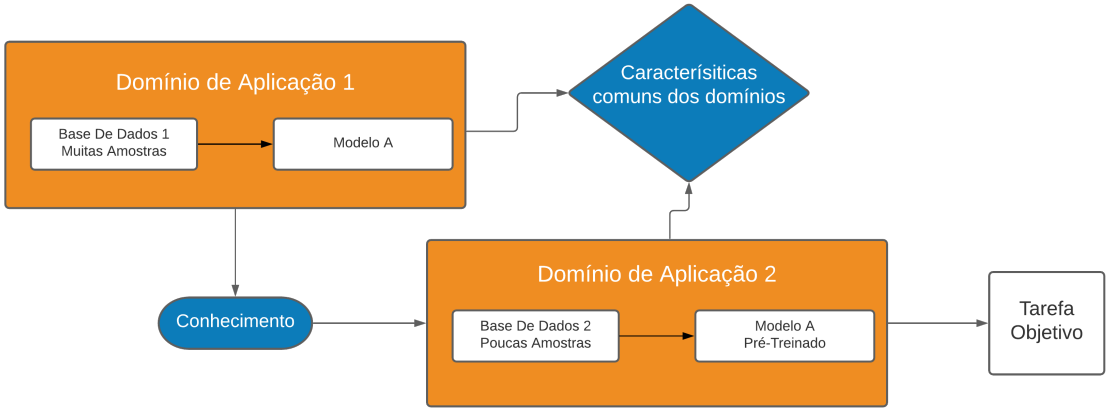

Figura 2.13: Transferência de Aprendizado.

Em um contexto de aprendizado de máquina profundo, a técnica de *transfer learning* consiste em treinar um modelo com uma base de dados para um domínio inicial, e então, utilizar essa rede pré-treinada em um domínio diferente do proposto inicialmente mas utilizando as característica aprendidas para inferir informações da nova tarefa. A técnica mostra-se muito útil principalmente em bases de dados em que pelo uma das classes é escassa, de forma que, o treinamento de uma rede neural com poucas amostras, na maioria das vezes, não é suficiente para possibilitar uma correta generalização do modelo [35], nesse sentido, a utilização da técnica possibilita que a rede seja pré-treinada inicialmente com um conjunto grande de amostras mesmo que de uma área de aplicação diferente daquela proposta como objetivo. É importante que o domínio inicial e o domínio de destino apresentem similaridades [36]. A técnica normalmente é utilizada em conjunto

com a técnica de *fine-tunning* em que as camadas mais profundas da rede após o *transfer learning* são congeladas e apenas as camadas mais superficiais são atualizadas com um novo processo de treinamento [37].

2.7 Data Augmentation

Para exemplificar a técnica de *data augmentation*, é interessante utilizar uma base de dados desbalanceada, como é o caso da base ISIC 2017 [38], constituída por imagens dermatoscópicas. Essa base sofre com o desbalanceamento das classes, e então, juntamente com *transfer learning*, a técnica de *data augmentation* é utilizada para minimizar os impactos da escassez de dados no processo de treinamento de uma rede profunda [11]. A aplicação de *data augmentation* pode ser feita por meio de diversas técnicas, as mais comuns, que vão ser utilizada no presente trabalho, são as técnicas da vertente tradicional. O procedimento consiste na replicação de imagens por meio de transformações das imagens, como por exemplo, rotação e translação. A Figura 2.14 demonstra o uso de *data augmentation* em que uma amostra Figura 2.14(A) dá origem a diversas outras Figura 2.14(B-M) por meio das técnicas descritas na Tabela 2.1.

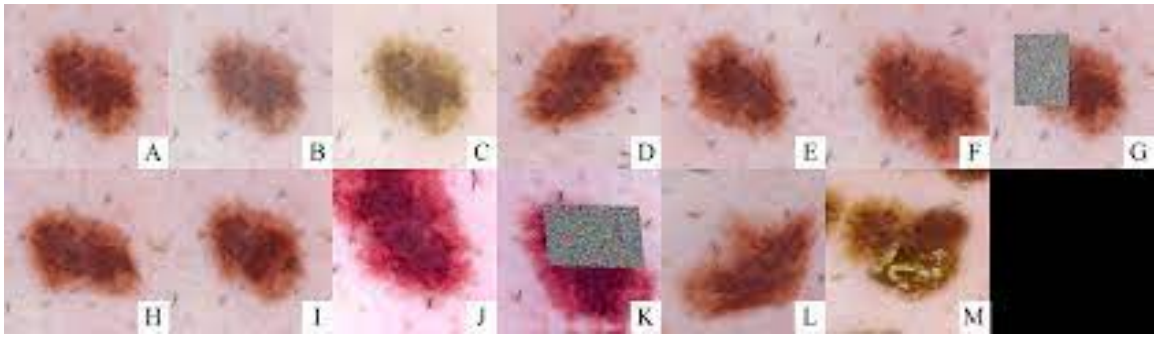

Figura 2.14: Exemplo de *Data Augmentation* sobre a Amostra A (Fonte: [39]).

2.8 Métricas de Desempenho

A avaliação do desempenho de um modelo de aprendizagem de máquina precisa ser feita de forma sistemática por meio de métricas já conhecidas na literatura que fornecem informações suficientes para indicar o quão próximo dos resultados esperados o modelo prediz. Para um modelo focado em classificação binária, as métricas mais comumente utilizadas são as seguintes: acurácia, sensibilidade, especificidade, acurácia balanceada, curva Curva Característica de Operação do Receptor (ROC) (do inglês *Receiver Operating Characte-*

Tabela 2.1: Técnicas de *data augmentation* utilizadas.

Imagem	Técnica Aplicada
A	Sem Técnica
B	Saturação/Contraste/Brilho
C	Saturação/Contraste/Brilho/Exposição
D	Rotação
E	Espelhamento
F	Recorte
G	*Erasing*
H	*Elastic*
I	Combinação de Imagens
J	Técnicas em F, D, E, C
K	Técnicas em F, G, D, E, C
L	Técnicas em F, D, H, E, C
M	Técnicas em I, F, D, E, C

ristic Curve) e Área Abaixo da Curva (AUC) (do inglês *Area Under the Curve*). As fórmulas elencadas nas seções seguintes são descritas em função dos seguintes termos [40]:

- Verdadeiro Positivo (VP): amostra da classe positiva classificada como positiva;
- Falso Positivo (FP): amostra da classe negativa classificada como positiva;
- Verdadeiro Negativo (VN): amostra da classe negativa classificada como negativa;
- Falso Negativo (FN): amostra da classe positiva classificada como negativa.

De forma a sumarizar o processo dedutivo do modelo, a matriz de confusão consegue expressar a relação entre as predições realizadas e o resultado esperado. Por meio dela é possível inferir conclusões sobre a capacidade de generalização do modelo em situações, por exemplo, em que a rede classifica corretamente amostras de uma classe e erra amostras de outra classe. A partir dela, várias métricas são formuladas. A Figura 2.15 apresenta um exemplo de matriz de confusão. A diagonal principal representa as amostras classificadas corretamente e a diagonal secundária as amostras classificadas incorretamente.

2.8.1 Acurácia

A acurácia é uma das métricas mais utilizadas, apesar de que sua utilização pode levar a conclusões equivocadas principalmente quando utilizada em um contexto de classificação de bases de dados desbalanceadas [41]. Embora não consiga, por si só, fornecer informações suficientes de eficiência de um modelo, é bastante comum devido a sua fácil aplicação

		Valores Preditos	
		SIM	**NÃO**
Valores Reais	**SIM**	VERDADEIRO POSITIVO (VP)	FALSO NEGATIVO (FN)
	NÃO	FALSO POSITIVO (FP)	VERDADEIRO NEGATIVO (VN)

Figura 2.15: Matriz de Confusão.

e entendimento [40]. Representa a quantidade de amostras classificadas corretamente em relação a quantidade de amostras totais:

$$Acurácia = \frac{VP + VN}{VP + FP + VN + FN}. \tag{2.7}$$

2.8.2 Sensibilidade

A sensibilidade representa a taxa de acerto do modelo de amostras da classe positiva em relação a quantidade total de predições [40], ou seja, conforme o modelo classifica mais amostras da classe positiva corretamente, seu valor aumenta:

$$Recall = \frac{VP}{VP + FN}. \tag{2.8}$$

2.8.3 Especificidade

Enquanto a sensibilidade representa a taxa de acerto das amostras da classe positiva, a especificidade demonstra a taxa de acerto para as amostras da classe negativa [40], ou seja, seu valor aumenta quando o modelo classifica mais amostras da classe negativa corretamente:

$$Especificidade = \frac{VN}{VN + FP}. \tag{2.9}$$

2.8.4 Acurácia Balanceada

A acurácia balanceada é uma alternativa à acurácia padrão uma vez que leva em consideração o desbalanceamento das classes [42] e, por esse motivo, será a métrica principal de avaliação dos resultados utilizada no presente trabalho:

$$AcuráciaBalanceada = \frac{Sensibilidade + Especificidade}{2} \tag{2.10}$$

2.8.5 Curva ROC e AUC

A ROC representa a relação entre a sensibilidade e a especificidade. Modelos cujas curvas se aproximam do canto superior esquerdo tendem a desempenhar melhor [43].

Juntamente a análise proporcionada pela curva ROC, está a AUC que representa a área sob a curva ROC. Algumas vezes é necessário sumarizar o desempenho de um modelo em uma única métrica para possibilitar uma comparação mais apropriada, nesse sentido, a AUC torna-se uma boa opção uma vez que reúne as características da curva ROC em um único valor [40]. A Figura 2.16 exibe um exemplo de uma curva ROC e AUC, representada pela área cinza. O gráfico é construído por meio dos valores obtidos para a sensibilidade e especificidade no conjunto de testes.

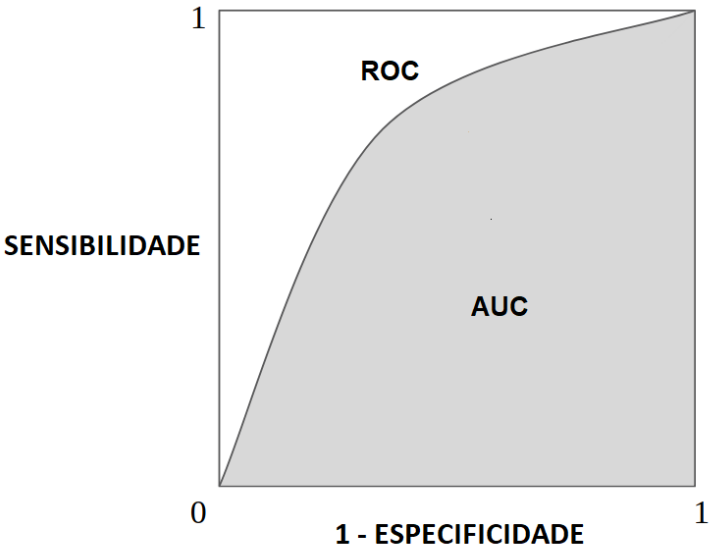

Figura 2.16: Exemplo de Curva ROC (Fonte: [44]).

Capítulo 3

Revisão de Literatura

Este capítulo reúne os principais projetos e experimentos encontrados na literatura que influenciaram o processo de elaboração do presente trabalho. As Seções 3.1 e 3.2 discorrem sobre as técnicas de *data augmentation* e *transfer learning* respectivamente, metodologias que representam o cerne do estudo desse documento. O principal objetivo aqui é determinar se existe, na literatura, indicativos de aumento de eficiência em redes profundas quando essas técnicas são utilizadas.

3.1 Data Augmentation

Perez e Wang [45] realizaram uma análise de diversas técnicas de *data augmentation*, não restringindo-se apenas a métodos tradicionais definidos pelos autores como: rotação, ampliação, deslocamento, distorção, reversão e sombreamento. Uma vez que, segundo os autores, esses métodos, por si só, já se mostravam efetivos para aumento da acurácia, realizaram uma comparação entre diversos outros com o intuito de comparar a eficiência das diferentes técnicas. Dentre as técnicas tradicionais utilizadas, destacam-se: rotação, ampliação e distorção. Além disso, para efeito de comparação, também foram utilizadas Redes Adversárias Generativas (GANs), do inglês *Generative Adversarial Networks*. Os experimentos foram realizados sobre três subconjuntos retirados das bases de dados tiny-imagenet-200 [46] e MNIST [19]. Um dos subconjuntos, por exemplo, possui 500 imagens de gatos e 500 imagens de cachorros, sendo 400 imagens de cada classe usadas para treinamento e 100 de cada para validação. Cada técnica foi testada durante dez rodadas e a rede, proposta pelos autores com estrutura baseada no modelo VGG16, foi treinada por 40 épocas por rodada com taxa de aprendizado de 0,0001. A acurácia foi utilizada como métrica principal de avaliação. No primeiro subconjunto de imagens, cachorros e gatos, os métodos tradicionais de *data augmentation* apresentaram uma acurácia superior aos modelos treinados sem *data augmentation* e com a utilização de GANs. A utilização

de *data augmentation* foi capaz de proporcionar um aumento acima de 5% na acurácia geral dos modelos.

Gu et al. [47] apresentam diversas técnicas para otimização de CNNs relacionadas ao problema de classificação. Os experimentos deste trabalho são realizados sobre a base de dados CIFAR-10 [48] e dentre outras análises, buscam comparar o desempenho de CNNs com e sem a aplicação de técnicas tradicionais de *data augmentation*, dentre elas: rotação, modificação na altura e largura e ampliação. Embora o artigo trate de outras formas de modificação do modelo de CNN para um melhor desempenho, quando os demais parâmetros são mantidos constantes e variando apenas o uso das técnicas de *data augmentation*, a acurácia dos modelos com a utilização é maior em pelo menos dois por cento.

Os resultados apresentados anteriormente são baseados em bases de dados de uso geral, ou seja, que contêm imagens dos mais variados temas, portanto, para o propósito do presente trabalho, vale a pena examinar os resultados de um estudo com uma base de dados similar àquela a ser utilizada aqui, imagens dermatoscópicas. É nesse contexto que Pham et al. [11] analisam o impacto de técnicas tradicionais de *data augmentation* em redes profundas no aumento de acurácia e superação da escassez de dados. Diferentes estratégias são utilizadas, como por exemplo modificações geométricas e alterações na coloração das imagens. Os experimentos propostos pelo autor foram feitos sobre a base de dados ISIC Challenge 2017 [38] que contém poucas imagens para treinamento, 6.162, e apresenta classes desbalanceadas, 18,08% de imagem com lesões malignas e 81,92% de lesões benignas. As métricas utilizadas foram AUC, acurácia média, sensividade, especificidade e porcentagem de predições positivas. A acurácia média apresentou o estado da arte para 2018 com valor de 89%, demonstrando mais uma vez a eficiência da técnica de *data augmentation*.

Como visto nos trabalhos anteriores, a aplicação de *data augmentation* proporciona um aumento considerável na acurácia de modelos baseados em redes profundas. Vale a pena investigar o impacto que o tamanho das bases de dados e o balanceamento das classes possui na variação da eficácia com a aplicação da técnica. Esses são pontos abordados no estudo realizado por Hu et al. [49]. Os autores buscam correlacionar essas variáveis e gerar diretrizes na escolha e utilização das técnicas de *data augmentation*. Os experimentos propostos pelo artigo são baseados nas seguintes técnicas tradicionais de *data augmentation*: translação, escalonamento, rotação, recorte, inversão, solarização, balanceamento de cor e contraste. As bases de dados utilizadas são a CIFAR-10 [48] e a MNIST [19]. Subconjuntos de 50, 100, 200, 300, 400, 500, 2000, 3000 e 4000 imagens são escolhidos, aleatoriamente, das bases de dados e sobre eles são aplicados as técnicas citadas anteriormente. Os autores chegaram em resultados que foram considerados no

presente trabalho, são eles: 1. A taxa ótima de aumento, entre classes, é entre duas e três vezes a quantidade de amostras da classe menos populosa; 2. métodos simplificados como translação, rotação e escalonamento tendem a proporcionar melhores resultados do que métodos mais complexos;

Outro aspecto a ser considerado é quais técnicas de *data augmentation* apresentam os melhores resultados em bases de dados utilizadas em CNNs. Taylor e Nitschke [50] em seu artigo, propõem um *benchmark* das técnicas mais populares de *data augmentation* e indicam as técnicas que apresentam o melhor desempenho. Os experimentos foram realizados sobre a base de dados Caltech101 [51], que contém 9.144 imagens em 102 categorias distintas, e fazem uma comparação entre dois tipos de técnicas diferentes em *data augmentation*: técnicas usando geométria, que consistem em modificações diretas na geometria dos *pixels* da imagem, e técnicas de fotometria, que modificam os valores *RGB* (*Red*, *Green* e *Blue*) de cada *pixel* da imagem, mapeando-os para novos valores. Os autores concluíram que para a base de dados utilizada, técnicas geométricas possuem melhor desempenho do que técnicas baseadas em fotometria e que, especificamente dentro do grupo de métodos geométricos, a técnica de *cropping*, que consiste em remover regiões periféricas de menor importância da imagem foi a que apresentou os melhores resultados.

