Identificação da Psoríase através de Dispositivos Móveis usando Redes Neurais Convolucionais Profundas

Gabriel Lenin Silva Lima - 42888

Dissertação apresentada à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Sistemas de Informação.

Trabalho orientado por:

Prof. Rui Pedro Lopes

Prof. Arlete Teresinha Beuren

Bragança

2019-2020

Identificação da Psoríase através de Dispositivos Móveis usando Redes Neurais Convolucionais Profundas

Gabriel Lenin Silva Lima - 42888

Dissertação apresentada à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Sistemas de Informação.

Trabalho orientado por:

Prof. Rui Pedro Lopes

Prof. Arlete Teresinha Beuren

Bragança

2019-2020

Dedicatória

Dedico este trabalho aos meus pais, Edgar Souza Lima e Martinha Silva Lima, pelo apoio, incentivo, confiança e amor incondicional que sempre me deram e aos meus irmãos Rafael Willian Silva Lima e Edgar Souza Lima Filho, por valiosos ensinamentos, conselhos e experiências compartilhadas e o grande incentivo e confiança que me passaram.

Agradecimentos

Agradeço ao meu orientador Prof. Rui Pedro Lopes, por toda paciência, dedicação, apoio e todo o tempo disponibilizado para mentoria deste trabalho.

Agradeço à minha coorientadora Prof. Arlete Teresinha Beuren, por todo auxílio, disponibilidade, apontamentos e sugestões prestadas.

Agradeço também a minha família, que sempre foi o meu porto seguro, minha base de referência e meu centro de motivação. Em especial, aos meus pais, Edgar e Martinha, por todo carinho, incentivo, compreensão e amor que por mim tiveram e têm, e aos meus irmãos, Rafael e Edgar Filho, por toda motivação e ensinamentos.

Por fim, mas não menos importante, agradeço a todos os professores e amigos que fizeram parte desta longa jornada de estudos. Em especial, ao meu melhor amigo e grande parceiro de estudos Willian de Oliveira Silva.

Muito obrigado! A todas as pessoas que de alguma forma contribuíram à minha formação.

Resumo

A psoríase é uma lesão dermatológica que se manifesta em diversas regiões do corpo. Seu diagnóstico tardio pode gerar agravamento da doença em si, bem como das comorbidades associadas a ela. Contudo, devido à grande diversidade de doenças dermatológicas existentes, este diagnóstico pode ser complexo.

Neste contexto, o trabalho proposto apresenta um sistema computacional para auxiliar o processo de diagnóstico da psoríase por meio de redes neurais convolucionais profundas e uma aplicação para dispositivos móveis.

Para classificação foi criado um dataset composto por 752 imagens de pele humana, utilizado como alimentação às 3 redes de classificação: MobileNetV2, Xception e InceptionResnetV2. Uma etapa de segmentação semântica através de 3 arquiteturas diferentes baseadas na topologia da rede U-net foi desenvolvida para avaliar a influência da segmentação na classificação.

Os resultados atingidos apontam um bom potencial do sistema para classificação de imagens de psoríase. Um conjunto de testes entre diferentes tipos de dataset e redes neurais foi aplicado e apontou que, apesar do modelo de segmentação proposto ter atingido mais de 97% de acurácia e uma pontuação DICE superior a 91%, proporcionando uma segmentação relativamente boa, o dataset original proporciona maior acurácia de classificação em relação ao dataset segmentado. Na classificação, o modelo MobileNetV2 apresentou uma acurácia maior que 97% com uma taxa de perda de aproximadamente 7%.

O aplicativo móvel desenvolvido é capaz de utilizar os modelos de classificação e segmentação de forma simples, auxiliando o utilizador final no processo de análise da lesão

dermatológica por meio de uma interface intuitiva.

Palavras-chave: Redes neurais, Classificação, Segmentação semântica, Aplicação móvel.

Abstract

Psoriasis is a dermatological lesion that manifests itself in several regions of the body. Its late diagnosis can worsen the disease itself, as well as the comorbidities associated with it, however, amid the great diversity of existing dermatological diseases, this diagnosis can be complex.

This work proposes a computational system to assist the psoriasis diagnosis process through deep convolutional neural networks and an application for mobile devices.

For classification, a dataset was created composed of 752 images of human skin, fed to 3 classification networks: MobileNetV2, Xception and InceptionResnetV2. A semantic segmentation step through 3 different architectures based on the topology of the U-net network was developed to assess the influence of segmentation on the classification.

The results present a good potential of the system for the classification of psoriasis images. A set of tests between different types of datasets and neural networks was applied and pointed out that, although the proposed segmentation model reached more than 97% accuracy and a DICE score greater than 91%, providing a relatively good segmentation, the original dataset provides greater classification accuracy compared to the segmented dataset. In the classification, the MobileNetV2 model presented an accuracy greater than 97% with a loss rate of approximately 7%.

The developed mobile application is able to use the classification and segmentation models in a simple way, assisting the end user in the process of analyzing the dermatological lesion through an intuitive interface.

Keywords: Neural networks, Classification, Segmentation, Mobile application.

Conteúdo

Lista de Tabelas

Lista de Figuras

Siglas

ANN Artificial Neural Network. 12, 13

API Application Programming Interface. 32, 67

ASPP Atrous Spatial Pyramid Pooling. 27

AUC Area Under The Curve. 35

BIC Border/Interior pixel Classification. 10

BN Batch Normalization. 57

CADx Computer-aided Diagnosis. 33

CLAHE Contrast Limited Adaptative Histogram Equalization. xvi, 65, 66

CNN Convolutional Neural Network. xv, 17, 18, 20

FCN Fully Convolutional Network. 22, 23, 54

FN Falso Negativo. 71

FP Falso Positivo. 71

HTTP Hypertext Transfer Protocol. 67, 68

IoT Internet of Things. 32

IPC International Psoriasis Council. 48

JSON JavaScript Object Notation. 67–69

LBP Local Binary Pattern. 11

ReLU Rectified Linear Units. 16, 19

RF Requisito Funcional. 37

RGB Red, Green, Blue. 10, 19, 49, 52, 56, 66

RNF Requisito Não Funcional. 37

SBD Sociedade Brasileira de Dermatologia. 3

SDK Software Development Kit. 68

SVM Support Vector Machine. 33

TF TensorFlow. 32

UML Unified Modeling Language. 2, 37

VN Verdadeiro Negativo. 71

VP Verdadeiro Positivo. 71

Capítulo 1

Introdução

A psoríase é uma doença inflamatória crônica da pele, que causa lesões avermelhadas geralmente em regiões de articulação do corpo [1]. Existem diversas variações da doença, que são caracterizadas desde pequenos pontos avermelhados em partes como cotovelos, joelhos e couro cabeludo, até tipos que causam lesões generalizadas em até todo o corpo [1]. Apesar da psoríase não ser uma doença contagiosa, ela causa um grande impacto negativo na autoestima do indivíduo, o que causa diversos problemas sociais, como dificuldades em relações pessoais e íntimas [2].

Tendo em conta a possível falta de informação acerca da doença, pode haver um período significativo desde os primeiros sintomas da psoríase até à consulta do paciente com um profissional especializado, para obter o diagnóstico.

Os procedimentos computacionais de reconhecimento de imagens estão cada vez mais robustos. Uma alternativa para auxiliar na redução do tempo deste diagnóstico é o uso da computação. Existem diversas aplicações computacionais em inteligência artificial auxiliando diretamente no diagnóstico de doenças realizado por profissionais da área da saúde. Desta forma, a utilização de um sistema que utiliza inteligência artificial para auxiliar o diagnóstico prévio da doença psoríase à população é de grande relevância.

1.1 Objetivos

O objetivo deste trabalho é propor uma abordagem para o diagnóstico da psoríase através de dispositivos móveis usando redes neurais convolucionais profundas.

1.2 Estrutura do Documento

O conteúdo deste projeto está organizado da seguinte forma: no capítulo 2 são apresentadas todas referências teóricas necessárias para entendimento do escopo geral do projeto, iniciando por informações sobre a psoríase, seguindo para visão computacional e Machine Learning e terminando com o estado da arte; no capítulo 3 é apresentado o planejamento e a documentação necessária ao projeto, oque inclui requisitos do sistemas, diagramas Unified Modeling Language (UML) e fluxogramas; no capítulo 4 são apresentados de forma sequencial todos os módulos desenvolvidos, iniciando pelas bases de dados, passando pelo aumento de dados, segmentação, classificação e finalizando no desenvolvimento da aplicação móvel e do servidor web; no capítulo 5 são apresentadas as metodologias e métricas utilizadas para avaliação do sistema, bem como discussões e interpretações acerca dos resultados obtidos; por fim, no capítulo 6 é apresentado uma descrição sintetizada de todo o trabalho, decorrendo sobre trabalhos futuros e finalizando com uma conclusão dos resultados e objetivos atingidos.

Capítulo 2

Contexto e Tecnologias

2.1 Psoríase

Segundo a Sociedade Brasileira de Dermatologia (SBD), a psoríase é uma doença inflamatória crônica de pele, que geralmente se apresenta nas articulações do corpo. Sua ocorrência é universal e frequente na prática clínica, afetando homens e mulheres do mesmo modo. Fatores ambientais, geográficos e étnicos podem induzir a incidência da psoríase. A doença pode ocorrer em qualquer idade, tendo maior probabilidade na segunda e na quinta década de vida. Manifestando-se geralmente a partir de placas eritemato escamosas, ocasionalmente pruriginosas, nas regiões de constantes traumas na pele, como joelhos, região pré-tibial, couro cabeludo e região sacra [1]. A Figura 2.1 apresenta um exemplo das lesões causadas pela doença psoríase.