3.2 Transfer Learning

Acosta et al. [10] apresentam uma metodologia baseada em CNNs com a utilização de *transfer learning* para detecção de lesões de pele malignas em imagens dermatoscópicas. Os experimentos tinham como objetivo obter não apenas uma acurácia alta, mas também uma acurácia balanceada alta para demonstrar a capacidade do modelo de identificar corretamente a existência ou não de lesões de pele malignas. Foi realizado um pré-treinamento da rede utilizando a base de dados ImagetNet [31], seguida pela utilização do *transfer learning* sobre a base de dados desbalanceada ISIC Challenge 2017 [38]. Cinco modelos, com diferentes conjuntos de hiper parâmetros, foram gerados após o treinamento. Para o treinamento do primeiro modelo, utilizou-se a base de dados sem aplicar nenhuma técnica. Para os outros quatro, aplicou-se *data augmentation* na tentativa de diminuir o desbalanceamento das classes na base ISIC. As métricas utilizadas para avaliação dos resultados foram: especificidade, sensibilidade, acurácia e acurácia balanceada. Os resultados obtidos demonstraram que no melhor modelo aplicou-se *data augmentation* na classe maligna e não modificou a classe benigna que, após mais uma etapa de treinamento, apresentou os seguintes resultados: 1. Sensividade 82,0%; 2. Especificidade 92,5%; 3. Acurácia 90,4%; 4. Acurácia balanceada 87,2%.

Menegola et al. [52] verificaram o impacto da utilização da técnica de *transfer learning* em Redes Neurais Profundas (DNNs) para detecção de lesões de pele e, então, quantificaram as variações nos resultados proporcionados por essa técnica. Os pesquisadores utilizaram as seguintes bases de dados: Interactive Atlas of Dermoscopy [53] e ISBI Challenge 2016 [54]. Os modelos são pré-treinados utilizando a base de dados ImageNet seguidos pela aplicação do *transfer learning*. *Data augmentation* também é utilizada nas bases de dados para diminuir o desbalanceamento entre as classes. As técnicas de *data augmentation* utilizadas não foram detalhadas. Os autores chegaram a conclusão em seus experimentos que aplicação de *transfer learning* proporciona melhores resultados, o que corresponde ao indicado na literatura. Assim, obtiveram um aumento no valor de AUC de 76% para 82,5% com a aplicação da técnica de *transfer learning* na classificação entre imagens benignas e malignas. Os resultados obtidos por meio do pré-treinamento dos modelos com uma base de dados com imagens mais generalistas, como a ImageNet, gerou melhores resultados do que com bases de dados focadas em um tema específico, no caso, imagens de retinopatia diabética.

Mahbod et al. [55] analisaram o efeito da variação da resolução de imagens em CNNs pré-treinadas usando *transfer learning*. As redes foram pré-treinadas com imagens da base de dados ImageNet [31]. As arquiteturas utilizadas foram a DenseNet-121, ResNet-18 e ResNet-50. Para essas redes, imagens de tamanho 64×64 até 768×768 retiradas das bases de dados ISIC 2016 challenge, [54] e ISIC 2017 challenge [38] foram geradas por meio da aplicação de um algoritmo denominado *grayworld colour constancy* [56] para normalizar a cor das imagens. Além disso, foi realizada uma subtração da intensidade média dos valores *RGB* das imagens pertencentes à base de dados ImageNet para cada *channel* em todas as imagens dos conjuntos de de treino e teste, e, por fim, as imagens foram redimensionadas em cinco resoluções diferentes por meio da utilização de interpolação bicúbica. Os resultados encontrados foram que a técnica de *transfer learning* é impactada negativamente com menores resoluções de imagens. Os autores concluíram portanto, que imagens com maior resolução levam à um melhor desempenho quando aplicadas em modelos pré-treinados utilizando *transfer learning*.

Hosny et al. [12] apresentam um trabalho focado em apenas uma arquitetura CNN, a AlexNet. Sua metodologia teve como objetivo principal aperfeiçoar os pesos da AlexNet por meio das técnicas de *transfer learning* e *fine-tuning*. Além disso, a técnica de *data augmentation* também foi utilizada por meio da técnica de rotação das amostras com o objetivo de obter uma maior quantidade de imagens. A técnica *data augmentation* foi aplicada nos seguintes datasets: ISIC 2017 challenge [38], MED-NODE [57] e DermQuest [58]. Os autores obtiveram uma acurácia balanceada de mais de 95% para todas as bases de dados utilizadas.

Zhang [59] tenta otimizar o desempenho de uma arquitetura customizada baseada na EfficientNet [60] na detecção de melanoma. O autor utiliza a técnica de *transfer learning* para otimizar os resultados. A rede foi pré-treinada utilizando a base de dados IMAGENET e a técnica de *fine-tuning* é aplicada em seguida com a base SIIM-ISIC 2020 [61]. A base SIIM-ISIC 2020 é divida em três conjuntos, treinamento, validação e teste, seguindo a proporção 7:2:1. A métrica de avaliação escolhida foi a AUC e, para ela, o autor obteve um valor de 0,917.

Por fim, considerando as informações discutidas nesse capítulo, é possível considerar que a técnica de *transfer learning* proporcionou um aumento de desempenho na maioria dos trabalhos analisados quando o pré-treinamento foi realizado com a base de dados ImageNet. Portanto, para a etapa de pré-treinamento no presente trabalho, a mesma base de dados será utilizada de forma a verificar se a técnica de *transfer learning* é capaz de promover o aumento de desempenho nas CNNs analisadas para a tarefa de detecção de melanoma. Já em relação a técnica de *data augmentation*, os melhores resultados foram provenientes da utilização de técnicas tradicionais de ampliação das amostras como por exemplo: rotação, ampliação e espelhamento. Portanto, uma combinação específica de técnicas tradicionais de *data augmentation* será utilizada. Além disso, como o contexto principal a ser analisado é a detecção de melanoma e as bases de dados com esse tipo de imagens médicas normalmente possuem como característica o desbalanceamento das classes em desfavor da classe positiva, a técnica de *data augmentation* será utilizada, preferencialmente, sobre a classe maligna. Considerando que as duas técnicas analisadas proporcionaram um aumento no desempenho dos trabalhos discutidos aqui, é interessante verificar se esse aumento de desempenho é maior quando as técnicas são utilizadas em conjunto do que quando são utilizadas individualmente.

Capítulo 4

Metodologia

O objetivo principal do presente trabalho é verificar o impacto das técnicas de *data augmentation* e *transfer learning* no desempenho das CNNs escolhidas, para isso, uma metodologia específica foi proposta de forma a isolar a influência de cada uma dessas técnicas nos resultados e então determinar a eficácia de sua aplicação. O fluxograma da Figura 4.1 apresenta as etapas seguidas durante a realização dos experimentos. As demais seções descrevem as bases de dados utilizadas com suas respectivas modificações, a metodologia de treinamento, e, ao fim, os procedimentos realizados em cada um dos cinco experimentos propostos.

4.1 Bases de Dados

Duas diferentes bases de dados são utilizadas nos experimentos, uma para realizar a técnica de *transfer learning* e outra para o treinamento da rede sobre o contexto de aplicação escolhido, detecção de melanoma.

4.1.1 Distribuição dos Dados

A base de dados ImageNet [25] foi a base utilizada para o processo de *transfer learning*, mais precisamente o subconjunto utilizado no Desafio de Reconhecimento da ImageNet (ILSVRC). Esse subconjunto possui mais de 1 milhão de imagens distribuídas em 1000 classes. Como o tamanho dessa base de dados é consideravelmente grande, espera-se que a sua utilização no processo de pré-treinamento por meio da técnica de *transfer learning* proporcione um aumento na acurácia geral das arquiteturas. O objetivo principal da técnica de *transfer learning* no presente trabalho é possibilitar às arquiteturas aprender a realizar o processo de extração de características das imagens com mais facilidade, como por exemplo a detecção de formas geométricas. Como esse processo de aprendizado é

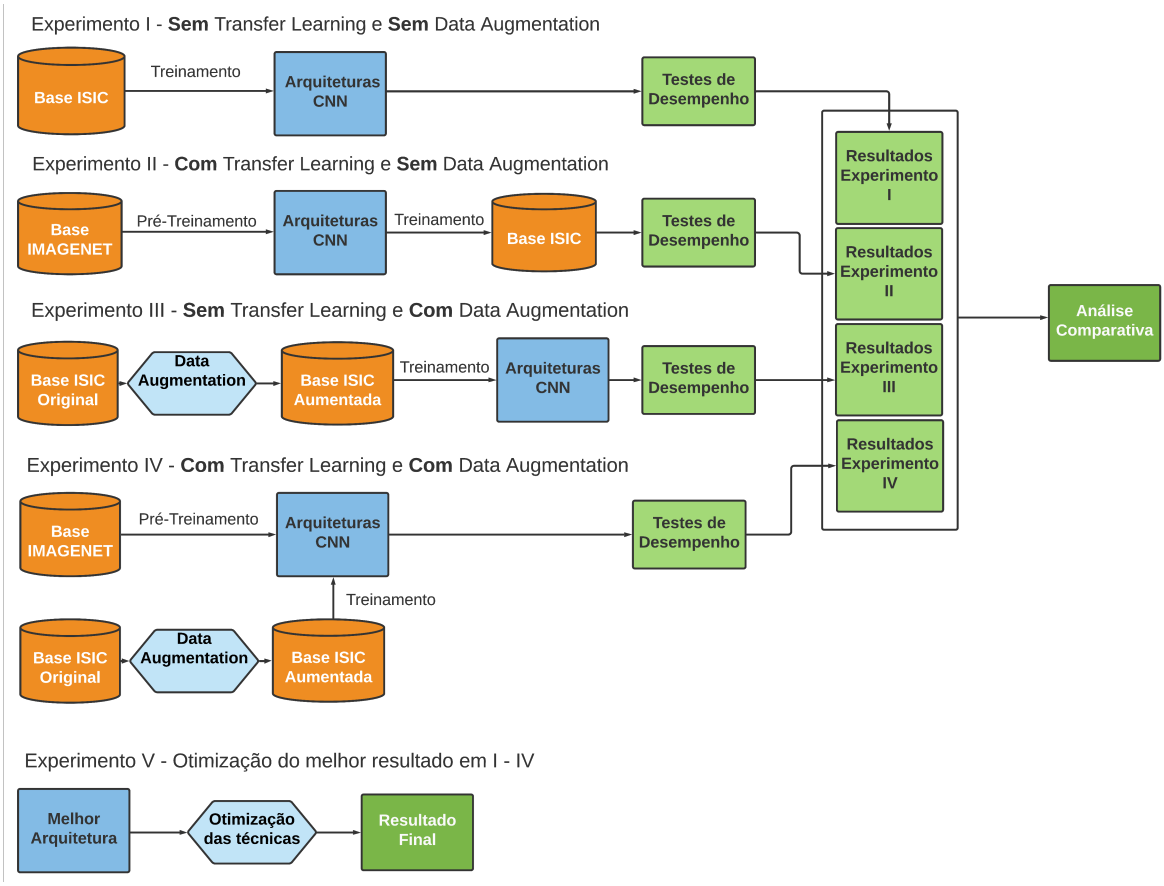

Figura 4.1: Fluxograma da metodologia utilizada.

realizado sobre uma base de dados com uma quantidade grande de amostras, espera-se que após a aplicação da técnica, quando as arquiteturas forem utilizadas para o treinamento com a base de dados com imagens de melanoma, seja possível perceber um aumento considerável na acurácia dos modelos. A Tabela 4.1 apresenta a distribuição da quantidade de amostras subdivididas por etapas de treinamento.

Tabela 4.1: Distribuição original das imagens por etapa - ILSVRC

Etapas	**Número de Amostras**	**Proporção**
Treinamento	1.281.167	92,76%
Validação	50.000	3,62%
Teste	100.000	7,24%
Total	1.381.167	100%

Essa base de dados não possui imagens dermatoscópicas de lesões de pele, entretanto, o objetivo com a utilização da transferência de aprendizado é possibilitar que as CNNs aprendam a realizar o processo de extração de características das imagens fornecidas, independentemente do contexto das classes.

A base de dados escolhida para treinar a rede na detecção de melanoma especificamente, foi a base compilada em SIIM-ISIC 2020 [61] que contém imagens dermatoscópicas subdividas em 2 classes: imagens com lesões de pele benignas e imagens com lesões de pele malignas. As imagens que apresentam lesões malignas são imagens com lesões do tipo melanoma. A base é formada por um subconjunto de treinamento e um subconjunto de testes particionados de acordo com a Tabela 4.2. Essa base foi escolhida por dois motivos principais: o diagnóstico das amostras malignas foi confirmado por meio de exames médicos, o que da credibilidade para a formulação da base de dados e a composição de suas amostras é dividida de forma binária, uma classe com amostras benignas e outra com amostras malignas, ou seja, ideal para o treinamento de um classificador binário que é o foco do presente trabalho.

Tabela 4.2: Distribuição original das imagens por etapa - ISIC20

Etapas	**Número de Amostras**	**Proporção**
Treinamento	33.126	75,10%
Teste	10.982	24,90%
Total	44.108	100%

Como a base foi utilizada em uma competição em que os competidores precisavam enviar as classificações obtidas para as imagens do conjunto de testes, para essas imagens não são fornecidos os *labels* das classes, portanto, essas 10.982 imagens de testes não são utilizadas no presente trabalho. De forma a contornar esse problema, as 33.126 imagens, propostas inicialmente para serem utilizadas apenas na etapa de treinamento, foram particionadas de maneira estratificada para serem utilizadas nas etapas de treinamento, validação e teste. A proporção utilizada é de 68:12:20, ou seja, 68% das imagens serão utilizadas para treinamento, 12% para validação e 20% para testes. A Tabela 4.3 apresenta a distribuição das imagens por etapa e a proporção entre a classes.

Tabela 4.3: Distribuição original das imagens por etapa - ISIC20

	Treinamento		**Validação**		**Teste**	
	Benigna	**Maligna**	**Benigna**	**Maligna**	**Benigna**	**Maligna**
Quantidade	22.129	397	3.904	70	6.510	117
Proporção	98,24%	1,76%	98,24%	1,76%	98,23%	1,77%
Total	22.526		3.974		6.627	

Durante o Experimento V, a técnica de *data augmentation* também é utilizada mas com proporções diferentes daquelas definidas na Tabela 4.3. Durante a primeira etapa é realizada uma diminuição da quantidade de amostras da classe benigna, seguida por um balanceamento da classe maligna. A quantidade resultante de amostras pode ser vista na Tabela 4.4. Já na segunda etapa, a quantidade de amostras malignas é dobrada em

relação à quantidade utilizada no Experimento IV. A quantidade de amostras para essa etapa pode ser vista na Tabela 4.5.

Tabela 4.4: Distribuição de amostras para o Experimento V - Etapa I

	Treinamento		**Validação**		**Teste**	
	Benigna	**Maligna**	**Benigna**	**Maligna**	**Benigna**	**Maligna**
Quantidade	11.064	11.064	3.904	70	6.510	117
Proporção	50%	50%	98,24%	1,76%	98,23%	1,77%
Total	22.128		3.974		6.627	

Tabela 4.5: Distribuição de amostras para o Experimento V - Etapa II

	Treinamento		**Validação**		**Teste**	
	Benigna	**Maligna**	**Benigna**	**Maligna**	**Benigna**	**Maligna**
Quantidade	22.129	44.258	3.904	70	6.510	117
Proporção	33,33%	66,66%	98,24%	1,76%	98,23%	1,77%
Total	66.387		3.974		6.627	

4.1.2 Ampliação dos Dados

É importante destacar o desbalanceamento das classes como mostrado na Tabela 4.3. Para cada amostra de lesão maligna existem aproximadamente 56 amostras de lesões benignas. A técnica de *data augmentation*, portanto, é utilizada de forma a minimizar os impactos desse desbalanceamento na capacidade dedutiva das arquiteturas CNNs. Considerando que o conjunto de imagens da classe maligna reservadas para a etapa de treinamento tem um tamanho de 397, um número muito baixo, é possível que a utilização dessa quantidade pequena de amostas sem a utilização da técnica de *data augmentation* prejudique a capacidade de classificação das CNNs para a classe positiva. Portanto, para os experimentos III e IV, durante a etapa de treinamento, será aplicada sobre a classe minoritária da base de dados uma taxa de ampliação de 56 vezes, ou seja, cada imagem de lesão maligna dará origem a novas 56 imagens, após o procedimento as classes ficam balanceadas. A Tabela 4.3 exibe a quantidade de amostras e as distribuições por etapa antes da aplicação da técnica. A quantidade de imagens separadas nos subconjuntos de teste e validação descrita na Tabela 4.3 permanece inalterada, ou seja, apenas as imagens do subconjunto de treinamento serão ampliadas.

Para realizar a ampliação dos dados, cinco técnicas tradicionais [45] serão utilizadas: alteração no brilho, rotação, espelhamento, ampliação e distorção. Todo o processo de ampliação de dados foi realizado através da ferramenta *Augmentor* [62]. A aplicação de cada técnica em uma amostra é condicionada a uma probabilidade pré-definida e é

limitada por um valor mínimo e um valor máximo. A Tabela 4.6 sumariza os procedimentos que serão utilizados com suas respectivas probabilidades de aplicação e intervalo de modificação, para cada técnica de ampliação como descrita a seguir:

Tabela 4.6: Técnicas de ampliação utilizadas

Técnicas de Ampliação	Valor Mínimo	Valor Máximo	Probabilidade
Alteração no Brilho	0,7	1,5	100%
Espelhamento	Aleatório	Aleatório	95%
Rotação	5º Esquerda	5º Direita	90%
Ampliação	1,1	1,5	80%
Distorção	Aleatório	Aleatório	80%

- Alteração no Brilho: cada amostra, com probabilidade de 100%, tem sua intensidade de brilho alterada. A nova amostra possui um brilho escolhido aleatoriamente com valor entre 0,7 e 1,5, sendo que 1 representa o brilho original. Esse intervalo foi definido pois foi observado durante os experimentos que imagens com brilho abaixo de 0,7 ficam muito escuras de forma que a visualização da lesão fica prejudicada, o mesmo ocorre para imagens com brilho maior ao limite superior estabelecido;

- Espelhamento: com probabilidade de 95%, cada amostra é espelhada verticalmente ou horizontalmente. A aplicação do espelhamento com a ferramenta utilizada, não gerou problemas de renderização, portanto, a aplicação da técnica é mais frequente;

- Rotação: cada amostra, com probabilidade de 90%, é rotacionada em um ângulo dentro do intervalo de 5 graus pra esquerda e 5 graus para a direita escolhidos aleatoriamente. Rotações com valores acima de 5 graus para qualquer lado geraram resultados inconsistentes devido as dimensões da imagem, portanto, usando uma margem de segurança, a quantidade de 5 graus é definida;

- Ampliação: com probabilidade de aplicação de 80%, a amostra é ampliada em um intervalo de 1,1 vezes até 1,5 vezes, sendo que, 1,5 equivale a uma ampliação de 50%. Ampliações com valores superiores a 50% podem, para algumas imagens, fazer com que a a lesão fique completamente fora do enquadramento da imagem, por isso esse limite foi definido. Mesmo com uma ampliação pequena 1,1, algumas vezes, algumas imagens têm suas bordas cortadas, por isso a probabilidade de aplicação dessa técnica foi reduzida;

- Distorção: a adição de ruído na imagem é realizada com probabilidade de 80%. Esse método apresenta a menor probabilidade de todos os métodos utilizados pois, para algumas imagens, a adição de ruído prejudica a qualidade visual da amostra.

A probabilidade de aplicação das técnicas foi escolhida de forma a proporcionar uma maior variabilidade das amostras. Além disso, algumas técnicas são processadas mais rapidamente do que outras como por exemplo a alteração do brilho da imagem, portanto essa técnica será aplicada em todas as amostras. A Figura 4.2 contém um exemplo de aplicação das técnicas descritas para duas imagens pertencentes à base ISIC20 [61]. A partir de cada imagem à esquerda, cinco novas imagens são criadas utilizando as técnicas descritas na Tabela 4.6, sendo assim, para essas novas amostras, as técnicas utilizadas são escolhidas de forma aleatória.

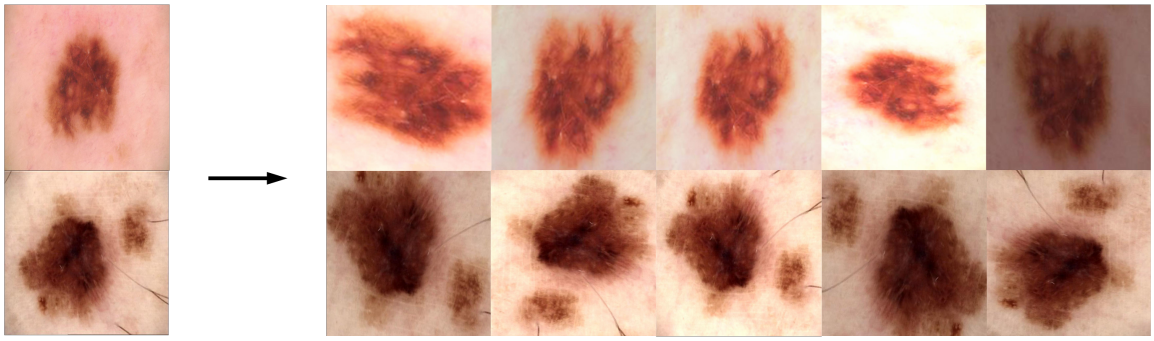

Figura 4.2: Exemplo de ampliação das imagens. Originais à Esquerda.

4.2 Treinamento da CNN

O processo de treinamento utiliza uma metodologia de parada customizada, a rede continuará a ser treinada até que o modelo não aprenda nenhuma informação relevante durante cinco épocas ou até atingir a décima quinta época. Esse limitação foi proposta para o presente trabalho de forma a limitar o tempo máximo de treinamento. O critério para determinar se a rede aprendeu na última época é o valor da perda sobre o subconjunto de validação, caso o valor da perda da época atual seja inferior ao menor valor obtido até o momento, considera-se que a rede aprendeu. A melhor época, e portanto a escolhida como melhor resultado do treinamento, é aquela que apresenta a menor perda durante toda a fase de treinamento.

Os hiper parâmetros utilizados no treinamento são mantidos constantes no transcorrer dos experimentos para todas as arquiteturas utilizadas de forma a padronizar e minimizar a variabilidade dos resultados quando a comparação entre as arquiteturas for realizada. Como otimizador, é utilizado o método Adam como descrito na Seção 2.4.1 e como critério de avaliação da perda, a metodologia de entropia cruzada de acordo com a Seção 2.4.2.

O treinamento das arquiteturas foi realizado em um computador com processador Ryzen 5 3600x, placa de vídeo AMD Radeon 5700 XT, 32GB de memória RAM e SSD SPECTRIX 512GB. Os procedimentos foram realizados no sistema operacional Windows 10.

4.3 Etapa Experimental

Essa seção detalha os cinco experimentos que serão realizados nesse trabalho e os resultados esperados de cada um deles. O processo de treinamento para os Experimentos I, II, III e IV será realizado três vezes, uma vez para cada CNN, AlexNet [26], GoogLeNet [29] e ResNet [32], e durante o experimento V, apenas a arquitetura que obteve os melhores resultados será utilizada. O treinamento das CNNs seguirá os procedimentos descritos na Seção 4.2.

4.3.1 Experimento I

Esse experimento tem como objetivo estabelecer o desempenho inicial das arquiteturas, servirá de base de comparação para os demais experimentos. Aqui, a CNN é treinada e testada sem a utilização das técnicas de *data augmentation* e *transfer learning* como pode ser visto na Figura 4.3. Os resultados obtidos nessa etapa serão o reflexo direto do desempenho de cada uma das arquiteturas mediante a utilização da base de dados SIIM-ISIC 2020 sem nenhuma modificação adicional. A quantidade e a proporção de amostras utilizadas para treinamento nessa etapa estão definidas na Tabela 4.3.

Espera-se que esse experimento apresente os piores resultados em comparação com os demais uma vez que aqui não é feito o pré-treinamento e a quantidade de imagens da base de aplicação possui muito poucas imagens de forma a dificultar a capacidade do modelo de extrair características das imagens de forma eficaz.

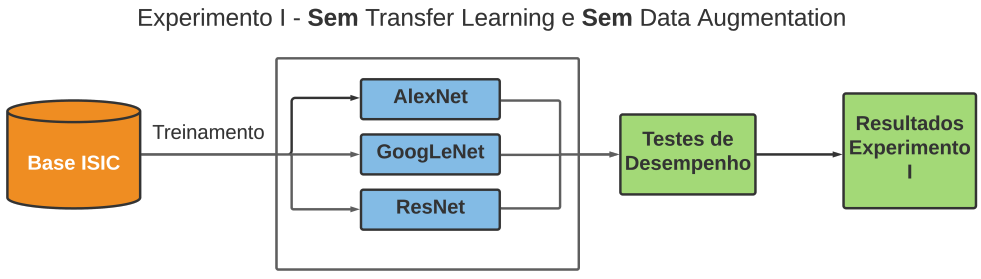

Figura 4.3: Fluxograma Do Experimento I.

4.3.2 Experimento II

De posse da performance base obtida no Experimento I, é possível mensurar a eficiência da técnica de *transfer learning* nesse experimento, dado que a base de dados permanecerá inalterada. A única diferença dessa etapa para a anterior é que aqui a técnica de *transfer learning* é aplicada. A ordem dos procedimentos pode ser vista na Figura 4.4. A Tabela 4.1 apresenta as subdivisões da base IMAGENET utilizada para pré-treinamento. Assim como no Experimento I, a Tabela 4.3 apresenta a quantidade e a proporção de amostras utilizadas durante o treinamento.

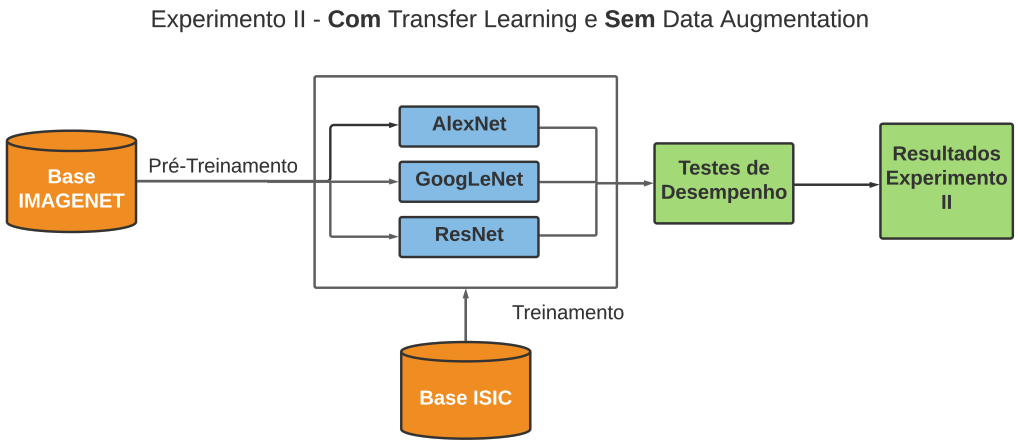

Figura 4.4: Fluxograma Do Experimento II.

4.3.3 Experimento III

Enquanto o foco do Experimento II é mensurar a eficiência da técnica de *transfer learning*, esse tem com objetivo analisar a técnica de *data augmentation*. Durante esse experimento, apenas a técnica de *data augmentation* será utilizada. Diferentemente do Experimento II em que ocorreu apenas a variação na aplicação de técnicas, aqui, a quantidade de amostras da base de dados com imagens de melanoma que foi utilizada para treinamento também muda. Uma explicação mais detalhada pode ser vista na Seção 4.1.2. A Figura 4.5 resume os procedimentos adotados.

4.3.4 Experimento IV

Esse experimento utiliza as técnicas de *data augmentation* e *transfer learning* simultaneamente como exibido na Figura 4.6. Nessa etapa, os procedimentos descritos na Seção 4.1.2 são aplicados. O principal objetivo aqui é mensurar a variação do desempenho das

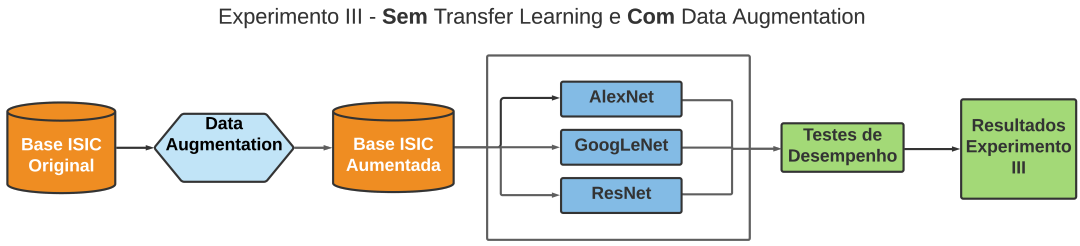

Figura 4.5: Fluxograma Do Experimento III.

arquiteturas com a aplicação simultânea das técnicas de *data augmentation* e *transfer learning*. Espera-se que esse experimento apresente melhores resultados em comparação aos experimentos anteriores, uma vez que, aqui, ao treinar a rede com a base de imagens de lesões de pele, a rede já tenha aprendido a extrair características das imagens na etapa de pré-treinamento e, além disso, a quantidade de imagens de cada uma das classes estará balanceada.

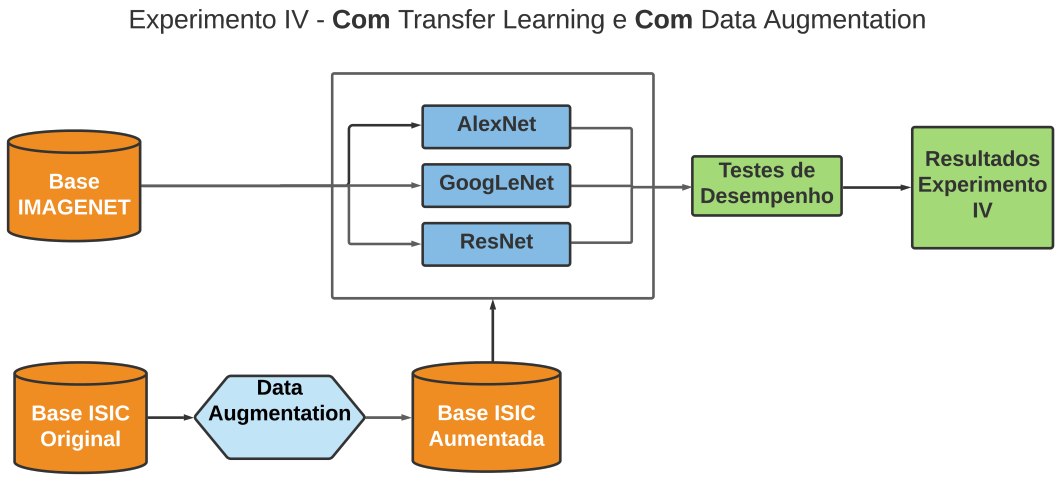

Figura 4.6: Fluxograma Do Experimento IV.