De acordo com o local da lesão e as suas características clínicas, a psoríase pode ser caracterizada por diferentes tipos [4]. São eles: psoríase vulgar ou em placas, caracterizada por manchas em formato oval, eritematosas e com limites bem definidos, coberta por escamas de cor branca-acinzentada e espessas; psoríase gutata, manifesta-se como pequenas lesões (1 a 10mm) com distribuição geralmente no tronco, mais frequente em crianças e adolescentes; psoríase inversa, caracteriza-se por grandes lesões eritematosas,

Figura 2.1: Lesão de psoríase [3].

brilhantes e com nenhuma ou pouca descamação nas pregas cutâneas; psoríase eritrodérmica, refere-se a cobertura de total ou quase total da pele por psoríase, com descamação e eritema generalizados; psoríase pustulosa, definida por múltiplas pústulas assépticas em base eritematosa; psoríase ungueal, pode ocorrer de forma isolada, porém frequentemente se dá juntamente com a psoríase vulgar e, em alguns casos, com a psoríase artropática. As alterações mais frequentes são micro-depressões de 0,5 a 2,0mm na tábua externa da unha; psoríase antropática, caracteriza-se como uma artrite inflamatória, suas características clinicas distintivas incluem entesite, tenosinovite, artrite das articulações interfalângicas distais, dactilite ("dedos de salsicha"), osteíte, peri-osteíte, oligo-artrite assimétrica e espondilite [5].

Embora a origem da psoríase seja desconhecida, as lesões são caracterizadas por crescimento rápido da epiderme associado com o aumento na velocidade de troca das células da epiderme. Em condições normais, as células naturais da pele são substituídas em ciclos de cerca de 28 dias. Na psoríase, células epidérmicas são substituídas em apenas quatro dias. Alguns fatores podem atuar como agravantes na psoríase, incluindo infecções, clima frio e alguns medicamentos como betabloqueadores e lítio. Além destes, estresse, obesidade e alcoolismo também implicam na exacerbação do quadro de psoríase [6].

A psoríase pode ter consequências que acabam por serem maiores que a própria doença, é algo que traz um impacto enorme para as vidas destes doentes, diminuindo a sua qualidade de vida, prejudicando assim, o seu desenvolvimento pessoal [2]. Para pacientes com psoríase, a exposição da pele pode gerar impactos de socialização, relações pessoais e relações íntimas, bem como influenciar na estabilidade mental e emocional do indivíduo [7]. De acordo com [3], diversas comorbidades como hipertensão arterial, obesidade, infarto agudo miocárdio e acidente vascular cerebral podem estar associados à psoríase.

O diagnóstico da psoríase em geral é dado pelo quadro clínico e histórico do paciente, mas não comumente pode se dar por meio do exame histopatológico, bem como, por outros exames do couro cabeludo e das unhas [1]. Não há necessidade de exames auxiliares para o diagnóstico da psoríase, mas exames como a biópsia cutânea, para estudos histopatológico, podem auxiliar no diagnóstico [8].

2.2 Visão Computacional

A visão computacional visa utilizar computadores para emular a visão humana, incluindo o aprendizado, a capacidade de fazer inferências e agir com base em informações visuais. Esta área representa um ramo da inteligência artificial cujo objetivo é simular a inteligência humana agregando flexibilidade, uma vez que a resposta se altera de acordo com as variações do ambiente onde ela é utilizada [9], [10]. Desta forma, a partir de uma imagem de entrada, a visão computacional gera como saída uma interpretação total ou parcial desta imagem [11].

Geralmente sistemas de visão computacional possuem arquitetura em etapas sequenciais, como pode ser observado na Figura 2.2, ordenadas da seguinte forma: aquisição, pré-processamento, segmentação, extração de características e, por fim, reconhecimento e interpretação [12], [13] .

Para simplificar este processo é possível estabelecer dois níveis de abstração: processamento de imagens (baixo nível) e análise de imagens (alto nível). De forma que, os métodos de baixo nível geralmente utilizam pouco conhecimento sobre o conteúdo das

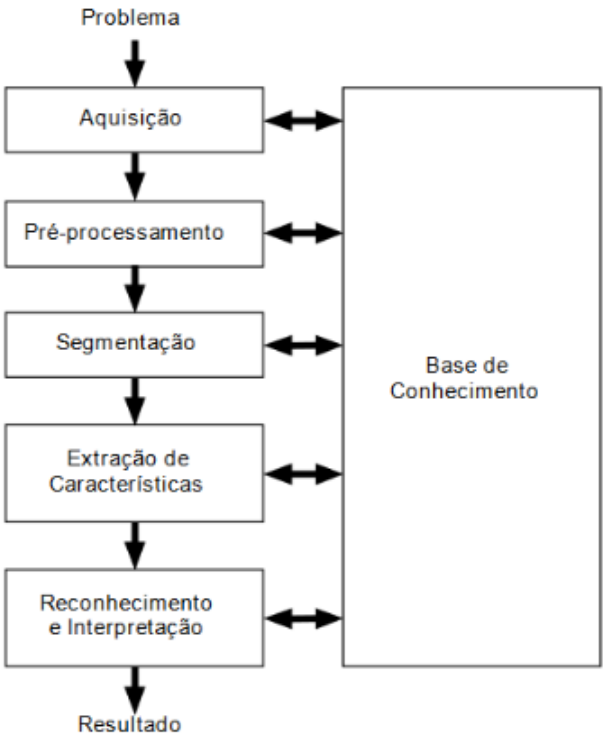

Figura 2.2: Representação de um sistema de visão computacional [12].

imagens. A redução de ruído, o aumento de contraste, a extração de bordas e a compressão de imagem são exemplos de operações realizadas neste método. Por outro lado, os métodos de alto nível envolvem tarefas como segmentação da imagem em regiões ou objetos de interesse, descrição desses objetos de modo a reduzi-los a uma forma mais apropriada para representar o conteúdo da imagem e reconhecimento ou classificação destes objetos [14].

2.2.1 Aquisição

A etapa de aquisição de imagens é o processo em que algum dispositivo de aquisição capta a imagem do mundo real e a representa a partir da transformação da informação de intensidade luminosa em sinal elétrico. Dependendo do modelo deste transdutor empregado, o sinal representado pode ser digital ou analógico. Em casos de saída analógica, este sinal de saída deve ser convertido por um conversor analógico/digital externo [10]. Segundo

[15], a saída gerada por este conversor é caracterizada como um conjunto de dígitos 0 e 1, gerando uma imagem digital, que pode ser bidimensional, tridimensional ou uma sequência de imagens.

Uma imagem pode ser contínua em relação à sua amplitude bem como às suas coordenadas x e y. Desta forma, para realizar a conversão de imagens representadas por dados contínuos em imagens digitais são utilizados os processos de amostragem e quantização. A digitalização dos valores de amplitude é denominada quantização e a digitalização dos valores de coordenada pode ser intitulada de amostragem [9].

A Figura 2.3(a) apresenta uma imagem contínua f que será submetida ao processo de amostragem e quantização. Estes métodos são realizados linha a linha, desta forma, a linha representada pela reta AB na Figura 2.3(a) é selecionada para exemplificar o processo de conversão. A Figura 2.3(b) representa os valores de amplitude da reta AB selecionada na imagem contínua. O processo de amostragem ocorre a partir da obtenção de amostras igualmente espaçadas, extraídas ao longo do segmento da reta AB. Como ilustra a Figura 2.3(c), as amostras são representadas por quadrados brancos, posicionados de acordo com as marcações verticais na parte inferior da imagem. Para o processo de quantização cada amostra é associada ao valor mais próximo verticalmente da tabela de intensidade localizada à direita da Figura 2.3(c). A tabela de intensidade é representada por oito intervalos de cores que variam do branco ao preto associados às pequenas marcações verticais. As amostras digitais produzidas pelos processos de amostragem e quantização são apresentadas na Figura 2.3(d). Para obtenção da imagem digital, é necessária a aplicação deste processo em toda a imagem, linha a linha [9].

Segundo [14], comumente ocorrem ruídos nesta etapa, ou seja, degradações na imagem durante o processo de aquisição.

2.2.2 Pré-processamento

Nesta etapa, o objetivo é aperfeiçoar a imagem de forma que auxilie na qualidade e sucesso das fases subsequentes. Geralmente o pré-processamento utiliza métodos para remover

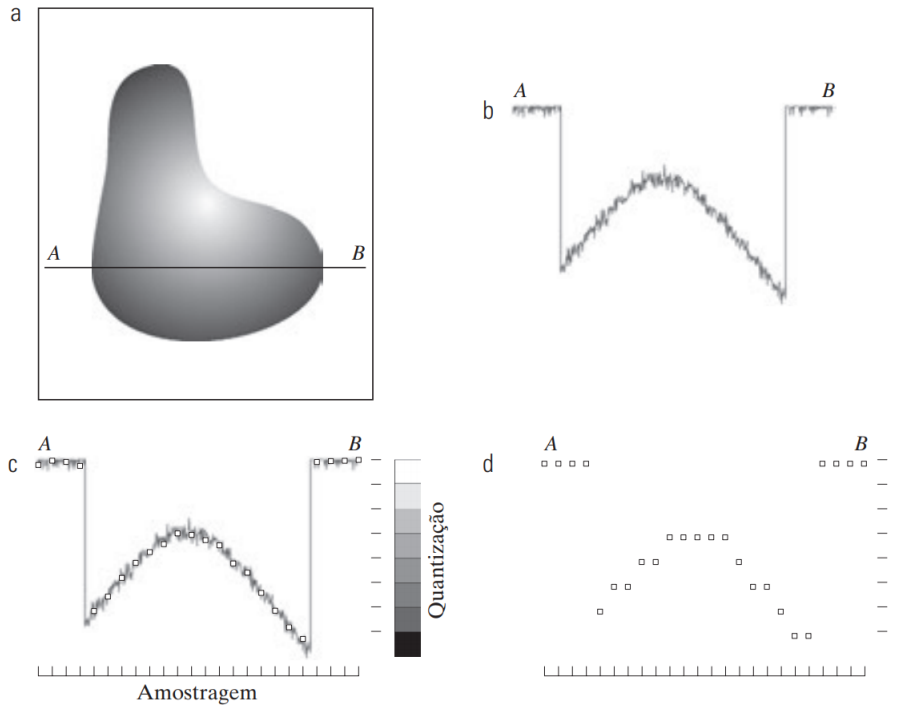

Figura 2.3: Processo de amostragem e quantização [9].

ruídos, realçar contrastes e isolar áreas cuja textura indique a probabilidade de informação alfanumérica [9]. De acordo com [16], o pré-processamento exerce principalmente as funções de realce e restauração das imagens que serão processadas.