4.3.5 Experimento V

De posse dos resultados obtidos nos experimentos anteriores, é possível determinar qual arquitetura e quais combinações de técnicas apresentaram os melhores resultados, especificamente qual modelo apresentou a maior acurácia balanceada. Nessa etapa, seguindo os passos descritos na Figura 4.7 e utilizando a arquitetura com melhor desempenho, serão utilizadas outras duas proporções na quantidade de imagens de cada classe para verificar a variação na acurácia balanceada e então gerar um resultado experimental final. Segundo

Acosta et al. [10], promover um balanceamento perfeito das classes não gera, necessariamente, uma melhora no desempenho do modelo. Sendo assim, na tentativa de aumentar o desempenho do melhor modelo, duas etapas serão realizadas. A primeira consiste em realizar uma diminuição da classe benigna em 50% seguida pela aplicação da técnica de *data augmentation* sobre a classe maligna até que a quantidade de amostras de cada classe seja igualada. O objetivo é verificar se um possível enviesamento para a classe negativa será diminuído uma vez que a variabilidade para essa classe também diminuirá com a redução da quantidade de amostras. A segunda etapa consiste em aumentar a quantidade de imagens da classe maligna em 100%, considerando a quantidade de imagens para cada classe utilizada no Experimento IV como inicial. O modelo tomado como melhor será aquele que apresentar a maior acurácia balanceada dentre os cinco experimentos propostos.

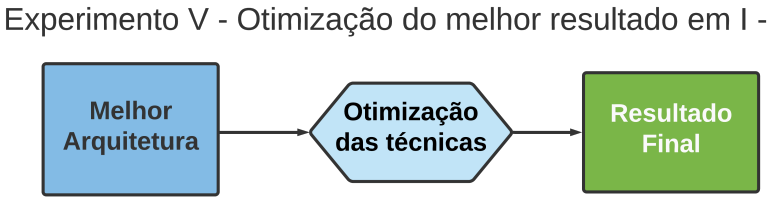

Figura 4.7: Fluxograma Do Experimento V.

Capítulo 5

Resultados Experimentais

Este capítulo apresenta os resultados obtidos em cada um dos cinco experimentos propostos na Seção 4.3 e ao fim, realiza uma comparação dos resultados obtidos em cada uma das etapas.

5.1 Experimento I

Como descrito na Seção 4.3.1, esse experimento tem como objetivo principal mensurar o desempenho das arquiteturas sem a aplicação das técnicas de *transfer learning* e *data augmentation*. A Tabela 5.1 resume os resultados obtidos nessa etapa experimental para o conjunto de teste, e, em seguida, são apresentados os resultados para as três arquiteturas utilizadas: AlexNet, GoogLeNet e ResNet.

Tabela 5.1: Resumo dos Resultados - Experimento I (Sem DA, Sem TL).

Métricas de Avaliação	**AlexNet**	**GoogLeNet**	**ResNet**
Acurácia Balanceada	0,500	0,500	0,500
Sensibilidade	0,000	0,000	0,000
Especificidade	1,000	1,000	1,000
AUC	0,767	0,800	0,798
Acurácia	0,982	0,982	0,982

A arquitetura AlexNet foi treinada por 13 épocas. A melhor época foi a de número 8 com perda no conjunto de validação de 0,080 como pode ser visto na Figura 5.1. Todas as amostras do conjunto de teste foram classificadas como benigna. Importante destacar que para esse modelo a acurácia foi de 0,982. É essencial, portanto, observar não só a acurácia mas também a acurácia balanceada que leva em consideração a sensibilidade e a especificidade para representar o real desempenho do modelo. Embora a acurácia apresente um valor alto, a sensibilidade foi igual a zero, ou seja, a arquitetura não classifica

corretamente imagens da classe positiva. Essa arquitetura apresentou um tempo de processamento de 0,04 segundos por imagem no conjunto de teste. A Figura 5.2 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

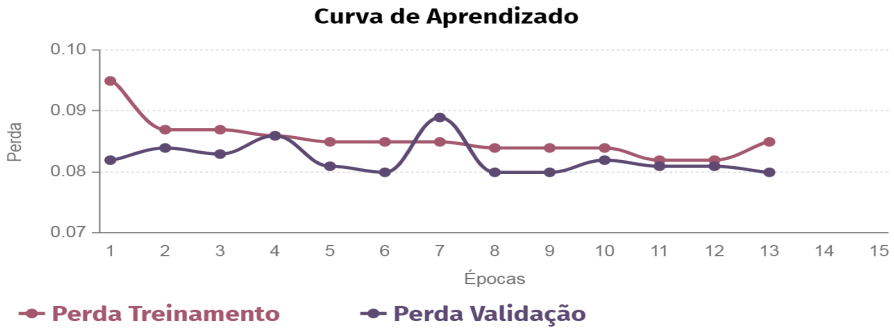

Figura 5.1: Curva de Aprendizado - AlexNet - Experimento I.

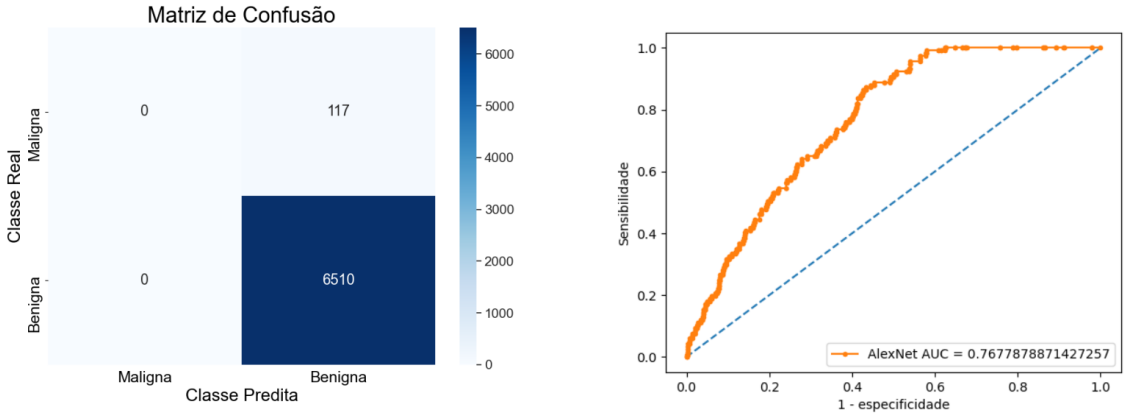

Figura 5.2: Matriz de Confusão e ROC - AlexNet - Experimento I.

A GoogLeNet foi treinada por 13 épocas. A melhor época foi a de número 8 com perda no conjunto de validação de 0,075 como pode ser visto na Figura 5.3. Esse modelo apresentou resultados similares aos obtidos na AlexNet, também classificou todas as amostras do conjunto de teste como benigna. O tempo de processamento foi de 0,08 segundos por imagem no conjunto de teste, o dobro do tempo da arquitetura anterior. A Figura 5.4 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

A ResNet foi treinada por 10 épocas. A melhor época foi a de número 5 com perda no conjunto de validação de 0,078 como pode ser visto na Figura 5.5. Assim como as outras

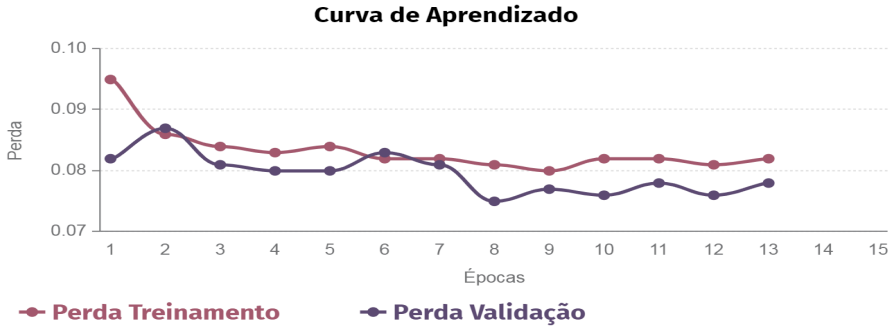

Figura 5.3: Curva de Aprendizado - GoogLeNet - Experimento I.

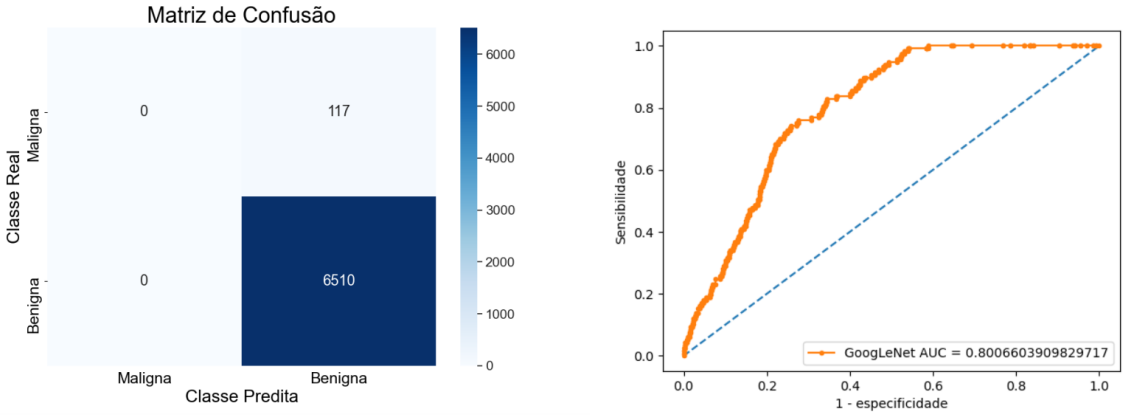

Figura 5.4: Matriz de Confusão e ROC - GoogLeNet - Experimento I.

duas arquiteturas, a ResNet também classificou todas as amostras do conjunto de teste como benigna. O tempo de processamento foi de 0,04 segundos por imagem no conjunto de teste. A Figura 5.6 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

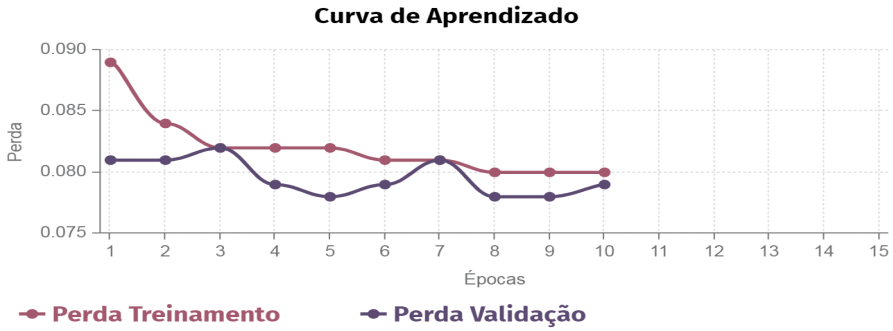

Figura 5.5: Curva de Aprendizado - ResNet - Experimento I.

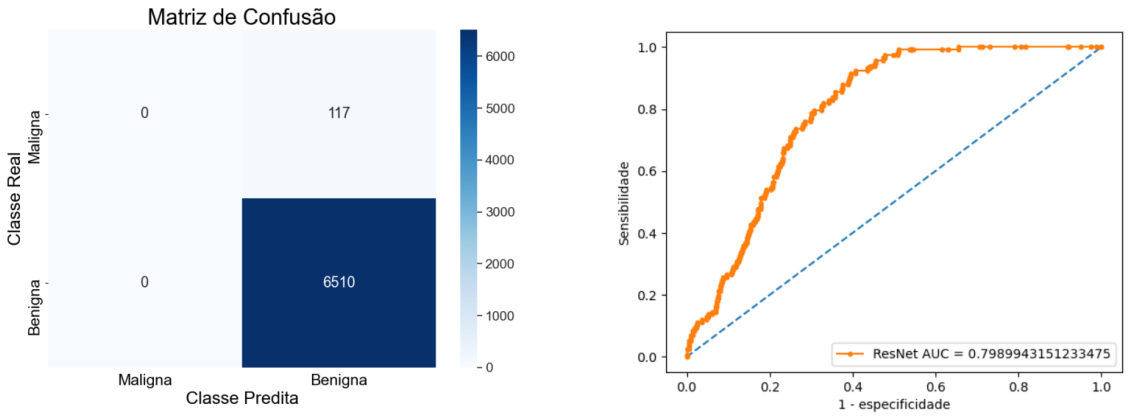

Figura 5.6: Matriz de Confusão e ROC - ResNet - Experimento I.

Como previsto, o Experimento I apresentou um resultado bem aquém do ideal, como pode ser visto na Tabela 5.1. O principal problema é que as arquiteturas classificaram todas as imagens como benignas. Uma possível justificativa é o desbalanceamento acentuado das classes como mostrado na Tabela 4.3. Como a quantidade de imagens benignas é muito superior a quantidade de imagens malignas, as redes neurais ficam enviesadas para a classe que possui a maior quantidade de amostras. Outro ponto importante é o valor relativamente alto para o AUC dos modelos. Embora as arquiteturas apresentem um desempenho ruim, a confiança das predições realizadas pela rede são altas, o que eleva o valor da AUC. Como a quantidade de imagens da classe positiva é muito baixa, os valores provenientes dessas classificações não são suficientes para promover uma queda significativa no valor da AUC, por isso a análise combinada das técnicas é importante. Importante destacar que a matriz de confusão obtida para as três arquiteturas foi a mesma, ou seja, todas as imagens foram classificadas como benignas em todas as arquiteturas, entretanto, o valor da AUC para cada arquitetura foi diferente, o que implica que embora a classificação para cada amostra seja igual, a confiança nas predições das amostra é diferente, por exemplo, duas imagens quaisquer classificadas como benignas apresentam probabilidades diferentes de pertencer a essa classe. Essas variações na precisão das classificações são responsáveis pelos diferentes valores obtidos para a AUC.

5.2 Experimento II

Seguindo a metodologia proposta na Seção 4.3.2, esse experimento analisa exclusivamente o desempenho da técnica de *transfer learning*, ou seja, a técnica de *data augmentation* não é utilizada nessa etapa. A base de dados utilizada e a proporção das classes podem ser vistas na Tabela 4.3. A Tabela 5.2 resume os resultados obtidos nessa etapa experimental

para o conjunto de teste, e, em seguida, são apresentados os resultados para as arquiteturas utilizadas.

Tabela 5.2: Resumo dos Resultados - Experimento II (Sem DA, Com TL).

Métricas de Avaliação	AlexNet	GoogLenet	ResNet
Acurácia Balanceada	0,500	0,500	0,500
Sensibilidade	0,000	0,000	0,000
Especificidade	1,000	1,000	1,000
AUC	0,767	0,829	0,841
Acurácia	0,982	0,982	0,982

A AlexNet foi treinada por 13 épocas. A melhor época foi a de número 8 com perda no conjunto de validação de 0,080 como pode ser visto na Figura 5.7. A AlexNet, com a utilização de *transfer learning*, continua classificando todas as amostras do conjunto de teste como benigna. O tempo de processamento foi de 0,04 segundos por imagem no conjunto de teste. A Figura 5.8 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

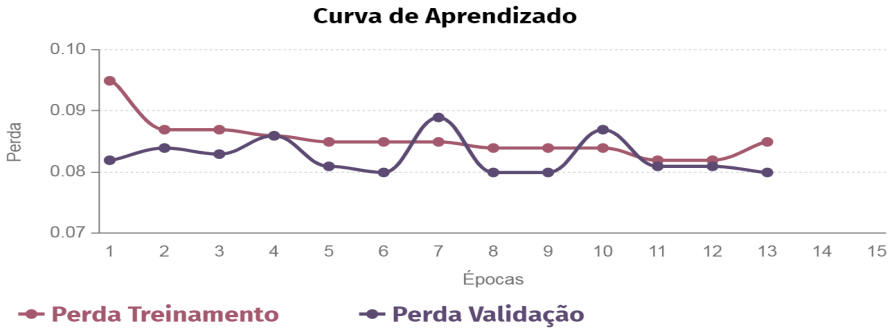

Figura 5.7: Curva de Aprendizado - AlexNet - Experimento II.

A GoogLeNet foi treinada por 15 épocas. A melhor época foi a de número 14 com perda no conjunto de validação de 0,075 como pode ser visto na Figura 5.9. Assim como a AlexNet, a GoogLeNet também continua classificando todas as amostras do conjunto de teste como benigna. O tempo de processamento foi de 0,09 segundos por imagem sobre o conjunto de teste. A Figura 5.10 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

A ResNet foi treinada por 15 épocas. A melhor época foi a de número 14 com perda no conjunto de validação de 0,075 como pode ser visto na Figura 5.11. Por fim, a ResNet também classificou todas as amostras do conjunto de teste como benigna. O tempo de processamento foi de 0,04 segundos por imagem sobre o conjunto de teste. A Figura 5.12

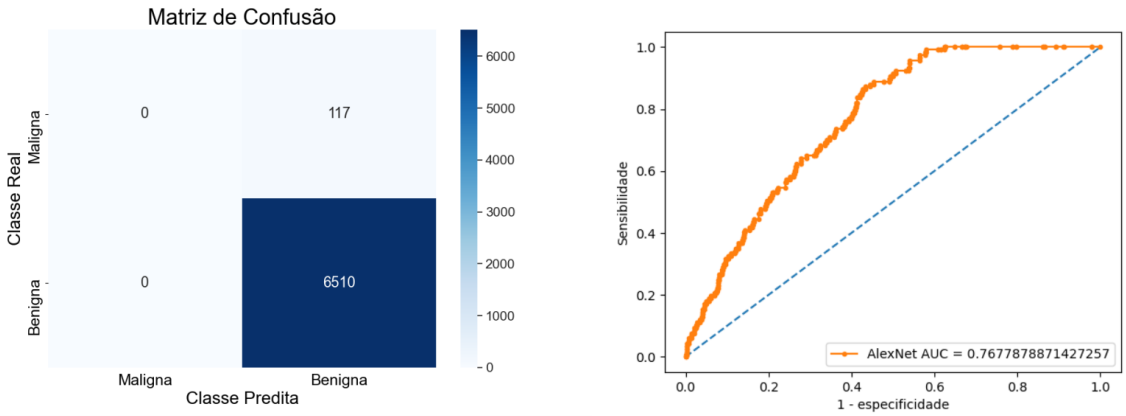

Figura 5.8: Matriz de Confusão - AlexNet - Experimento II.

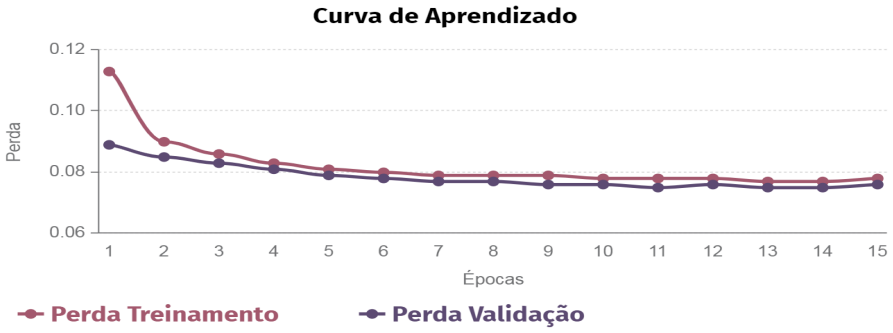

Figura 5.9: Curva de Aprendizado - GoogLeNet - Experimento II.

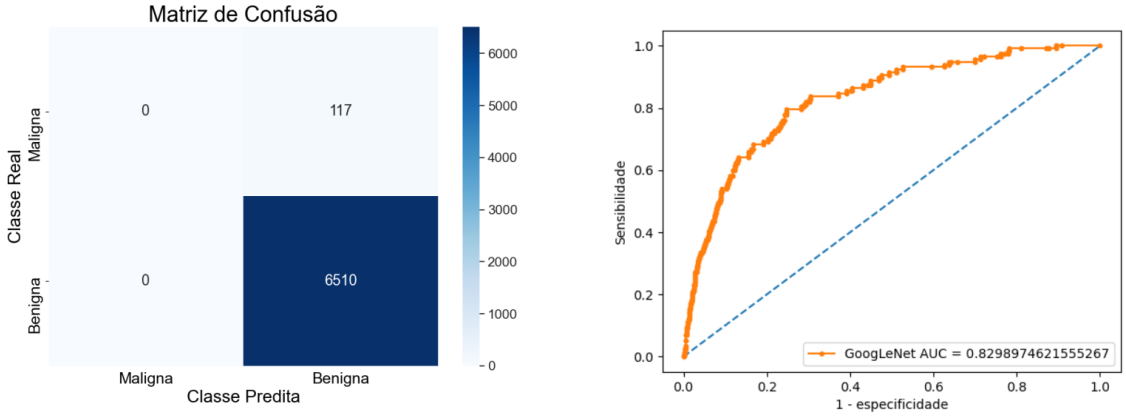

Figura 5.10: Matriz de Confusão e ROC - GoogLeNet - Experimento II .

apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

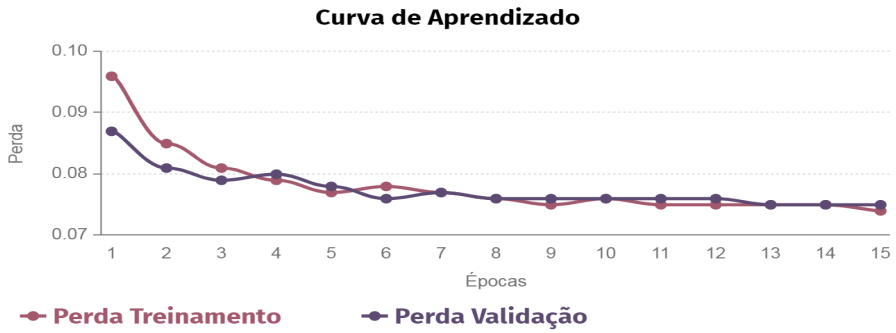

Figura 5.11: Curva de Aprendizado - ResNet - Experimento II.

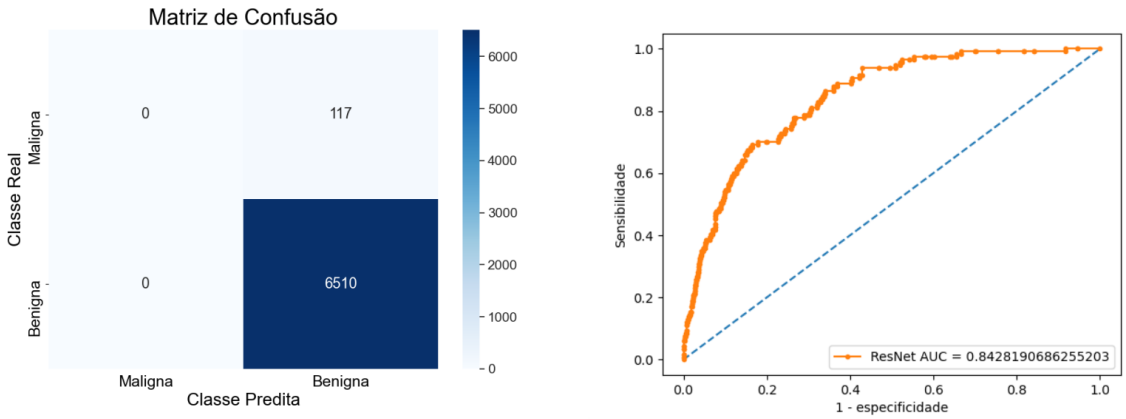

Figura 5.12: Matriz de Confusão e ROC - ResNet - Experimento II.

De forma similar ao que foi observado no experimento I, as arquiteturas classificaram todas as imagens como benignas mesmo após a aplicação da técnica de *transfer learning*. É razoável afirmar, portanto, que a técnica aplicada individualmente, não é suficiente para possibilitar que as arquiteturas superem o desbalanceamento das classes. Sendo assim, nesse experimento não foi possível observar se a técnica de *transfer learning* proporciona um aumento na acurácia geral dos modelos pois os resultados foram camuflados pelo enviesamento proporcionado pelo desbalanceamento das classes, problema que ainda não foi superado no experimento II.

5.3 Experimento III

De acordo com os procedimentos descritos na Seção 4.3.4, esse experimento analisa exclusivamente o desempenho da técnica de *data augmentation*, ou seja, a técnica de *transfer learning* não é utilizada nessa etapa. A Tabela 5.3 resume os resultados obtidos nessa

etapa experimental para o conjunto de teste, e, em seguida, são apresentados os resultados para as arquiteturas utilizadas.

Tabela 5.3: Resumo dos Resultados - Experimento III (Com DA, Sem TL).

Métricas de Avaliação	AlexNet	GoogLenet	ResNet
Acurácia Balanceada	0,628	0,661	0,653
Sensibilidade	0,333	0,402	0,385
Especificidade	0,923	0,920	0,922
AUC	0,819	0,854	0,841
Acurácia	0,912	0,911	0,912

A AlexNet foi treinada por 15 épocas. A melhor época foi a de número 13 com perda no conjunto de validação de 0,198 como pode ser visto na Figura 5.13. Pela primeira vez, a AlexNet classificou amostras como malignas. A sensibilidade ainda apresenta um resultado ruim, inferior a 0,5, ou seja, a rede ainda não fornece resultados confiáveis na detecção da classe positiva. A chance de acerto ainda se comporta pior do que uma escolha aleatória para a classe alvo. O tempo de processamento foi de 0,04 segundos por imagem sobre o conjunto de teste. A Figura 5.14 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

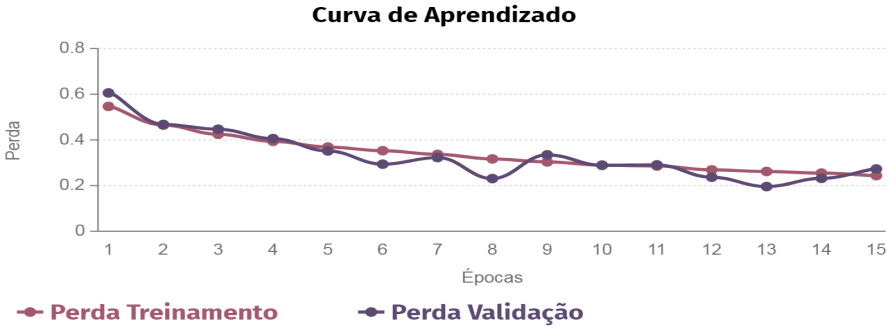

Figura 5.13: Curva de Aprendizado - AlexNet - Experimento III.

A GoogLeNet foi treinada por 15 épocas. A melhor época foi a de número 14 com perda no conjunto de validação de 0,179 como pode ser visto na Figura 5.15.

Assim como a AlexNet, a GoogLeNet começou a classificar imagens como malignas. O valor da sensibilidade também continua baixo para essa arquitetura, mas, assim como na arquitetura anterior, é possível verificar um avanço na acurácia balanceada em comparação ao Experimento I. O tempo de processamento foi de 0,08 segundos por imagem sobre o conjunto de teste. A Figura 5.16 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

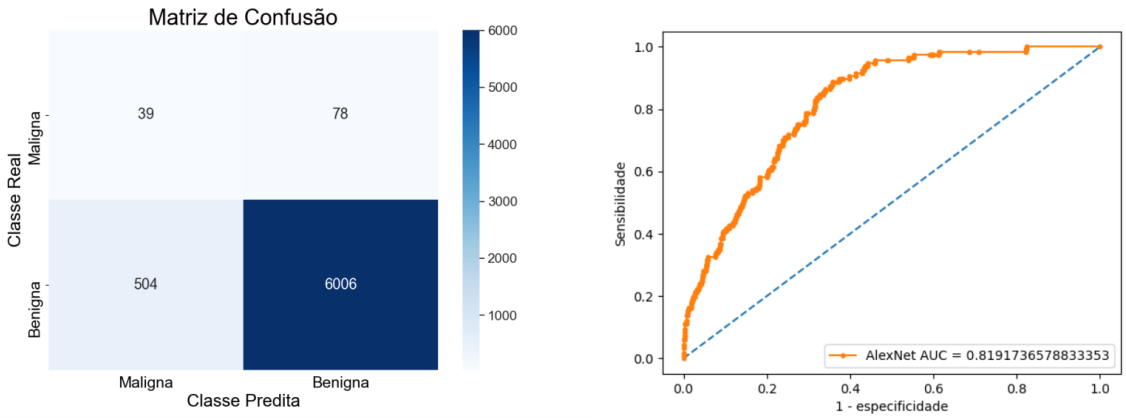

Figura 5.14: Matriz de Confusão e ROC - AlexNet - Experimento III.

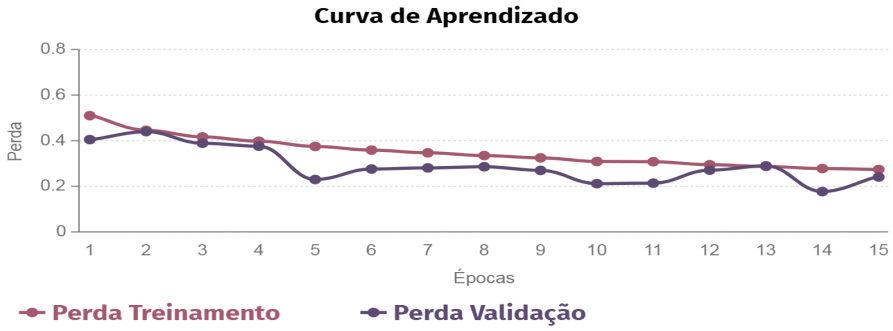

Figura 5.15: Curva de Aprendizado - GoogLeNet - Experimento III.

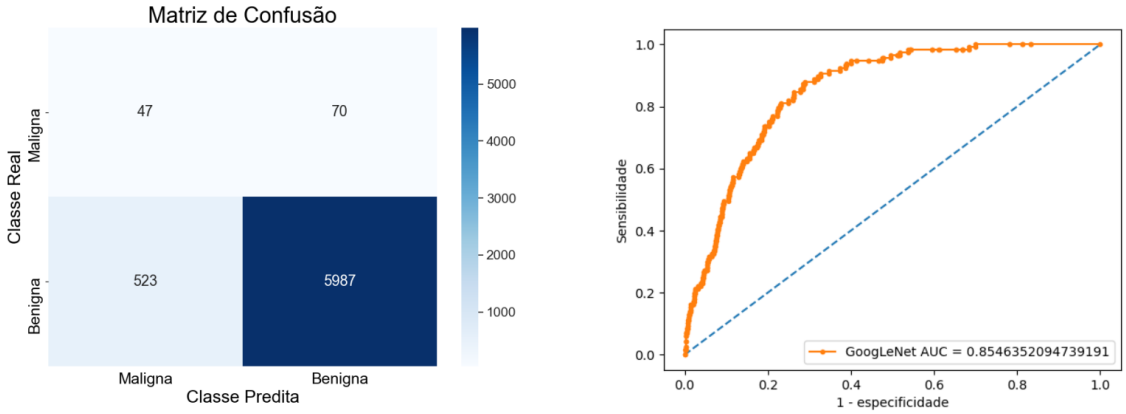

Figura 5.16: Matriz de Confusão e ROC - GoogLeNet - Experimento III.

A ResNet foi treinada por 15 épocas. A melhor época foi a de número 10 com perda no conjunto de validação de 0,183 como pode ser visto na Figura 5.17. A ResNet também

apresentou uma melhora na classificação pois classificou algumas amostras do conjunto de teste como maligna, apesar disso, sofre com o mesmo problema das outras arquiteturas do presente experimento, o valor da sensibilidade continua baixo. O tempo de processamento foi de 0,05 segundos por imagem sobre o conjunto de teste. A Figura 5.18 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

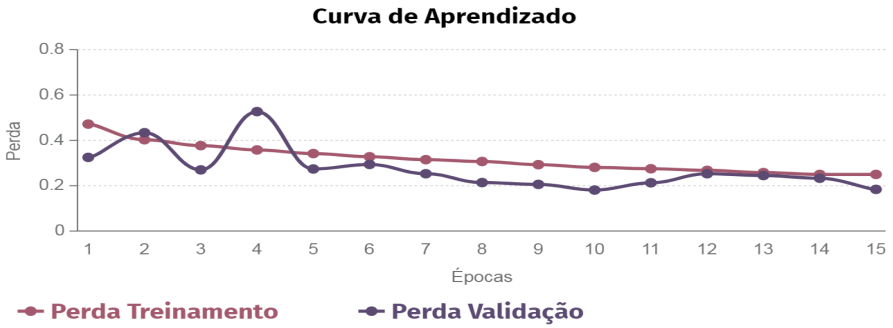

Figura 5.17: Curva de Aprendizado - ResNet - Experimento III.