O realce tem como objetivo destacar características, facilitando a extração e análise da informação desejada na imagem. Este método é relativamente subjetivo, pois a sua efetividade depende fortemente da informação que o observador deseja extrair, e se a informação existe realmente na imagem em questão. Em casos em que a imagem se encontra imperfeita ou corrompida, aplica-se a técnica de restauração, objetivando a recuperação da imagem original ou a aproximação de sua qualidade por meio do conhecimento dos processos físicos que levaram a sua formação [17].

2.2.3 Segmentação

O processo de segmentação tem o objetivo dividir uma imagem em regiões ou objetos distintos. Esta etapa é conduzida pelas características do objeto ou região em questão, como, por exemplo, proximidade ou coloração. A resolução da imagem e a tarefa a ser executada determinam o nível de detalhamento da segmentação [11]. Segundo [18], a separação entre os objetos de interesse e o restante da imagem, é realizada para extrair informações que irão auxiliar na identificação dos objetos desejados.

Não existe um método padrão ou único que tenha a capacidade de segmentar diversos tipos de imagem, mesmo havendo múltiplas técnicas de segmentação. Desta forma, a escolha da técnica para esta etapa deve ser selecionada de acordo com as características da imagem, dos procedimentos de pós-segmentação e das propriedades que serão avaliadas em seguida [19].

A segmentação a partir de limiares de intensidade de cor é uma das formas mais comuns de segmentação de imagens. Limiares de intensidade são valores definidos onde a imagem é separada em regiões por meio do processo comparativo dos valores absolutos de intensidade de cada pixel com estes limiares, este processo permite que a imagem seja fragmentada em duas, se tiver apenas um limiar, ou diversas regiões [20]. Esta segmentação ocorre, pois valores de pixeis que excedam o valor do limiar são agrupados em uma região e pixeis que tenham valores inferiores ao limiar são agrupados na região adjacente [17]. A Figura 2.4 exemplifica este processo.

Nesta aplicação pode-se observar que as moedas contidas na imagem foram segmentadas corretamente. Porém, neste caso, o limite atribuído foi determinado de maneira manual por meio de tentativa e erro. Existem casos em que tal atribuição de limiar por um humano é eficaz, contudo, diversas pesquisas de processamento de imagens necessitam de uma automação geral, ou seja, a seleção de um limiar deve ocorrer de forma automática [17].

De acordo com [9], a precisão da segmentação define o sucesso ou falha dos métodos

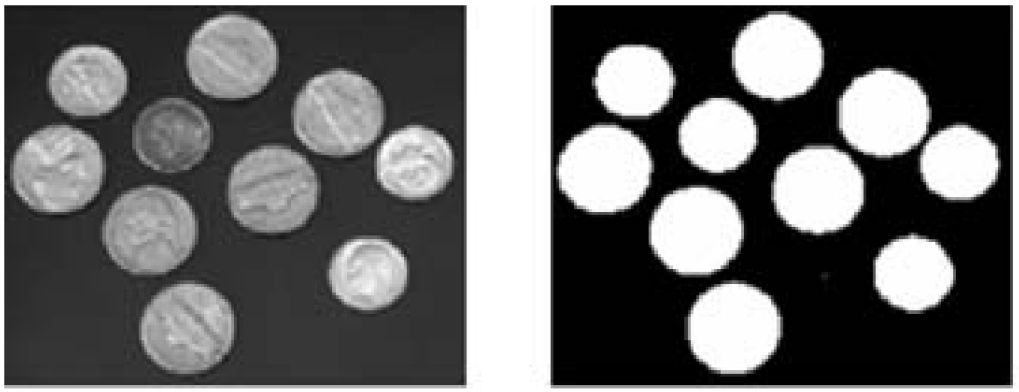

Figura 2.4: Aplicação da técnica de segmentação por limiares de intensidade [17].

de análise computadorizada dos dados. O fato de haver uma necessidade da maior precisão torna a segmentação de imagens um dos procedimentos mais complexos da área de processamento de imagens.

2.2.4 Extração de Características

Para que as propriedades de interesse sejam destacadas, faz-se necessário o uso de um método para as selecionar. Esta etapa é denominada extração de características e tem como objetivo extrair propriedades das imagens que são fundamentais para discriminação entre objetos, ou essenciais para obter informações quantitativas de interesse [9].

Uma região pode ser representada por meio de sua característica interna, que se refere a pixeis que fazem parte do objeto, como cor e textura, bem como parte de sua área externa, que diz respeito a borda do objeto [9], [17].

Para extrair características de imagens a partir de sua cor, é possível utilizar o extrator Border/Interior pixel Classification (BIC), baseado no espaço de cores Red, Green, Blue (RGB). Este método utiliza dois histogramas para subdividir os pixeis entre as categorias interior e borda. Um pixel é atribuído à categoria de interior se os quatro pixeis adjacentes a ele tiverem a mesma cor, caso contrário, este pixel é atribuído a categoria borda. No final deste processo, os dois histogramas concatenados formam um vetor de características

[21], [22].

Em casos em que se deseja extrair características de textura, pode-se optar por métodos como o Local Binary Pattern (LBP), que utiliza uma operação matemática sobre cada pixel, gerando uma nova imagem de padrões binários locais. A partir dos padrões extraídos da imagem gerada é produzido o histograma, que frequentemente é utilizado como vetor de características [23]. Segundo [24], o LBP contém resultados relativamente eficientes, invariante à rotação e intensidade de escala de cinza.

Para a representação da forma de um objeto, dentre os diversos métodos, o processo denominado esqueleto é uma maneira de reduzir a forma em questão a uma estrutura mais simples. Este processo pode ser realizado por meio de procedimentos como a transformada de distância, que gera um mapa da distância entre cada ponto do interior do objeto com a sua margem mais próxima. Os pontos com valores mais altos representam o esqueleto do objeto [14].

2.2.5 Reconhecimento e Interpretação

Nesta etapa, o procedimento de reconhecimento tem o objetivo atribuir identificadores a um objeto, de acordo com as propriedades fornecidas pelos seus descritores. No processo de interpretação busca-se a atribuição de significado a um conjunto de objetos identificados anteriormente [25].

Pode-se definir um padrão como um arranjo de características. Desta forma, é possível compreender uma classe de padrões como uma família de padrões que compartilham algumas propriedades equivalentes. As classes de padrões comumente são representadas como $w1$, $w2$, ..., ww sendo que w é a quantidade de classes. O processo de reconhecimento de padrões por máquina envolve procedimentos de atribuição de padrões às suas respectivas classes de maneira automática e com menor intervenção humana possível [9]. Segundo [14], quando se atribui um mesmo rótulo a amostras distintas, compreende-se que tais elementos pertencem a uma mesma classe, que contém elementos com propriedades em comum. Para atribuir corretamente cada amostra a sua respectiva classe $w1$,

$w2$, ..., ww são utilizados os classificadores, que podem ser definidos como algoritmos que visam mapear as características das amostras e o conjunto de rótulos. Os processos de classificação podem ser supervisionados ou não supervisionados.

Nas técnicas supervisionadas são consideradas classes previamente definidas por meio da etapa de treinamento. Esta etapa é executada anteriormente à aplicação do algoritmo de classificação, para que os parâmetros que caracterizam cada classe sejam obtidos. Utilizando vetores de características já alocados a uma classe definida como dado de treinamento, é possível projetar um sistema de classificação de características generalizado, em que novos exemplos de vetores de características que não foram utilizados no projeto sejam classificados com precisão [14], [17].

Uma vez que não necessite de parâmetros ou conhecimentos adquiridos anteriormente à aplicação do algoritmo de classificação, o processo é denominado como não supervisionado, ou seja, as informações de interesse devem ser adquiridas por meio das próprias amostras a serem rotuladas [14]. De acordo com [17], exemplos de classificação não supervisionada são as técnicas de aglomeração (clustering).

As técnicas de reconhecimento de padrões podem ser divididas em duas classes: decisão teórica, que engloba os padrões de descritores quantitativos, e a decisão estrutural, que abrange os padrões de descritores qualitativos. Ambas são utilizadas para extrair informações semânticas de acordo com determinado critério [18].

2.3 Redes Neurais Artificiais

Segundo [26], uma Artificial Neural Network (ANN) pode ser definida como um processador paralelamente distribuído composto por unidades de processamento simples, propensos naturalmente para armazenar conhecimento por meio de experimentos e disponibilizá-lo para o uso.

Para Braga, Carvalho e Ludermir [27], uma ANN pode ser definida como um sistema paralelo distribuído, constituído por elementos de processamento simples (nodos) que calculam determinadas funções matemáticas. A organização desses elementos é feita por

meio de uma ou mais camadas, e a relação entre eles é realizada através de uma ampla quantidade de conexões, comumente unidirecionais. Geralmente estas conexões estão relacionadas a pesos, que armazenam o conhecimento retratado no modelo e servem para ponderar a entrada obtida por cada neurônio da rede. Tal funcionamento de uma ANN é inspirado no cérebro humano [27].

O cérebro humano possui cerca de 10^{11} neurônios, que se comunicam entre si continuamente em paralelo. O cérebro humano é responsável por funções como pensamento, emoção, cognição e funções sensório motoras. Possui ainda a capacidade de realizar o reconhecimento de padrões, utilizar e armazenar o conhecimento por experiência [27]. Segundo Barros [28], a partir de sua ampla capacidade de absorção de conhecimento exerce funções como intuição, inferência, controle motor, reconhecimento de padrões e percepção

A base de estudos das Artificial Neural Networks (ANNs) é composta pela estrutura singular dos nodos de uma rede biológica, a topologia de suas conexões e o comportamento composto que estes nodos exercem. Desta forma, apesar das diferenças do ponto de vista físico entre redes artificiais e redes biológicas, as ANNs têm como objetivo reproduzir a dinâmica das redes biológicas e seus comportamentos básicos [27]. Segundo Haykin [26], uma ANN possui duas características semelhantes ao cérebro humano: a primeira refere-se ao conhecimento que é adquirido pela rede através de um método de aprendizagem; e, a segunda diz respeito aos pesos sinápticos, que são forças de conexão entre neurônios, empregados para armazenar o conhecimento adquirido.

2.3.1 Aprendizado Profundo

Para um comportamento inteligente, a aptidão de aprendizado é considerada fundamental. Atividades como observar, memorizar e analisar situações para compreender fatos, aprimorar habilidades motoras/cognitivas mediante práticas e organizar novo conhecimento em representações podem ser consideradas atividades relacionadas ao aprendizado [29].