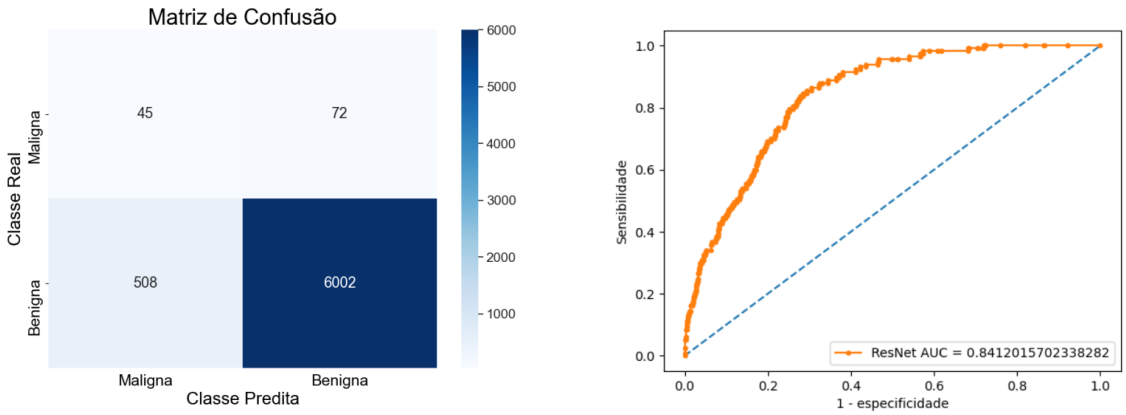

Figura 5.18: Matriz de Confusão e ROC - ResNet - Experimento III.

O Experimento III apresentou melhores resultados em comparação ao Experimento II pois todas as arquiteturas nesse experimento tiveram um aumento no valor da acurácia balanceada, visto que que começaram a acertar classificações de imagens da classe positiva. Nesse experimento, a quantidade de imagens de cada classe foi equilibrada, o que poderia fazer com que o enviesamento para a classe negativa fosse eliminado, entretanto, isso não ocorreu, embora a quantidade de imagens da classe positiva tenha sido aumentada de forma a igualar a quantidade de imagens da classe negativa, os modelos arquiteturais

continuam dando preferência na classificação para as imagens da classe negativa, isso pode ser observado nos valores de especificidade que se apresentam muito superiores aos de sensibilidade, além disso, os valores da função de perda para o conjunto de validação mantêm-se abaixo dos valores da função de perda para o conjunto de treinamento, o que pode ser verificado nas Figuras 5.13, 5.15 e 5.17. Como o conjunto de validação não foi balanceado, ou seja, contém uma proporção maior de imagens da classe negativa, e a taxa de acerto para esse conjunto continua mais alta, haja vista que a função de perda para ele é menor, a ideia de que as arquiteturas continuam enviesadas para a classe negativa é corroborada.

5.4 Experimento IV

Dando continuidade aos procedimentos experimentais, essa etapa analisa o desempenho das arquiteturas mediante a utilização combinada das técnicas de *transfer learning* e *data augmentation*. A Tabela 5.4 resume os resultados obtidos nessa etapa experimental para o conjunto de teste, e, em seguida, são apresentados os resultados para as arquiteturas utilizadas.

Tabela 5.4: Resumo dos Resultados - Experimento IV (Com DA, Com TL).

Métricas de Avaliação	**AlexNet**	**GoogLenet**	**ResNet**
Acurácia Balanceada	0,738	0,743	0,746
Sensibilidade	0,607	0,641	0,667
Especificidade	0,869	0,845	0,826
AUC	0,825	0,842	0,840
Acurácia	0,864	0,842	0,823

A AlexNet foi treinada por 15 épocas. A melhor época foi a de número 14 com perda no conjunto de validação de 0,321 como pode ser visto na Figura 5.19. Pela primeira vez, o valor da sensibilidade ultrapassou a marca de 0,6, indicando que a arquitetura está realmente aprendendo a fazer predições corretas da classe alvo. Em comparação ao Experimento III, a AlexNet apresenta aqui, uma melhora na acurácia geral de mais de 10 pontos percentuais, o que demonstra a eficiência das técnicas combinadas. O tempo de processamento foi de 0,04 segundos por imagem sobre o conjunto de teste. A Figura 5.20 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

A GoogLeNet foi treinada por 13 épocas. A melhor época foi a de número 8 com perda no conjunto de validação de 0,363 como pode ser visto na Figura 5.21. Assim como a AlexNet, a GoogLeNet também apresentou melhoras significativas em relação ao experimento anterior, apresentando uma acurácia balanceada de mais de 74%. O tempo

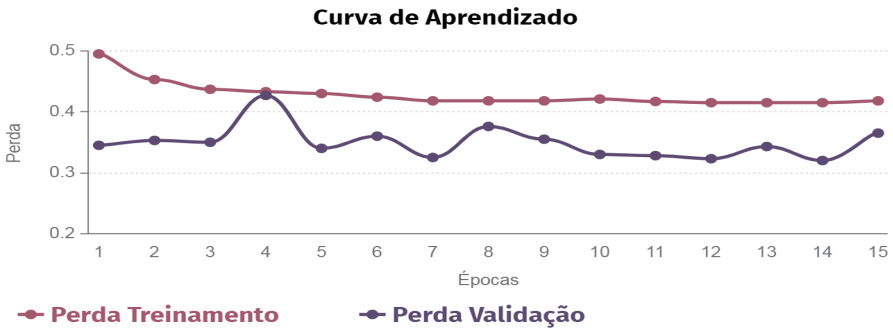

Figura 5.19: Curva de Aprendizado - AlexNet - Experimento IV .

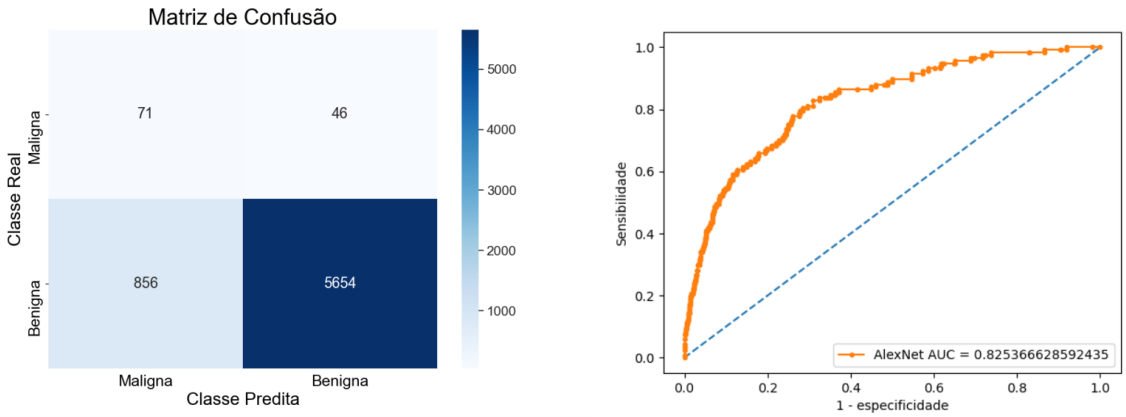

Figura 5.20: Matriz de Confusão e ROC - AlexNet - Experimento IV .

de processamento foi de 0,09 segundos por imagem sobre o conjunto de teste. A Figura 5.22 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

A ResNet foi treinada por 6 épocas. A melhor época foi a de número 1 com perda no conjunto de validação de 0,390 como pode ser visto na Figura 5.23. Por fim, a arquitetura ResNet também apresentou melhorias. Embora o valor da especificidade tenha sofrido uma leve diminuição, a sensibilidade e a acurácia balanceada aumentaram. O tempo de processamento foi de 0,05 segundos por imagem sobre o conjunto de teste. A Figura 5.24 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

Nesse experimento, em comparação ao experimento anterior, a técnica de *transfer learning* passou a ser utilizada. Como aqui as classes já não estão mais desbalanceadas, a ofuscação da técnica de *transfer learning* causada pelo desbalanceamento das classes que ocorreu no Experimento II não teve um impacto tão grande, e, inclusive, foi possível

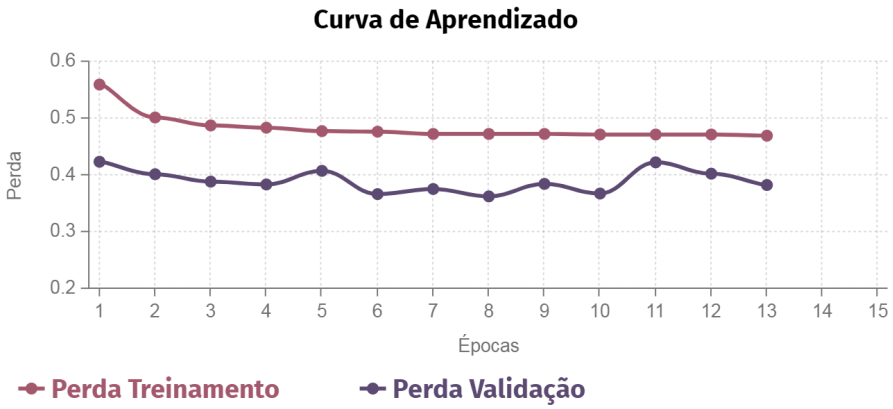

Figura 5.21: Curva de Aprendizado - GoogLeNet - Experimento IV .

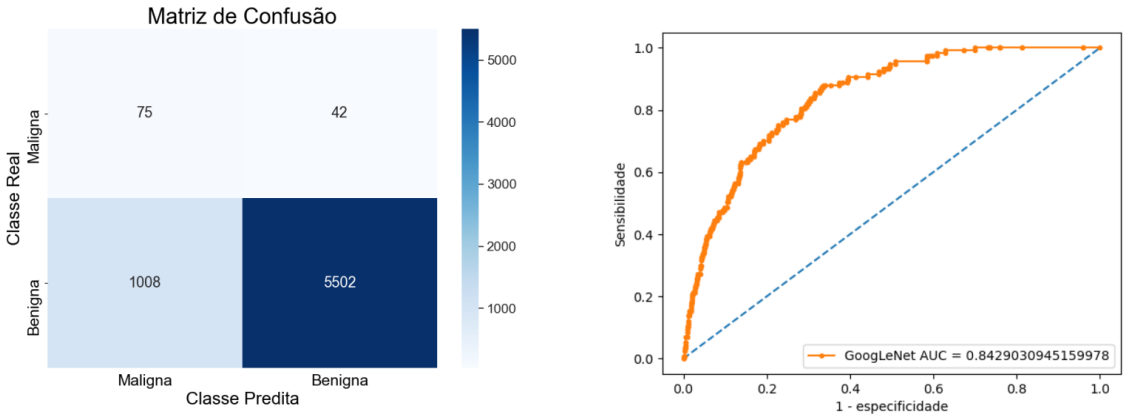

Figura 5.22: Matriz de Confusão - GoogLeNet - Experimento IV .

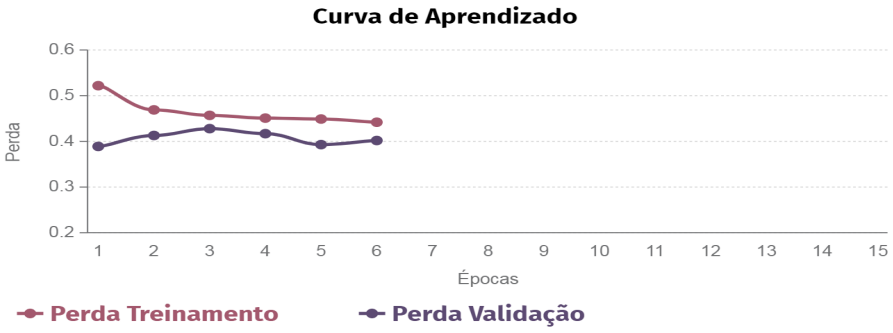

Figura 5.23: Curva de Aprendizado - ResNet - Experimento IV .

observar uma melhora significativa nos resultados. Os valores de sensibilidade e acurácia para as três arquiteturas tiveram aumentos consideráveis. A arquitetura que obteve os

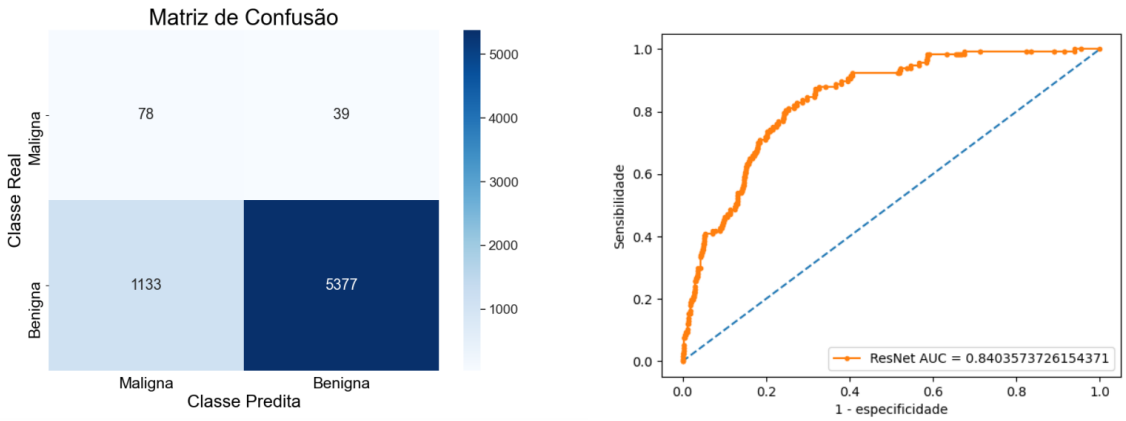

Figura 5.24: Matriz de Confusão e ROC - ResNet - Experimento IV .

melhores resultados foi a ResNet com acurácia balanceada de 0,738.

5.5 Experimento V

Considerando o melhor resultado obtido nos experimentos anteriores, e, seguindo a metodologia descrita na Seção 4.3.5, a primeira etapa desse experimento consiste em utilizar uma quantidade diferente de amostras de cada classe. O número de imagens benignas é reduzido em 50%, e, após isso, é aplicada a técnica de *data augmentation* sobre a classe maligna até que a quantidade de amostras em cada classe seja igualada. Esse procedimento será realizado apenas com a arquitetura ResNet que foi a que obteve a maior acurácia balanceada nos experimentos anteriores, 0,0746. A Tabela 5.5 resume os resultados obtidos nessa etapa experimental para o conjunto de teste, e, em seguida, são apresentados os resultados obtidos para a arquitetura ResNet.

Tabela 5.5: Resumo dos resultados - Experimento V.

Métricas de Avaliação	**ResNet Etapa I**	**ResNet Etapa II**
Acurácia Balanceada	0,725	0,752
Sensibilidade	0,581	0,752
Especificidade	0,864	0,753
AUC	0,823	0,839
Acurácia	0,859	0,753

Para a primeira etapa, a melhor época foi a de número 4 com perda no conjunto de validação de 0,312. É possível observar na Figura 5.25 que os valores para da perda do conjunto de validação continuam inferiores aos valores da perda do conjunto de treinamento, essa relação permanece durante todas as nove épocas da etapa de treinamento.

Embora a classe benigna tenha sido reduzida em 50%, como pode ser visto na Tabela 4.4, as predições para as imagens do conjunto de validação continuam mais precisas do que as predições para as imagens do conjunto de treinamento, uma vez que, para esse conjunto, a perda é menor, o que sugere que mesmo após a redução da quantidade de amostras das classes, a rede continua enviesada para as amostras da classe negativa.