O campo da aprendizagem de máquina tem como objetivo construir programas de

computador com capacidade de adquirir conhecimentos novos, bem como métodos para organizar os conhecimentos já compreendidos, aprimorando automaticamente seu desempenho com a experiência [30]. Segundo Faber [31], técnicas de aprendizado de máquina buscam prever futuros resultados por meio de informações de saídas passadas.

Uma classe de técnicas de aprendizado de máquina que utilizam diversas camadas de processamento de informação não lineares é denominada de aprendizado profundo. A aprendizagem profunda é composta pela interseção entre as áreas de pesquisa de inteligência artificial, modelagem gráfica, reconhecimento de padrões, processamento de sinais e redes neurais [32]. De acordo com Pacheco [33], aprendizado profundo se baseia nas técnicas de aprendizado de máquina e redes neurais artificiais, e comumente é aplicado em áreas de reconhecimento de imagens, áudio, caracteres e detecção facial.

Métodos que utilizam aprendizado profundo tem como objetivo encontrar um modelo por meio de uma coleção de dados utilizada como exemplo e um procedimento para orientação do aprendizado através desses exemplos. Após o procedimento de aprendizagem obtém-se uma função que a partir de uma entrada de dados brutos é capaz de conceder como saída uma representação pertinente ao problema tratado [34].

Arquitetura básica

Arquiteturas de aprendizado profundo adicionam "n" (quantidade de camadas ocultas) camadas entre a camada de entrada e a de saída. Estas camadas são denominadas de camadas ocultas, e são diretamente conectadas entre si [33], [35]. Segundo Ponti e Costa [34], métodos que utilizam aprendizado profundo aprendem mediante composições de funções (Equação 2.1):

$$(f_L(f_2(f_1(x_1, W_1); W_2)), W_L) \tag{2.1}$$

em que,

f = Função de ativação;

W = Coleção de parâmetros;

x = Vetor de dados de entrada;

L = Número de camadas.

no qual cada função f recebe como entrada um vetor de dados xl e sua própria coleção de parâmetros Wl, fornecendo como saída o próximo vetor xl+1 que será utilizado como entrada à seguinte função, geralmente o índice l destas funções se refere a uma camada. Desta forma, a equação acima representa um conjunto de L camadas, que permite obter o resultado desejado a partir do aprendizado em uma sequência de funções que modificam vetores mapeando-os de um espaço a outro. A Figura 2.5 exemplifica esta estrutura apresentando uma arquitetura profunda com duas camadas ocultas [34].

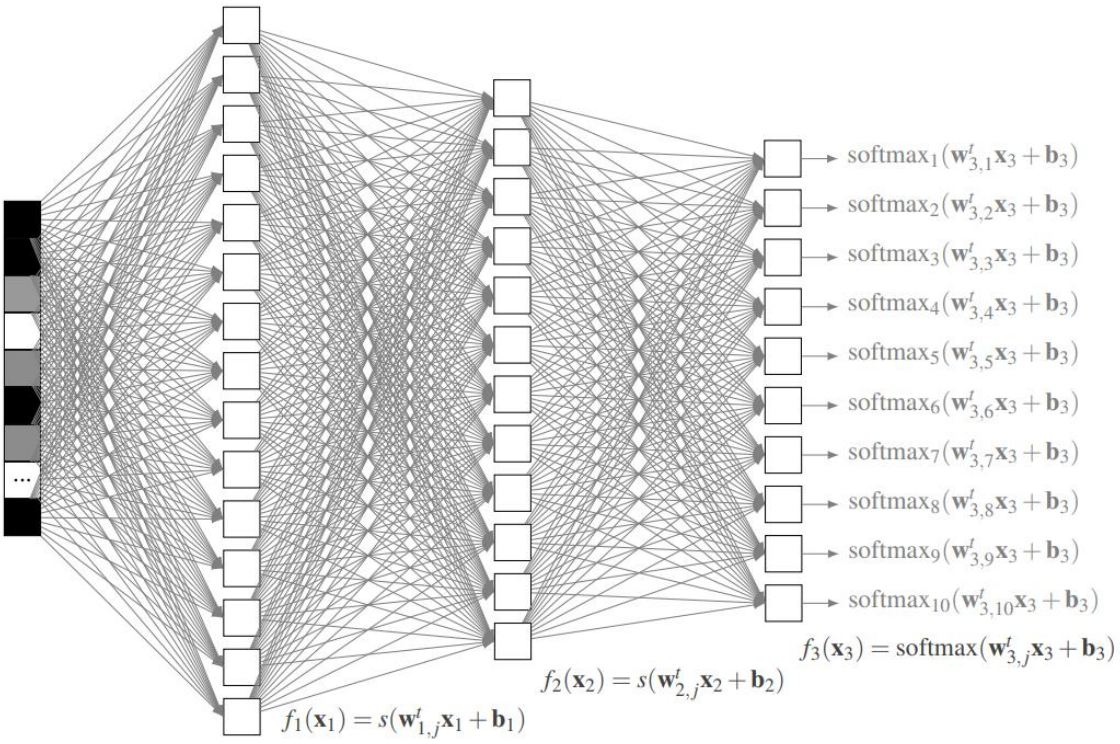

Figura 2.5: Arquitetura profunda com duas camadas ocultas [34].

Tal estrutura pode ser definida pela Equação 2.2, sendo b o vetor de valores de ajustes bias:

$$\hat{y} = f(x) = (f_3(f_2(f_1(x_1; W_1; b_1); W_2; b_2); W_3; b_3), \tag{2.2}$$

em que,

f = Função de ativação;

W = Coleção de parâmetros;

x = Vetor de dados de entrada;

b = Vetor de valores de ajustes bias.

no qual, $f_1(x_1) = x_2$, $f_2(x_2) = x_3$ e por fim $f_3(x_3) = y$. A função SoftMax é considerada uma função de ativação para classificação utilizada comumente em camadas de saída, por outro lado camadas ocultas geralmente utilizam a Rectified Linear Units (ReLU) como função de ativação [34]. Segundo Block [36], a função ReLU (Equação 2.3) gera como saída valores entre 0 e o máximo fornecido pela função:

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

em que,

x = Valor máximo fornecido pela função;

Funções de ativação tem como objetivo restringir os valores fornecidos pelo neurônio auxiliando na classificação dos sinais transmitidos das redes neurais. O desempenho destas funções é calculado por meio da capacidade da rede em classificar de maneira correta os dados em sua etapa final [36].

Aprendizado supervisionado

Neste modo de aprendizado o processo de aprendizagem conta com um "professor" que "ensina" a rede, fornecendo informações para levá-la a saída desejada. Desta forma, com base em um conjunto de pares entrada-saída, em que a entrada é composta pela informação de como deveria ser a saída, o modelo de aprendizado é treinado. A medida

de erro entre a saída obtida e a saída desejada permite que o supervisor realize ajustes nos parâmetros, visando a minimização deste erro a cada iteração e a obtenção de uma estrutura que encontre as saídas mais próximas dos valores esperados [37], [38].

Para Russell e Norvig [39], aprendizagem supervisionada busca por meio de um conjunto de treinamento de N pares de exemplos de entrada e saída (Equação 2.4):

$$(x_1, y_1), (x_2, y_2), ...(x_n, y_n), \tag{2.4}$$

em que,

x = Exemplos de entrada;

y = Exemplos de saída.

no qual cada yj foi constituído por meio de uma função desconhecida $f(x) = y$, encontrar uma função h (hipótese de saída desejada) que tenha uma saída próxima a saída gerada pela função verdadeira f. A função h é uma hipótese e os valores x e y não são necessariamente números, ou seja, podem ter qualquer valor. Desta forma, dado um espaço de hipóteses possíveis, a aprendizagem é uma busca pela hipótese que terá uma boa eficiência, mesmo com novos exemplos. Portanto diz-se que uma hipótese tem um bom desempenho se a partir de um novo exemplo ela seja capaz de prever de maneira correta o valor de y [39].

De acordo com Sánchez-Agustino [35], o aprendizado supervisionado pode ser utilizado em classificação de imagens a partir de algoritmos classificadores, que geram um modelo por meio de um conjunto de imagens anteriormente rotuladas. Desta forma, o modelo desenvolvido é utilizado para classificar novas imagens.

2.3.2 Rede Neural Convolucional

Convolutional Neural Networks (CNNs) são redes neurais inspiradas no córtex visual. Este tipo de arquitetura faz parte dos métodos empregados na área do aprendizado profundo, e visa a aplicação no campo de visão computacional voltada para classificação de imagens

[40]. As CNNs podem ainda ser empregadas em outras tarefas como processamento de linguagem natural e classificação de textos [35]. De acordo com Ponti e Costa [34], a principal utilização das CNNs é para o processamento de imagens, uma vez que a convolução possibilita a filtragem das imagens considerando a sua estrutura bidimensional.

Oliveira [41], relata que CNNs empregam a operação de convolução a fim de obter um processamento mais rápido e extrair características em diversas áreas de uma imagem, desta forma diversas camadas de convolução aplicadas a uma mesma imagem possibilitam extrair diferentes características.

O funcionamento de uma CNN pode ser influenciado por meio dos seguintes parâmetros: número de filtros (kernel), tamanhos dos filtros, stride, que se refere ao passo dado ao longo da imagem de entrada, função de ativação e função de agrupamento, bem como a organização da estrutura de camadas da rede [36]. Segundo Teixeira [42], a estrutura de uma CNN é constituída por camadas de convolução, camadas de pooling e camadas totalmente conectadas. A Figura 2.6 ilustra a estrutura geral de uma CNN.

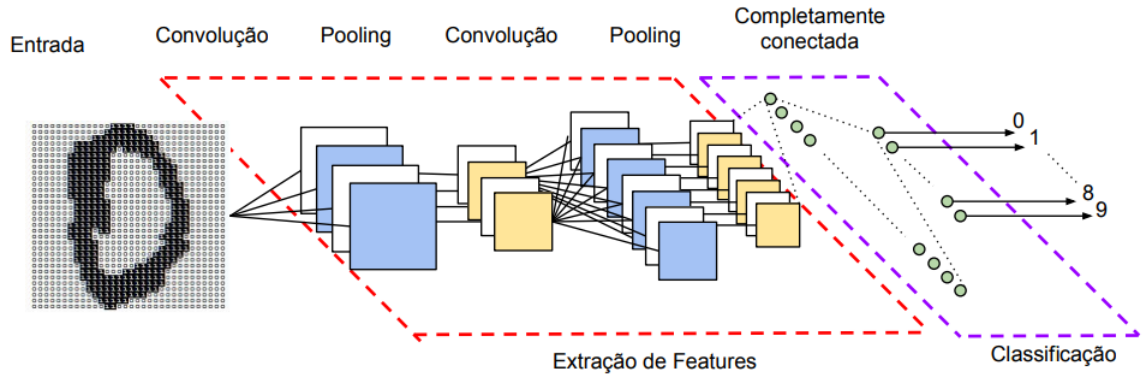

Figura 2.6: Estrutura geral de uma CNN [43].

Camada Convolucional

Camadas convolucionais são constituídas por "n" filtros, com pesos inicializados previamente, ajustados por meio de métodos como o backpropagation à medida que a rede aprende a reconhecer regiões de interesse. Comumente, os filtros são matrizes reduzidas,

por exemplo, matrizes 5x5 nos 3 canais (RGB) de cor, constituídas de valores reais que após a convolução com os dados de entrada geram um mapa de características, estes mapas apontam onde estão localizadas as características extraídas por meio dos filtros [40]. De acordo com Ponti e Costa [34], os filtros, ou tensores, de tamanho 5x5xd, 3x3xd e 1x1xd, onde d é a profundidade, são os mais utilizados.

Uma vez que o filtro é aplicado por toda a imagem, o tamanho do passo realizado pelo filtro a cada deslocamento é denominado stride e seu valor deve ser definido pelo projetista da rede. Desta maneira, caso o stride seja definido como 1, o filtro irá deslocar-se uma unidade por vez durante o processo de convolução [42].

O padding ou preenchimento de zero é utilizado para controle do tamanho da saída da convolução, desta maneira a saída terá o mesmo tamanho da entrada, por meio da adição de zeros em toda a margem da matriz de entrada [41], [44], [45].

Após a aplicação de operações de convolução existe a possibilidade de se gerar uma saída linear, desta forma, funções como a ReLU conectada à camada de convolução evitam esta ocorrência de saídas lineares por meio da adição de uma não linearidade à camada convolucional [40].

Camada de Pooling

Geralmente, durante as operações de convolução, um grande volume de dados é manipulado, desta forma, visando a redução da quantidade deste volume para melhor desempenho de processamento utilizam-se as camadas de pooling [36].

Por meio da combinação de pixel adjacentes de uma região de agrupamento composta por um filtro, um stride e pelo tipo de agrupamento, a camada de pooling realiza a redução da dimensão das características aprendidas [41]. Existem duas funções de agrupamento comumente utilizadas: a primeira MaxPooling utiliza os maiores valores de cada região, e a segunda AveragePooling realiza o agrupamento através da média dos valores da região [36], [45]. A Figura 2.7 apresenta um exemplo das operações de agrupamento realizadas pela camada de pooling.

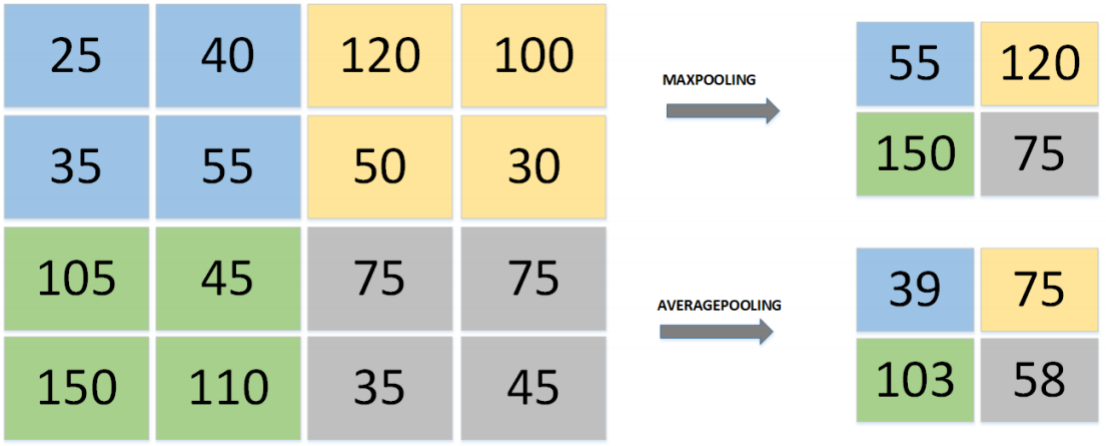

Figura 2.7: Exemplo do processo realizado pela camada de pooling [36].

De acordo com Silva [46], após esta operação obtém-se uma representação aproximadamente invariante a pequenas translações na entrada. Desta maneira, uma vez que a prioridade é classificar se uma entrada apresenta ou não uma determinada propriedade, independentemente de sua localização, a invariância à translação torna-se importante.

Camada Totalmente Conectada

Após as operações das camadas anteriores, é utilizada a operação Flatten para converter os mapas de características gerados em vetores, para que possam servir como entrada à camada totalmente conectada [42]. Desta maneira, se um tensor 4x4x40 for gerado por uma camada convolucional anteriormente a camada totalmente conectada, estes dados serão redimensionados e passaram a possuir o tamanho 1 x (4x4x40) = 1 x 640. Consequentemente, todo neurônio da camada totalmente conectada terá 640 pesos, gerando uma combinação linear do vetor [34].

Segundo Teixeira [42], o processo de extração de padrões das imagens é realizado pelas camadas anteriores. Uma vez que as características tenham sido extraídas das imagens, a função da última camada de uma CNN é classificá-las de acordo com a sua devida classe.

Block [36] pontua que, o objetivo das camadas totalmente conectadas é unir os dados obtidos das camadas seguintes, utilizando comumente a função SoftMax (Equação 2.5), para realizar a classificação.

$$\epsilon(z)_j = \frac{e^{zj}}{\sum_{k=1}^{k} e^{zk}}, \tag{2.5}$$

em que,

z = Vetor de valores;

k = Tamanho do vetor de valores;

j = Número de iteração.

Sendo z um vetor de valores e k o seu tamanho, esta função atribui a cada coleção de dados uma probabilidade para cada uma das classes de saída da rede. Desta maneira, valores decimais entre 0 e 1 são associados a cada classe, sendo 1 o resultado da soma de todos esses valores [36].

2.4 Segmentação Semântica

A segmentação semântica prevê etiquetas densas para todos os pixeis contidos em uma imagem, tal tarefa é considerada muito importante pois fornece um auxílio a compreensão profunda de objetos, cenas e do ser humano [47]. Segmentação semântica é um dos métodos de alto nível que facilita o entendimento completo da cena. A importância do entendimento de uma cena torna-se um grande problema da visão computacional. A partir do momento em que um número crescente de aplicações como condução autônoma e motores de pesquisa de imagem se alimentam de inferências de conhecimento sobre imagens [48].

A rotulagem por pixel pode ser definida da seguinte forma: procure um meio de atribuir um estado do espaço de etiquetas $L = l_1, l_2, ..., l_k$ para cada elemento contido em uma coleção de valores aleatórios $X = x_1, x_2, ..., x_n$. Cada um dos rótulos l corresponde a um objeto ou classe distinto, por exemplo, sinal de trânsito, carro ou avião. Comumente, o espaço de etiquetas possui k+1 estados possíveis, pois l_0 é utilizado para representação do *background*. Geralmente, X é uma imagem em duas dimensões W (width) x H

(height), contudo, este conjunto de variáveis pode assumir qualquer dimensionalidade, como imagens hiperespectrais, ou dados volumétricos [48]. A Figura 2.8 apresenta um exemplo de segmentação semântica.

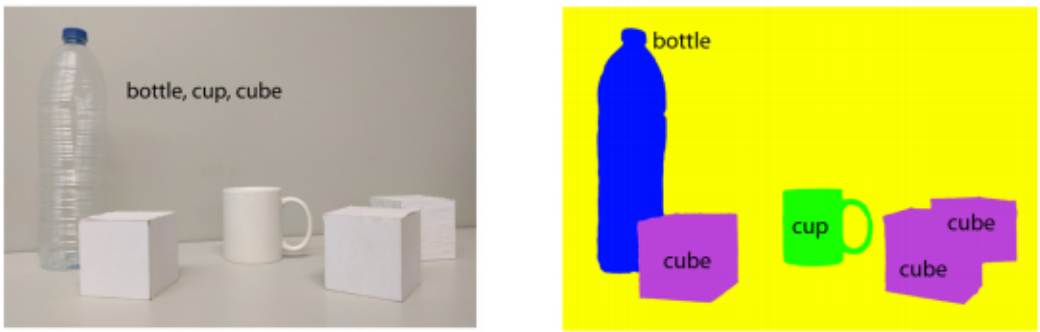

Figura 2.8: Aplicação de segmentação semântica a uma imagem. Adaptado de [48].

Atualmente algoritmos de segmentação semântica são frequentemente projetados com base em CNN para solucionar problemas de rotulação estruturada em pixeis [49]. A segmentação semântica baseada em CNN explora em maior parte as Fully Convolutional Networks (FCNs). O bom desempenho dessas redes geralmente está relacionado ao design eficaz do modelo quanto à largura e profundidade, no qual deve conter diversas operações e parâmetros [47]. A Fully Convolutional Network (FCN) recebe como entrada uma imagem de tamanho arbitrário que passa por diversas camadas convolucionais, produzindo uma probabilidade a cada pixel, por meio de um mapa de pontuação, para todas as categorias, fornecendo uma solução de ponta a ponta, ágil e precisa para segmentação semantica, graças a eficiencia, simplicidade e a propriedade de compartilhamento local de pesos das convoluções [50].

2.4.1 U-net

Uma grande preocupação na segmentação semântica é a relação entre localização e a semântica, no qual informações globais traduzem “o que”, que seria a semântica, e as informações locais representam “onde”, ou seja, a localização [51]. Arquiteturas baseadas

em FCN substituem camadas totalmente conectadas, que normalmente se localizam ao final de CNNs [51]. A Figura 2.9 apresenta um exemplo deste processo.