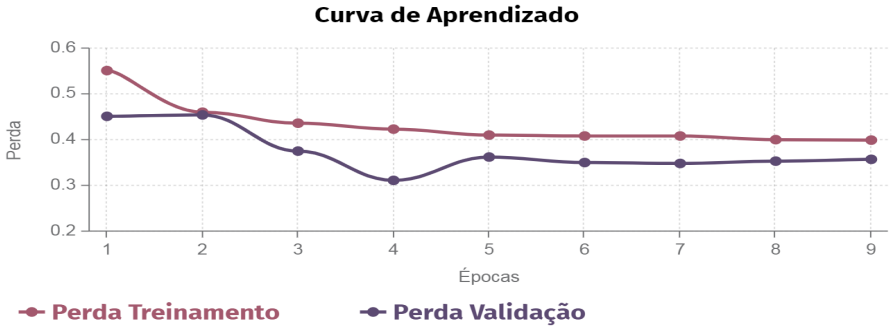

Figura 5.25: Curva de Aprendizado - ResNet - Experimento V (Diminuição da classe negativa em 50%).

Os resultados obtidos nessa etapa não foram superiores aos valores obtidos no Experimento IV para a ResNet considerando a acurácia balanceada como métrica de avaliação principal, tanto a sensibilidade como a acurácia balanceada sofreram diminuição em seus valores. A diminuição das amostras, entretanto, promoveu um aumento no valor da acurácia e da especificidade. O tempo de processamento foi de 0,05 segundos por imagem sobre o conjunto de teste. A Figura 5.26 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

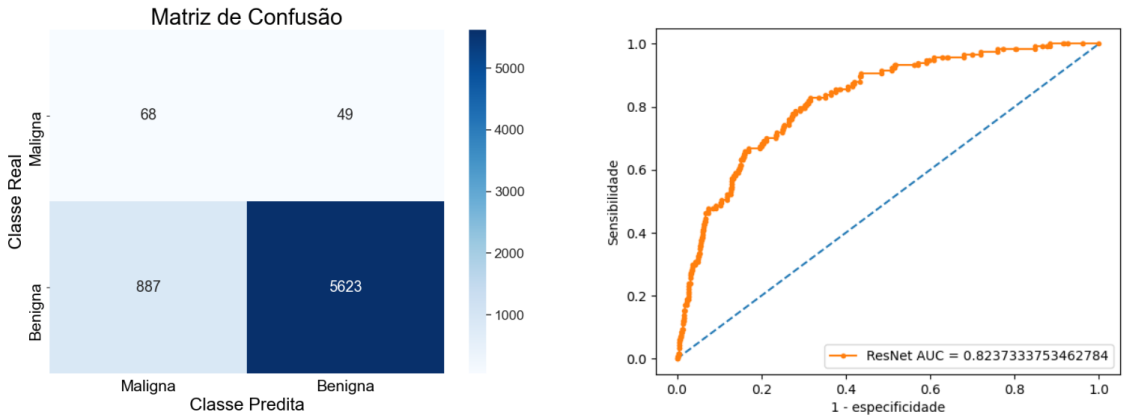

Figura 5.26: Matriz de Confusão e ROC - ResNet - Experimento V (Diminuição da classe negativa em 50%).

Por fim, a última etapa desse experimento consiste no aumento na quantidade de amostras da classe maligna em 100%, como pode ser visto na Tabela 4.5. A rede foi treinada por 12 épocas e a melhor época foi a de número 7 com perda no conjunto de validação de 0,517.

Pela primeira vez em todos os experimentos realizados, como pode ser visto na Figura 5.27, a perda do conjunto de treinamento apresenta valores menores do que a perda do conjunto de validação. Analisando apenas a relação entre as perdas, é possível perceber que a rede apresenta predições mais precisas no conjunto de treinamento do que no conjunto de validação, algo ainda não visto nos experimentos anteriores. Esses resultados sugerem uma diminuição no enviesamento em favor da classe negativa como discutido na Seção 5.3.

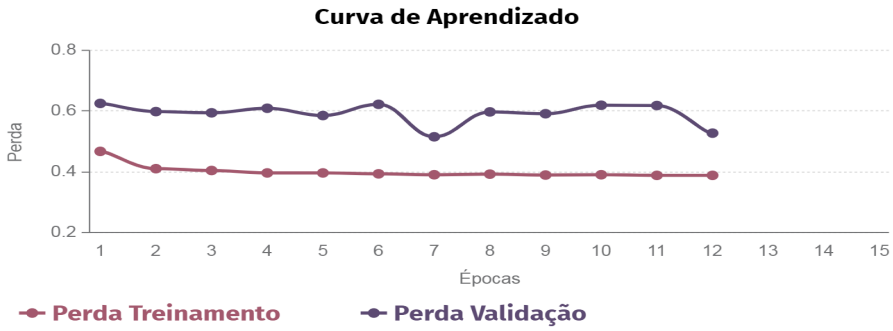

Figura 5.27: Curva de Aprendizado - ResNet -Experimento V (Aumento da classe positiva em 100%) .

Em comparação com os melhores resultados dos experimentos IV, a sensibilidade, durante essa etapa, teve um aumento de aproximadamente 8% e a especificidade uma redução aproximada na mesma proporção. Esses resultados eram esperados uma vez que a quantidade de amostras da classe positiva aumentou e a quantidade de amostras da classe negativa permaneceu inalterada. Essas variações proporcionaram um aumento na acurácia balanceada de aproximadamente 0,5%, fazendo com que o modelo dessa etapa seja o melhor modelo dentre todos os experimentos. O tempo de processamento foi de 0,05 segundos por imagem sobre o conjunto de teste. A Figura 5.28 apresenta a matriz de confusão e o gráfico da curva ROC obtidas sobre o conjunto de teste para essa arquitetura.

Esse experimento tentou otimizar o melhor resultado encontrado até o experimento IV. O resultado obtido para a primeira etapa não promoveu melhora nos resultados mas os resultados da segunda etapa foram suficientes para aumentar a acurácia balanceada do melhor modelo em aproximadamente 0,5%. Na segunda etapa também foi possível observar pela primeira vez a inversão no gráfico das perdas como visto na Figura 5.27.

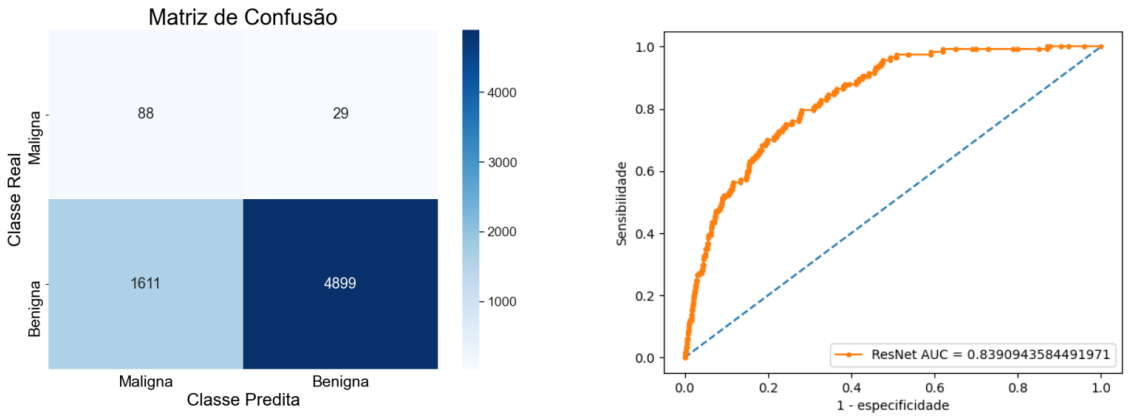

Figura 5.28: Matriz de Confusão e ROC - ResNet - Experimento V (Aumento da classe positiva em 100%) .

5.6 Discussão dos Resultados

Os resultados observados durante toda a etapa experimental foram diretamente afetados pelo grande desbalanceamento das classes. Durante os Experimentos I e II, em que a técnica de *data augmentation* ainda não tinha sido utilizada, esse desbalanceamento foi responsável por fazer com que as arquiteturas ficassem extremamente tendenciosas a classificar as amostras apenas como benignas, como observado na Tabela 5.1. Em especial, durante o Experimento II, o impacto do desbalanceamento foi suficiente para impedir que a técnica de *transfer learning* promovesse uma melhora na acurácia balanceada dos modelos como pode ser observado na Tabela 5.2. Mesmo com a aplicação da técnica de *transfer learning*, a capacidade de classificação dos modelos permaneceu igual ao observado no Experimento I, classificando todas as amostras como benignas. Como a acurácia balanceada observada nas arquiteturas para esses dois experimentos iniciais foi de 0,5, a chance de acerto para uma amostra qualquer foi de 50%.

A partir do Experimento III foi possível observar uma melhora na capacidade dedutiva das arquiteturas, uma vez que, a partir desse experimento, ocorreram tentativas de minimizar o impacto do desbalanceamento por meio da técnica de *data augmentation*. Nesse experimento, as arquiteturas deixaram de apresentar resultados similares a uma classificação aleatória uma vez que a acurácia balanceada, como mostrado na Tabela 5.3, atingiu valores superiores a 0,62.

Durante o Experimento IV, em que as técnicas de *data augmentation* e *transfer learning* foram aplicadas simultaneamente, foi possível observar o real impacto da técnica de *transfer learning* que durante o Experimento II não foi suficiente para promover aumento na acurácia dos modelos. Quando o impacto do desbalanceamento foi minimizado pela

técnica de *data augmentation*, a técnica de *transfer learning* foi capaz de promover, para todas as arquiteturas, um aumento da acurácia balanceada em mais de 7% em relação ao Experimento III e em mais de 20% em relação aos Experimentos I e II. Ao final desse experimento, comparando os resultados com os obtidos nos Experimentos I, II e III, foi possível determinar a arquitetura com melhor desempenho. A ResNet foi a que apresentou os melhores resultados. A acurácia balanceada para ela foi de 0,746, resultado quase 25% superior ao observado nos Experimentos I e II.

Durante o Experimento V, foi feita uma tentativa de otimização de desempenho para a arquitetura que apresentou os melhores resultados, ResNet. A diminuição da quantidade de amostras das classes na etapa I não foi suficiente para promover um aumento na acurácia balanceada, entretanto, o aumento de amostras da classe maligna na etapa II promoveu um aumento na acurácia balanceada e na sensibilidade. A acurácia balanceada foi aumentada em cerca de 0,5% após essa etapa.

Importante destacar que os valores de AUC para todos os experimentos não sofreram tanta variação, o pior valor de AUC, 0,76, foi obtido para a arquitetura AlexNet durante o Experimento II, o melhor 0,85, para a arquitetura GoogLeNet durante o Experimento III e o AUC obtido para a arquitetura tomada como vencedora nos experimentos foi de aproximadamente 0,84. A competição SIIM-ISIC de onde a base de dados foi retirada, utiliza a AUC como métrica principal de avaliação das submissões, o vencedor conseguiu atingir um valor de 0,93. A comparação do melhor resultado obtido nesse trabalho com o melhor resultado obtido na competição fica prejudicada uma vez que os conjuntos de testes utilizados são diferentes, as imagens de teste da competição não foram utilizadas nesse trabalho como descrito na Seção 4.1.1. A Figura 5.29 compila os resultados obtidos em todas as etapas experimentais.

Experimento	Acurácia Balanceada	Sensibilidade	Especificidade	AUC	Acurácia
Exp. I (SEM DA, SEM TL) - AlexNet	0,500	0,000	**1,000**	0,767	**0,982**
Exp. I (SEM DA, SEM TL) - GoogLeNet	0,500	0,000	**1,000**	0,800	**0,982**
Exp. I (SEM DA, SEM TL) - ResNet	0,500	0,000	**1,000**	0,798	**0,982**
Exp. II (SEM DA, COM TL) - AlexNet	0,500	0,000	**1,000**	0,767	**0,982**
Exp. II (SEM DA, COM TL) - GoogLeNet	0,500	0,000	**1,000**	0,829	**0,982**
Exp. II (SEM DA, COM TL) - ResNet	0,500	0,000	**1,000**	0,841	**0,982**
Exp. III (COM DA, SEM TL) - AlexNet	0,628	0,333	0,923	0,819	0,912
Exp. III (COM DA, SEM TL) - GoogLeNet	0,661	0,402	0,920	**0,854**	0,911
Exp. III (COM DA, SEM TL) - ResNet	0,653	0,385	0,922	0,841	0,912
Exp. IV (COM DA, COM TL) - AlexNet	0,738	0,607	0,869	0,825	0,864
Exp. IV (COM DA, COM TL) - GoogLeNet	0,743	0,641	0,845	0,842	0,842
Exp. IV (COM DA, COM TL) - ResNet	0,746	0,667	0,826	0,840	0,823
Exp. V (COM DA, COM TL) - RESNET(-50% BENIG.)	0,725	0,581	0,864	0,823	0,859
Exp. V (COM DA, COM TL) - RESNET(+100% MALIG.)	**0,752**	**0,752**	0,753	0,839	0,753

Figura 5.29: Resumo do Resultado Final.

Capítulo 6

Conclusão

A aplicação combinada das técnicas de *data augmentation* e *transfer learning* foram capazes de promover um aumento de mais de 25% na acurácia balanceada das arquiteturas examinadas no presente trabalho. Apesar disso, a utilização da técnica de *transfer learning*, isoladamente, para a base de dados escolhida, não foi suficiente para promover um aumento do desempenho. O desbalanceamento das classes teve um impacto muito maior do que o esperado, de forma que melhorias nas arquitetura só foram observadas quando o problema do desbalanceamento começou a ser diretamente atacado. A arquitetura que obteve o melhor desempenho foi a ResNet com uma acurácia balanceada de 0,752, sensibilidade de 0,752, especificidade de 0,753 e valor AUC de 0,839. Considerando que o desbalanceamento das classes é significativo, um resultado em que tanto a sensibilidade como a especificidade apresentam valores aproximadamente iguais para a arquitetura com maior acurácia balanceada é um resultado promissor. Embora durante os experimentos um maior desempenho fosse buscado, o principal objetivo do trabalho não foi a tentativa de promover a máxima otimização das arquiteturas de forma a obter o maior desempenho possível, mas verificar o impacto das técnicas estudadas no desempenho das redes neurais escolhidas. Entretanto, na tentativa de otimizar os resultados obtidos no presente trabalho, seria possível partir para duas principais abordagens: realizar modificações na técnica de *data augmentation* ou modificar o processo de treinamento.

Esse trabalho utilizou técnicas específicas para o aumento de amostras durante a aplicação da técnica de *data augmentation* como pode ser visto na Seção 4.1.2. Outras técnicas não tradicionais poderiam ser utilizadas durante o processo de transformação das amostras, como por exemplo, a geração de imagens utilizando GANs ou mesmo a utilização de uma rede neural auxiliar focada na geração de amostras [45]. Alternativamente, as técnicas de *data augmentation* que foram utilizadas permaneceriam iguais mas os valores mínimos e máximos e suas probabilidades de aplicação, como mostrados na Tabela 4.6, seriam modificados.

Outra possibilidade seria modificar o treinamento das redes neurais, especificamente no cálculo da função de perda. Seria possível a utilização de pesos para as classes de forma a definir um peso maior para a classe com menor quantidade de amostras com o objetivo de penalizar a rede neural em uma situação em que uma imagem da classe minoritária é classificada incorretamente. Uma metodologia de pesos para a função de custo que utiliza essa ideia pode ser vista no trabalho de Ho e Wookey [63]. Esse procedimento seria uma tentativa de combater diretamente o desbalanceamento das classes. Outra opção seria a otimização dos hiper parâmetros definidos no presente trabalho, como a taxa de *learning rate*, o tamanho do *batch* ou até mesmo o otimizador utilizado. Além disso, seria interessante testar outras proporções nas divisões dos conjuntos de treinamento, validação e testes que foram mantidas fixas no presente trabalho de forma a reduzir a quantidade de experimentos.

A combinação de diversas arquiteturas por meio de técnicas como *ensemble* também seria possível, nessa situação, diversas arquiteturas seriam treinadas simultaneamente e as saídas fornecidas por cada uma seriam combinadas de forma a diminuir a variabilidade das respostas das redes [64].