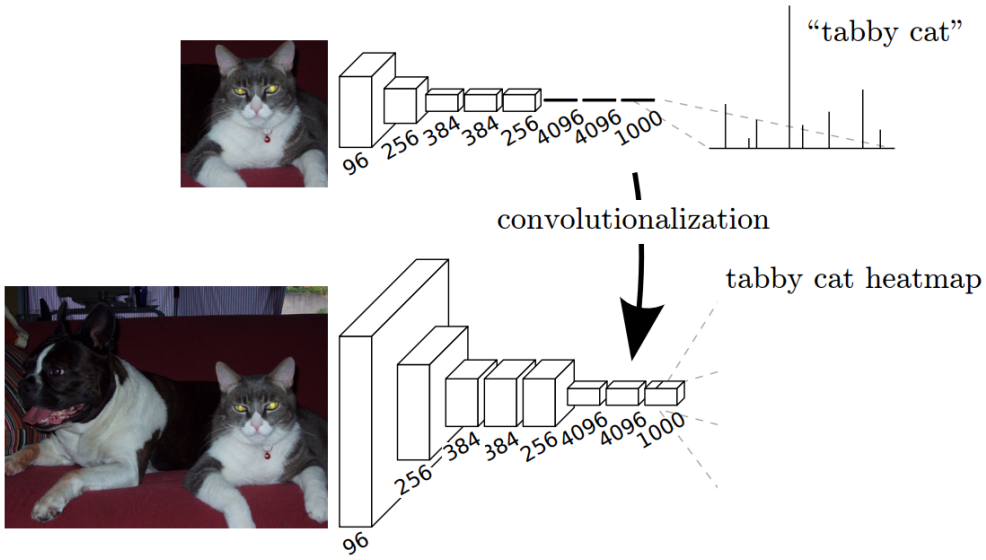

Figura 2.9: Comparação entre CNNs e FCNs [51].

De acordo com [52], a proposta da FCN é complementar à arquitetura padrão de contração da rede, por meio de sucessivas camadas de *upsample*, ao invés das camadas de pooling. Desta forma, tais camadas incrementam a resolução de saída, que são combinadas com camadas de alta resolução do caminho de contração para realizar a localização do conteúdo de interesse. Em seguida, uma camada convolucional subsequente pode aprender a gerar uma saída a partir destas informações extraídas.

Baseada na arquitetura FCN, a U-net possui modificações nas camadas de *upsampling*, que passam a dispor de uma grande quantidade de canais de recursos, permitindo a propagação de informações de contexto para camadas que possuem maior resolução. Desta maneira, o caminho de expansão é relativamente simétrico ao caminho de contração, gerando uma arquitetura em forma de "u". Tais modificações foram necessárias para que a U-net fosse capaz de produzir segmentações precisas a partir de bases de imagens pequenas [52]. A Figura 2.10 apresenta a arquitetura da rede U-net.

Esta arquitetura tem como resolução mais baixa a dimensão 32x32 pixeis. Cada caixa

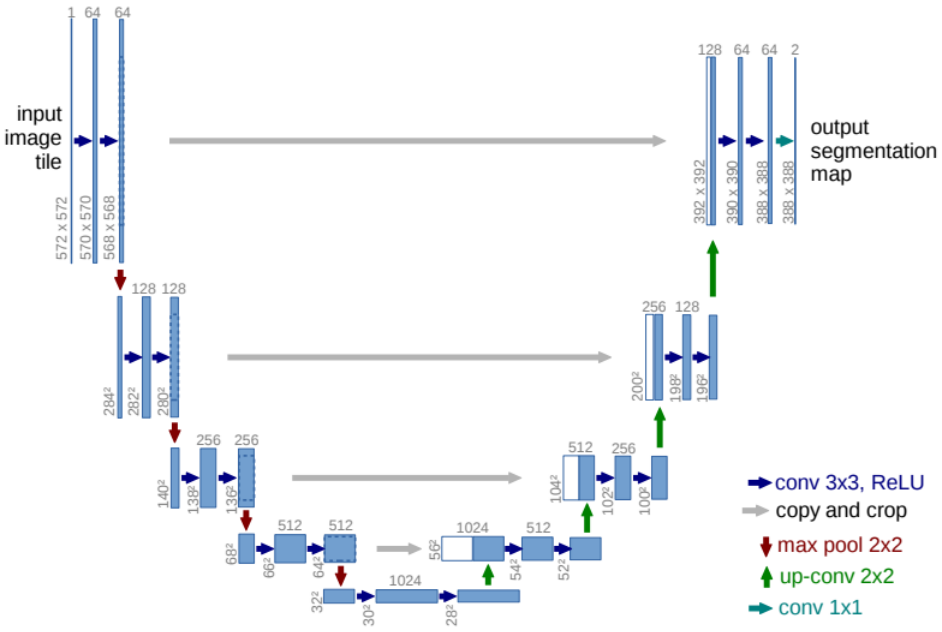

Figura 2.10: Arquitetura do modelo Unet [52].

azul representada na figura equivale a um mapa de recursos multicanais. Caixas em branco representam mapas de recursos copiados. O número acima das caixas representa o número de canais, a dimensão x-y está localizada na borda inferior esquerda de cada caixa [52].

A rede U-net possui um caminho de contração e um caminho de expansão. Ambos podem ser vistos na Figura 2.10, encontrando-se a contração na parte esquerda da figura e a expansão na zona direita. A contração é baseada na arquitetura típica de uma rede neural convolucional, portanto, efetua repetidas aplicações de duas convoluções *unpadded*, sendo cada uma delas seguida de uma ReLU e uma camada de pooling máximo de 2x2 com stride 2 para realização de *downsampling*. Após cada etapa de redução da amostragem, a quantidade de canais de recursos é dobrada. Por outro lado, no caminho expansivo cada etapa é constituida por um incremento do mapa de recursos seguido de uma convolução ascendente que reduz o número de canais de recursos pela metade, a saída produzida por esta convolução é concatenada com o mapa de recursos cortado correspondente do caminho de contração e duas convoluções 3x3 são aplicadas, sendo cada uma seguida por

uma ReLU. Na parte final da arquitetura, uma convolução 1x1 é utilizada para mapear cada um dos 64 componentes do vetor de características ao respectivo número de classes de saída. Em toda a arquitetura da rede U-net existem 23 camadas convolucionais [52].

De acordo com [53], a U-net trata-se de um codificar-decodificador que cada vez mais têm sido utilizado como base para diversas arquiteturas de aprendizado profundo com ênfase em análise de imagem biomédica. Existem evidências de que os resultado produzidos pela U-net são comparáveis em qualidade com anotações realizadas manualmente [53].

Uma característica que difere a U-net de diversas estratégias de anotação automatizada é a influência individual do anotador. Este recurso pode auxiliar o processo de aprendizado, pois existem casos em que diversos parâmetros de um determinado protocolo são considerados pelas pesquisas no momento da anotação, porém estas característica não são mencionadas explicitamente. Tais regras de rotulagem não são passíveis de reprodução por ferramentas de rotulagem automática comuns, devido à sua complexidade. Contudo, o modelo U-net aprende a partir de amostras fornecidas durante o seu treino e caso os exemplos não forem equivalentes a tarefa real ou as anotações manuais forem de baixa qualidade o treinamento da U-net não será eficaz e produzirá rotulações imprecisas em novos dados [53].

2.4.2 DeepLabv3

A arquitetura DeepLabv3 utiliza convolucões *atrous* em módulos *cascaded modules* e *spatial pyramid pooling*, desta forma, é possivel redirecionar redes pré-treinadas no ImageNet para extrair mapas de recursos densos, removendo as operações de redução da amostragem das últimas camadas e adicionando os filtros correspondentes [54]. Um exemplo da aplicação da convolução *atrous* é apresentado na Figura 2.11.

Na parte superior da ilustração, uma operação de *downsampling* de fator 2 é utilizada e em seguida uma convolução com um kernel é aplicada no mapa de recursos de baixa resolução, neste cenário, se o mapa de recursos resultante for ampliado para as coordenadas originais da imagem, apenas 1/4 da imagem original será obtida. Por outro

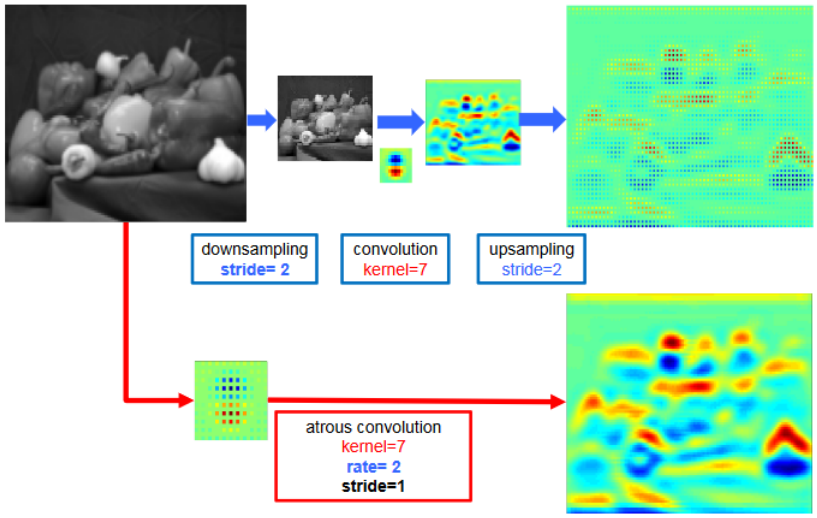

Figura 2.11: Aplicação de uma convolução *atrous* em um plano de duas dimensões [55].

lado, na parte inferior da ilustração uma convolução *atrous* é utilizada na imagem de alta resolução, pela qual o filtro é amostrado em um fator de 2 e zeros são adicionados entre os valores do filtro, permitindo assim que o filtro possa calcular saídas para todas as posições da imagem em resolução máxima [55]. Após a dilatação do filtro quantidade de parâmetros diferentes de zero é exatamente a mesma que o filtro original, o que mantém a complexidade computacional inalterada. Este método permite que o modelo capte mais informações contextuais [56].