Referências

[1] Turing, Alan M.: *Computing machinery and intelligence*. Mind, 59(October):433–60, 1950. Publisher: Oxford University Press. 1, 5

[2] Samuel, A. L.: *Some Studies in Machine Learning Using the Game of Checkers*. IBM Journal of Research and Development, 3(3):210–229, 1959, ISSN 0018-8646. Conference Name: IBM Journal of Research and Development. 1

[3] Hayman, S.: *The McCulloch-Pitts model*. Em *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339)*, volume 6, páginas 4438–4439 vol.6, 1999. ISSN: 1098-7576. 1

[4] Rosenblatt, F.: *The perceptron: A probabilistic model for information storage and organization in the brain.* Psychological Review, 65(6):386–408, 1958, ISSN 1939-1471(ELECTRONIC),0033-295X(PRINT). Place: US Publisher: American Psychological Association. 1, 6, 7

[5] Shortliffe, Edward H., Randall Davis, Stanton G. Axline, Bruce G. Buchanan, C. Cordell Green e Stanley N. Cohen: *Computer-based consultations in clinical therapeutics: Explanation and rule acquisition capabilities of the MYCIN system*. Computers and Biomedical Research, 8(4):303–320, 1975, ISSN 0010-4809. `https://www.sciencedirect.com/science/article/pii/0010480975900099`, acesso em 2021-05-15. 1

[6] Roehrig, Jimmy, Takeshi Doi, Akira Hasegawa, Bob Hunt, Julian Marshall, Harlan Romsdahl, Alex Schneider, Ron Sharbaugh e Wei Zhang: *Clinical Results with R2 Imagechecker System*. Em Karssemeijer, Nico, Martin Thijssen, Jan Hendriks e Leon van Erning (editores): *Digital Mammography: Nijmegen, 1998*, Computational Imaging and Vision, páginas 395–400. Springer Netherlands, Dordrecht, 1998, ISBN 978-94-011-5318-8. `https://doi.org/10.1007/978-94-011-5318-8_64`, acesso em 2021-05-15. 1

[7] Haenssle, H. A., C. Fink, R. Schneiderbauer, F. Toberer, T. Buhl, A. Blum, A. Kalloo, A. Ben Hadj Hassen, L. Thomas, A. Enk, L. Uhlmann, Reader study level-I and level-II Groups, Christina Alt, Monika Arenbergerova, Renato Bakos, Anne Baltzer, Ines Bertlich, Andreas Blum, Therezia Bokor-Billmann, Jonathan Bowling, Naira Braghiroli, Ralph Braun, Kristina Buder-Bakhaya, Timo Buhl, Horacio Cabo, Leo Cabrijan, Naciye Cevic, Anna Classen, David Deltgen, Christine Fink, Ivelina Georgieva, Lara Elena Hakim-Meibodi, Susanne Hanner, Franziska Hartmann, Julia Hartmann, Georg Haus, Elti Hoxha, Raimonds Karls, Hiroshi Koga, Jürgen Kreusch, Aimilios

Lallas, Pawel Majenka, Ash Marghoob, Cesare Massone, Lali Mekokishvili, Dominik Mestel, Volker Meyer, Anna Neuberger, Kari Nielsen, Margaret Oliviero, Riccardo Pampena, John Paoli, Erika Pawlik, Barbar Rao, Adriana Rendon, Teresa Russo, Ahmed Sadek, Kinga Samhaber, Roland Schneiderbauer, Anissa Schweizer, Ferdinand Toberer, Lukas Trennheuser, Lyobomira Vlahova, Alexander Wald, Julia Winkler, Priscila Wölbing e Iris Zalaudek: *Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists*. Annals of Oncology: Official Journal of the European Society for Medical Oncology, 29(8):1836–1842, 2018, ISSN 1569-8041. 2, 3

[8] Adegun, Adekanmi e Serestina Viriri: *Deep learning techniques for skin lesion analysis and melanoma cancer detection: a survey of state-of-the-art*. Artificial Intelligence Review, 54(2):811–841, 2021, ISSN 1573-7462. 2

[9] *Câncer de pele melanoma*, 2018. `https://www.inca.gov.br/tipos-de-cancer/cancer-de-pele-melanoma`, acesso em 2021-05-13. 2

[10] Jojoa Acosta, Mario Fernando, Liesle Yail Caballero Tovar, Maria Begonya Garcia-Zapirain e Winston Spencer Percybrooks: *Melanoma diagnosis using deep learning techniques on dermatoscopic images*. BMC Medical Imaging, 21(1):6, 2021, ISSN 1471-2342. 3, 9, 28, 40

[11] Pham, Tri Cong, Chi Mai Luong, Muriel Visani e Van Dung Hoang: *Deep CNN and Data Augmentation for Skin Lesion Classification*. Em Nguyen, Ngoc Thanh, Duong Hung Hoang, Tzung Pei Hong, Hoang Pham e Bogdan Trawiński (editores): *Intelligent Information and Database Systems*, páginas 573–582, Cham, 2018. Springer International Publishing, ISBN 978-3-319-75420-8. 3, 22, 27

[12] Hosny, Khalid M., Mohamed A. Kassem e Mohamed M. Foaud: *Classification of skin lesions using transfer learning and augmentation with Alex-net*. PLOS ONE, 14(5):e0217293, 2019, ISSN 1932-6203. Publisher: Public Library of Science. 3, 29

[13] Mitchell, Thomas M.: *Machine Learning*. McGraw-Hill, Inc., USA, 1ª edição, 1997, ISBN 978-0-07-042807-2. 5, 6, 7, 9

[14] Géron, Aurlien: *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 1st edição, 2017, ISBN 1-4919-6229-1. 6, 7, 8, 12, 13, 15, 16

[15] Rumelhart, David E., Geoffrey E. Hinton e Ronald J. Williams: *Learning representations by back-propagating errors*. \nat, 323(6088):533–536, 1986. 9

[16] Mittal, Ansh, Anu Soorya, Preeti Nagrath e D. Jude Hemanth: *Data augmentation based morphological classification of galaxies using deep convolutional neural network*. Earth Science Informatics, 13(3):601–617, 2020, ISSN 1865-0481. 9, 12, 13

[17] Fukushima, Kunihiko: *Neocognitron: A hierarchical neural network capable of visual pattern recognition*. Neural networks, 1(2):119–130, 1988. Publisher: Elsevier. 9

[18] Hubel, D. H. e T. N. Wiesel: *Receptive fields, binocular interaction and functional architecture in the cat's visual cortex*. The Journal of Physiology, 160(1):106–154, 1962, ISSN 1469-7793. Publisher: John Wiley & Sons, Ltd. 10

[19] LeCun, Yann e Corinna Cortes: *The MNIST DATABASE of handwritten digits*. 2010. 10, 26, 27

[20] Goodfellow, Ian, Yoshua Bengio e Aaron Courville: *Deep Learning*. MIT Press, 2016. 10, 11, 12, 13

[21] Aljaafari, Nura: *Ichthyoplankton Classification Tool using Generative Adversarial Networks and Transfer Learning*. Thesis, abril 2018. `https://repository.kaust.edu.sa/handle/10754/627578`, acesso em 2021-05-16, Accepted: 2018-04-19T13:08:13Z. 14

[22] Ruder, Sebastian: *An overview of gradient descent optimization algorithms*. arXiv:1609.04747 [cs], 2017. arXiv: 1609.04747. 16

[23] Ho, Yaoshiang e Samuel Wookey: *The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling*. IEEE Access, 8:4806–4813, 2020, ISSN 2169-3536. Conference Name: IEEE Access. 17

[24] Khan, Asifullah, Anabia Sohail, Umme Zahoora e Aqsa Saeed Qureshi: *A survey of the recent architectures of deep convolutional neural networks*. Artificial Intelligence Review, 53(8):5455–5516, 2020, ISSN 1573-7462. Publisher: Springer Science and Business Media LLC. 18

[25] Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg e Li Fei-Fei: *ImageNet Large Scale Visual Recognition Challenge*. International Journal of Computer Vision (IJCV), 115(3):211–252, 2015. 18, 19, 31

[26] Krizhevsky, Alex, Ilya Sutskever e Geoffrey E. Hinton: *ImageNet Classification with Deep Convolutional Neural Networks*. Commun. ACM, 60(6):84–90, 2017, ISSN 0001-0782. Place: New York, NY, USA Publisher: Association for Computing Machinery. 18, 37

[27] Alom, Md. Zahangir, Tarek Taha, Christopher Yakopcic, Stefan Westberg, Mahmudul Hasan, Brian Esesn, Abdul Awwal e Vijayan Asari: *The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches*. 2018. 18, 19

[28] Nayak, Sunita: *Understanding AlexNet*, 2018. 18

[29] Szegedy, C., Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke e A. Rabinovich: *Going deeper with convolutions*. Em *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 1–9, 2015. ISSN: 1063-6919. 19, 37

[30] Zisserman, Andrew: *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv 1409.1556, 2014. 19

[31] Deng, Jia, Wei Dong, Richard Socher, Li Jia Li, Kai Li e Li Fei-Fei: *Imagenet: A large-scale hierarchical image database*. Em *2009 IEEE conference on computer vision and pattern recognition*, páginas 248–255. Ieee, 2009. 20, 28, 29

[32] He, Kaiming, Xiangyu Zhang, Shaoqing Ren e Jian Sun: *Deep Residual Learning for Image Recognition*. 2015. _eprint: 1512.03385. 20, 37

[33] Xu, Joyce: *Guía para arquitecturas de redes profundas*, 2018. 20

[34] Pan, S. J. e Q. Yang: *A Survey on Transfer Learning*. IEEE Transactions on Knowledge and Data Engineering, 22(10):1345–1359, 2010, ISSN 1558-2191. 21

[35] Wang, Yaqing, Quanming Yao, James T. Kwok e Lionel M. Ni: *Generalizing from a Few Examples: A Survey on Few-Shot Learning*. ACM Comput. Surv., 53(3), 2020, ISSN 0360-0300. Place: New York, NY, USA Publisher: Association for Computing Machinery. 21

[36] Tsiakmaki, Maria, Georgios Kostopoulos, Sotiris Kotsiantis e Omiros Ragos: *Transfer Learning from Deep Neural Networks for Predicting Student Performance*. Applied Sciences, 10(6), 2020, ISSN 2076-3417. 21

[37] Peng, Peng e Jiugen Wang: *How to fine-tune deep neural networks in few-shot learning?* CoRR, abs/2012.00204, 2020. _eprint: 2012.00204. 22

[38] Codella, N, D Gutman, ME Celebi, B Helba, MA Marchetti, S Dusza, A Kalloo, K Liopyris, N Mishra, H Kittler e A Halpern: *Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)*. 2017. 22, 27, 28, 29

[39] Perez, Fábio, Cristina Vasconcelos, Sandra Avila e Eduardo Valle: *Data Augmentation for Skin Lesion Analysis*. 2018. 22

[40] Hossin, Mohammad e Sulaiman M.N: *A Review on Evaluation Metrics for Data Classification Evaluations*. International Journal of Data Mining & Knowledge Management Process, 5:01–11, 2015. 23, 24, 25

[41] Chawla, Nitesh, Nathalie Japkowicz e Aleksander Kołcz: *Editorial: Special Issue on Learning from Imbalanced Data Sets*. SIGKDD Explorations, 6:1–6, 2004. 23

[42] Gupta, Akhilesh, Nesime Tatbul, Ryan Marcus, Shengtian Zhou, Insup Lee e Justin Gottschlich: *Class-Weighted Evaluation Metrics for Imbalanced Data Classification*. 2020. _eprint: 2010.05995. 24

[43] Zou, Kelly H., A. James O'Malley e Laura Mauri: *Receiver-Operating Characteristic Analysis for Evaluating Diagnostic Tests and Predictive Models*. Circulation, 115(5):654–657, 2007. _eprint: https://www.ahajournals.org/doi/pdf/10.1161/CIRCULATIONAHA.105.594929. 25

[44] Paulino, Arthur: *Área Abaixo da Curva ROC*, junho 2018. `https://miro.medium.com/max/548/1*VpWTqZi3i4v2HxavnBN0Uw.png`, acesso em 2021-03-24. 25

[45] Perez, Luis e Jason Wang: *The Effectiveness of Data Augmentation in Image Classification using Deep Learning*. 2017. 26, 34, 60

[46] Li, Fei Fei, Andrej Karpathy e Justin Johnson: *Tiny ImageNet*, 2017. 26

[47] Gu, Shanqing, Manisha Pednekar e Robert Slater: *Improve Image Classification Using Data Augmentation and Neural Networks*. SMU Data Science Review, 2(2), 2019. 27

[48] Krizhevsky, Alex: *Learning Multiple Layers of Features from Tiny Images*. 2009. 27

[49] Hu, Benlin, Cheng Lei, Dong Wang, Shu Zhang e Zhenyu Chen: *A Preliminary Study on Data Augmentation of Deep Learning for Image Classification*. 2019. 27

[50] Taylor, Luke e Geoff Nitschke: *Improving Deep Learning using Generic Data Augmentation*. 2017. 28

[51] Fei-Fei, L, R Fergus e P Perona: *Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories.* páginas 178–178, Washington, DC, USA, 2004. 28

[52] Menegola, Afonso, Michel Fornaciali, Ramon Pires, Flavia Vasques Bittencourt, Sandra Avila e Eduardo Valle: *Knowledge transfer for melanoma screening with deep learning*. 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017), 2017. ISBN: 9781509011728 Publisher: IEEE. 29

[53] Argenziano, Giuseppe, Peter Soyer, Vincenzo De Giorgi, Domenico Piccolo, P. Carli, M. Delfino, Angela Ferrari, Rainer Hofmann-Wellenhof, Daniela Massi, G. Mazzocchetti, M. Scalvenzi e Ingrid Wolf: *Interactive atlas of dermoscopy. Dermoscopy: a tutorial (Book) and CD-ROM.* 2000. 29

[54] Gutman, David, Noel C. F. Codella, Emre Celebi, Brian Helba, Michael Marchetti, Nabin Mishra e Allan Halpern: *Skin Lesion Analysis toward Melanoma Detection: A Challenge at the International Symposium on Biomedical Imaging (ISBI) 2016, hosted by the International Skin Imaging Collaboration (ISIC)*. 2016. _eprint: 1605.01397. 29

[55] Mahbod, Amirreza, Gerald Schaefer, Chunliang Wang, Rupert Ecker, Georg Dorffner e Isabella Ellinger: *Investigating and Exploiting Image Resolution for Transfer Learning-based Skin Lesion Classification.* arXiv:2006.14715 [cs], 2020. arXiv: 2006.14715. 29

[56] Barata, Catarina, M. Emre Celebi e Jorge S. Marques: *Improving dermoscopy image classification using color constancy.* IEEE journal of biomedical and health informatics, 19(3):1146–1152, 2015, ISSN 2168-2208. 29

[57] Giotis, Ioannis, Nynke Molders, Sander Land, Michael Biehl, Marcel F. Jonkman e Nicolai Petkov: *MED-NODE: A computer-assisted melanoma diagnosis system using non-dermoscopic images*. Expert Systems with Applications, 42(19):6578–6585, 2015, ISSN 0957-4174. 29

[58] Filali, Idir: *Contrast based lesion segmentation on DermIS and DermQuest datasets.* 2, 2019. Publisher: Mendeley. 29

[59] Zhang, Runyuan: *Melanoma Detection Using Convolutional Neural Network*. Em *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, páginas 75–78, 2021. 30

[60] Tan, Mingxing e Quoc V. Le: *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. 2020. _eprint: 1905.11946. 30

[61] Rotemberg, V, N Kurtansky, B Betz-Stablein, L Caffery, E Chousakos, N Codella, M Combalia, S Dusza, P Guitera, D Gutman, A Halpern, B Helba, H Kittler, K Kose, S Langer, K Lioprys, J Malvehy, S Musthaq, J Nanda, O Reiter, G Shih, A Stratigos, P Tschandl, J Weber e P Soyer: *A patient-centric dataset of images and metadata for identifying melanomas using clinical context. Sci Data 8, 34 (2021).https://doi.org/10.1038/s41597-021-00815-z*. 2021. 30, 33, 36

[62] Bloice, Marcus D, Peter M Roth e Andreas Holzinger: *Biomedical image augmentation using Augmentor*. Bioinformatics, 35(21):4522–4524, 2019, ISSN 1367-4803. _eprint: https://academic.oup.com/bioinformatics/article-pdf/35/21/4522/30330763/btz259.pdf. 34

[63] Ho, Yaoshiang e Samuel Wookey: *The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling*. IEEE Access, PP:1–1, dezembro 2019. 61

[64] Dietterich, Thomas G.: *Ensemble Methods in Machine Learning*. Em *Multiple Classifier Systems*, páginas 1–15, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg, ISBN 978-3-540-45014-6. 61