A aplicação das convoluções *atrous* é feita em um modelo ResNet, pelo qual, foram feitas duplicatas do último bloco de origem do modelo, tais duplicatas foram disposta em cascata. A grande modificação realizada na arquitetura padrão é a redução do stride do último pooling ou camada convolucional para 1, evitando a perda de resolução. Em seguida, todas as camadas convolucionais seguintes são substituídas por camadas *atrous* com fator = 2, permitindo a extração de características mais densas sem a necessidades do aprendizado de novos parâmetros [54]. A Figura 2.12 apresenta tais modificações realizadas em um modelo sem convolução *atrous*.

Cada um dos referidos blocos possui três convoluções 3x3, sendo que a última convolução possui stride 2, exceto no último bloco. Em um modelo sem convolução *atrous* existe

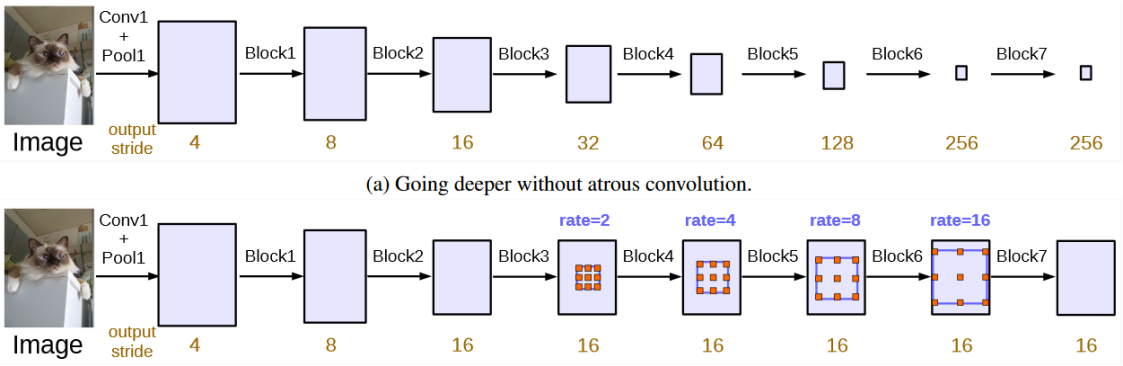

Figure 3. Cascaded modules without and with atrous convolution.

Figura 2.12: Módulos em cascata com e sem convolução *atrous* [54].

uma perda de informações detalhadas no último mapa de recursos, devido a sua baixa resolução, como se pode ver na Figura 2.12 (a), portanto são aplicadas convoluções *atrous*, utilizando taxas de expansão determinadas por meio do valor do *output* stride (*output* stride refere-se a razão entre as resolução da imagem de entrada e de saída) desejado, conforme apresentado na Figura 2.12 (b), no qual o *output* stride é 16. Neste modelo, caso nenhuma convolução *atrous* for utilizada o *output* stride seria 256 [54].

Na abordagem do módulo paralelo utilizando Atrous Spatial Pyramid Pooling (ASPP) quatro convoluções *atrous* são aplicadas no topo do mapa de recursos, reamostrando características em diversas escalas para classificar regiões de uma escala arbitrária. A Figura 2.13 apresenta uma arquitetura com ASPP [54].

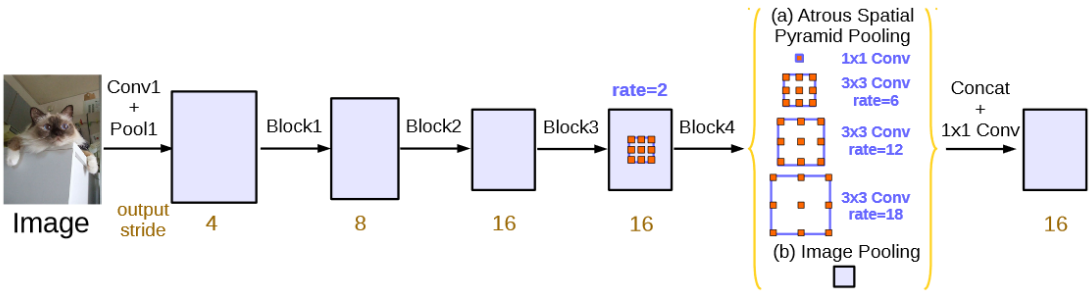

Figura 2.13: Módulos paralelos com convolução *atrous* [54].

O ASPP possui uma convolução 1x1 seguida de 3 convoluções *atrous* 3x3 com taxas 6, 12 e 18 quando o *output* stride é 16 (Figura 2.13 (a)), e os recursos no nível de imagem

ilustrado na Figura 2.13 (b). Para integrar informações de contexto global, um *global average pooling* é aplicado no último mapa de recursos do modelo, os recursos em nível de imagem passam por um convolução 1x1 com 256 filtros e *batch normalization*, em seguida a imagem é ampliada bilateralmente à dimensão desejada. Por fim, os recursos de todas as ramificações são concatenados e uma convolução 1x1 com 256 filtros e *batch normalization* é aplicada antes da convolução final 1x1 que produz os logits [54].

2.5 Transferência de Aprendizado

Diversos métodos de aprendizado de máquina apresentam um bom desempenho quando os dados de treinamento e teste são retirados do mesmo espaço de recursos e da mesma distribuição. Contudo, tais modelos estatísticos geralmente necessitam de uma reconstrução do zero quando uma alteração na distribuição ocorre, utilizando as características de treinamento recém adquiridas. Normalmente, em determinadas aplicações, é impossível ou caro reaver os dados de treinamento necessários para reconstrução dos modelos. Em casos como este, a utilização de transferência de aprendizado ou transferência de conhecimento torna-se interessante [57]. A Figura 2.14 apresenta a diferença entre um treinamento tradicional e um treinamento utilizando *transfer learning*.

As técnicas tradicionais de aprendizado de máquina empenham-se em aprender cada uma das tarefas desde o início. Em contrapartida os métodos de *transfer learning* utilizam o conhecimento já obtido de tarefas anteriores e tentam transferir este conhecimento para uma tarefa nova, quando esta possui menos dados de treinamento de qualidade [57].

O propósito do *transfer learning* é utilizar um conhecimento de uma tarefa origem para aprimorar o processo de aprendizado em uma determinada tarefa alvo. Existem três métricas comuns pelas quais a transferência pode aprimorar o resultado. Um primeiro ponto é a comparação entre a performance inicial atingível na tarefa alvo utilizando apenas o conhecimento transferido, sem qualquer aprendizado realizado antes, e a performance inicial de um agente ignorante. O segundo é a diferença de tempo necessário para o aprendizado completo de uma tarefa alvo, dado o conhecimento transferido em

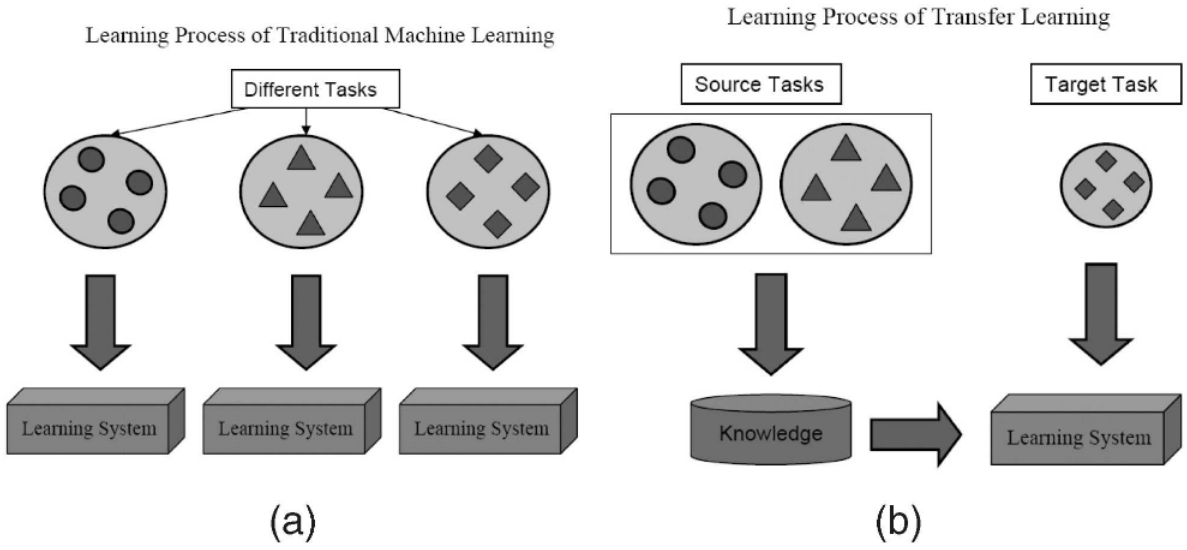

Figura 2.14: Processos de aprendizado (a) método tradicional de *machine learning* e (b) *transfer learning* [57].

comparação ao aprendizado do zero. Por fim, o terceiro diz respeito ao desempenho final do aprendizado da tarefa alvo, utilizando transferência e sem a utilizar [58]. A Figura 2.15 apresenta a comparação entre o aprendizado por transferência e o aprendizado sem auxílio de conhecimento externo.

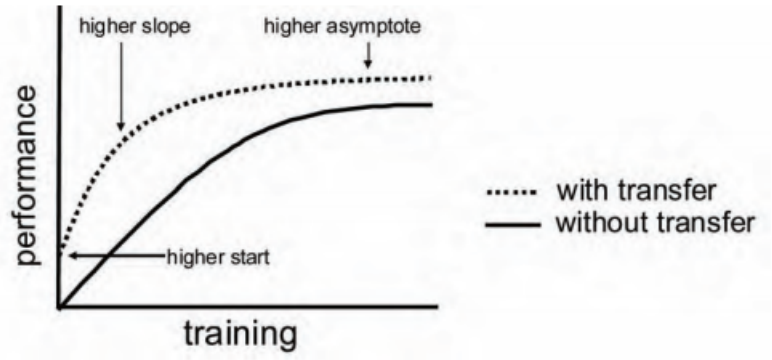

Figura 2.15: Aumento de desempenho fornecido pelo *transfer learning* [58].

O *transfer learning* pode ser subdividido em três métodos, o primeiro é o aprendizado de transferência indutiva, no qual a tarefa de origem é diferente da tarefa de destino, independentemente dos domínios de origem e destino serem equilaventes ou não. No segundo,

denominado aprendizado de transferência transdutiva, as tarefas de destino e origem são equivalentes, porém os domínios de origem e destino são divergentes. Por fim, o aprendizado de transferência não supervisionado, que atua de maneira análoga ao aprendizado de transferência indutiva, a tarefa de destino é divergente, porém é semelhante à tarefa de origem. Neste método ambos os domínios não possuem dados rotulados [57].

Uma arquitetura CNN é composta por camadas convolucionais responsáveis pela extração de características, seguidas de camadas totalmente conectadas que fazem o papel de classificador. Neste cenário, a transferência de aprendizado consiste em reutilizar o módulo de extração de características do domínio de origem no domínio de destino e modificar o classificador treinado no domínio de origem às novas amostras do domínio de destino [59]. De acordo com [60], o *transfer learning* pode ser usado em casos em que a quantidade de dados de uma base de dados é limitada, ou seja, a substituição de treinamento do zero por CNNs pré treinadas em bancos de grande dimensão (como o ImageNet [61]) pode evitar o *overfitting*. A utilização deste recurso pode ainda reduzir o tempo necessário para o treinamento no domínio de destino, bem como diminuir significativamente a quantidade de dados exigidos para o treinamento [62].

2.6 Aumento de Dados

Geralmente métodos de classificação de imagens e vídeos têm como grande obstáculo a escassez de dados. Em especial, pesquisas relacionadas com a indústria médica, pois o acesso aos dados é protegido devido a questões de privacidade destes dados. O real problema de conjuntos de dados pequenos é que modelos treinados com base nestas bases não generalizam bem. Este problema é denomidado como *overfitting* [63].

O *data augmentation* é uma das maneiras de reduzir o *overfitting* dos modelos [63]. Esta técnica tem como objetivo aumentar artificialmente a base de dados existente por meio de operações como translação, rotação, inversão, corte e adição de ruídos. Dentre estas operações os métodos mais eficazes no treinamento de CNN são corte aleatório e inversão aleatória [64]. A Figura 2.16 apresenta exemplos de operações utilizadas para

aumento de dados.

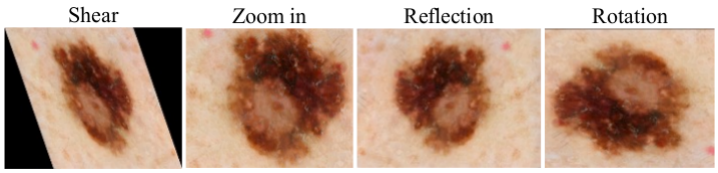

Figura 2.16: Exemplo de operações de *data augmentation* [65].

Geralmente características de iluminação também são desafios frequentes em problemas de reconhecimento de imagens. Desta forma, a eficácia de operações no espaço de cores tornam tais transformações intuitivas em técnicas de *data augmentation*. Por exemplo, em situações de imagens demasiadamente escuras ou claras é possível utilizar um valor constante para aplicar aumentos ou decrementos aos valores dos pixeis. Outra técnica consiste na restrição dos valores de pixeis a valores máximos e mínimos. Existem diversas transformações possíveis para aplicação no espaço de cores que podem solucionar problemas como iluminação ou variação de cores nas imagens [66]. A Figura 2.17 apresenta alguns exemplos de trasformações realizadas por meio do espaço de cores.

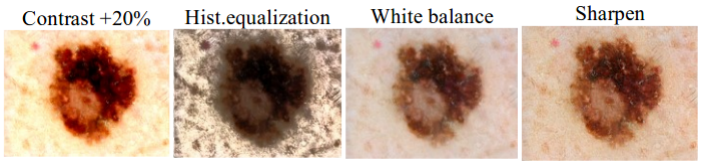

Figura 2.17: Exemplo de operações de *data augmentation* aplicadas ao espaço de cores [65].

Tais operações permitem que um modelo de aprendizado profundo aprenda invâriancia nas imagens da base de dados. A invariância é um fator de grande relevância na segmentação biomédica, pois transformações utilizadas para variação dos dados podem ser simuladas com eficiência [52]. Desta forma, torna-se possível a superação de problemas como iluminação, oclusão, plano de fundo, escala e outros [66].

2.7 TensorFlow

O TensorFlow (TF) pode ser definido como uma plataforma de código aberto destinada ao aprendizado de máquina. A partir de um amplo e adaptável ecossistema de ferramentas, bem como diversas bibliotecas e recursos da comunidade, o TF permite a aplicação de técnicas de aprendizado de máquina em áreas de pesquisa e desenvolvimento [67]. Apesar do TF englobar uma ampla gama de funcionalidades seu foco é principalmente aos modelos de redes neurais profundas [68]. A flexibilidade do TF permite o treinamento e a implantação de um modelo independentemente da linguagem ou da plataforma utilizada. Desta forma, é possível criar e implantar modelos de aprendizado de máquina para web por meio do JavaScript utilizando o TensorFlow.js, ou utilizar o TensorFlow Lite para executar inferências em dispositivos móveis e embarcados [67]. González [69] pontua que, a partir da criação de um modelo, é possível salvá-lo e implementá-lo em outro dispositivo.

O TF Lite é definido com um grupo de ferramentas que auxiliam os desenvolvedores a executar modelos do TensorFlow em dispositivos móveis, Internet of Things (IoT) e embarcados. A partir dessas ferramentas ele permite que dispositivos executem inferência de aprendizado de máquina com baixa latência e um tamanho binário reduzido [67].

Diversos níveis de abstração são oferecidos pelo TF com o intuito de facilitar a utilização e o primeiro contato com técnicas de aprendizado de máquina, desta maneira, utilizando a *high-level* Keras API é possível construir e treinar modelos de forma facilitada. Por outro lado, para criação e treino de modelos robustos, o TF fornece flexibilidade e controle por meio de recursos como a Model Subclassing API e Keras Functional API, facilitando o desenvolvimento de protótipos e fornecendo depuração rápida [67].

Apesar de existirem diversas bibliotecas para aprendizagem profunda, o TF se destaca por possuir sintaxe clara e uma ferramenta robusta para visualização denominada TensorBoard [35]. O conjunto de ferramentas TensorBoard tem como objetivo facilitar a compreensão, a depuração e a otimização dos programas do TF [67].

O TF possui Redes neurais convolucionais como MobileNet e Inception que são modelos pré-treinados para classificar centenas de classes, como pessoas, atividades, animais,

plantas e lugares [67]. A técnica transferência de aprendizado é utilizada para reaproveitar uma rede neural já treinada anteriormente para um conjunto de classes. A execução de um novo treinamento sobre os pesos da rede permite que novas classes sejam determinadas. Desta forma, a partir do treinamento somente da última camada é possível obter bons resultados, com baixo tempo de treino e menor necessidade de recursos de hardware [70].

2.8 Estado da arte

Apesar de não ser possível realizar uma comparação justa de desempenho entre outros trabalhos e a presente pesquisa devido à utilização de uma base de imagens própria, a seguir serão apresentados trabalhos relacionados ao objetivo principal desta pesquisa, especificamente pesquisas relacionadas a doença psoríase, com base em soluções de segmentação, classificação e dispositivos móveis, com o intuito de contextualização e resumo do estado atual desta linha de pesquisa.

Na abordagem de Shrivastava et al. [71], um sistema dermatológico Computer-aided Diagnosis (CADx) foi proposto para classificar imagens de pele como saudável ou com psoríase, por meio de uma base de 540 imagens de 30 pacientes diferentes, sendo 270 imagens de pele com psoríase e 270 imagens de pele saudável, cada imagem foi cortada manualmente por meio da ferramenta MATLAB. A partir de um espaço de recurso exclusivo, composto por espaço de tons de cinza, espaço de cores e agressividade da doença, e um classificador Support Vector Machine (SVM), obteve-se como melhor precisão 99,81% por meio do procedimento de validação cruzada.

Em outra abordagem, Shrivastava et al. [72] propõe um sistema de diagnóstico para classificação de imagens de pele saudável e com lesão de psoríase. Utilizando SVM de núcleo polinomial do tipo dois e parâmetros de cor para o aprendizado de 540 imagens com igual número em cada classe, saudável e doente, obteve-se acurácia de 99,94%, por meio do método de validação cruzada com 10 divisões.

Em Wei, Gan e Ji [73] é proposto um sistema para identificação de três doenças de pele:

herpes, dermatite e psoríase. Os métodos de pré-processamento, matriz de concorrência e SVM foram utilizados para realizar a identificação das classes, por meio de características de textura e cor. Após avaliações de desempenho o sistema obteve maior acurácia nas imagens com psoríase com 95% de taxa de reconhecimento.

Velasco et al. [74] utilizaram um modelo MobileNet com *transfer learning* para classificação de 7 tipos de doenças dermatológicas diferentes, sendo uma delas psoríase, a partir de uma aplicação Android. O modelo foi treinado a partir de um dataset com um total de 3406 imagens desbalanceadas de todas as 7 lesões analisadas. Técnicas como *under-sampling*, *oversampling* e *data augmentation* foram utilizadas para selecionar a melhor abordagem e aumentar o desempenho do modelo. Para treinamento, foi utilizada uma entrada de dimensão 224x224x3, taxa de aprendizado de 0,0001, optimizador Adam e 30 épocas, todas as camadas do modelo foram congeladas e somente a última camada foi treinada. A melhor abordagem foi a combinação das técnicas de *oversampling* e *data augmentation*, fazendo com que o modelo atingisse 94,4% de acurácia. A aplicação móvel conta com o módulo de captura de imagens, um módulo para análise a partir do modelo MobileNet e um módulo para apresentação de informações acerca dos resultados das classificações.

Dash et al. [75] aplicaram um modelo U-net modificado para segmentar imagens de psoríase. Foram utilizadas 5241 imagens de lesões de psoríase, com dimensão 128x128x3, adquiridas de 1026 pacientes por um dermatologista. Utilizando menos camadas e parâmetros do que o modelo U-net original, os autores atingiram uma acurácia de 94,80%.

Utilizando um sistema de aprendizado profundo com 3 módulos, Dash et. al [76] desenvolveram um framework que classifica uma imagem como com psoríase ou sem psoríase, segmenta a imagem destacando a lesão e posteriormente utiliza tal imagem segmentada para avaliar a severidade da lesão. Foram utilizadas 5000 imagens de pele com psoríase e 5000 imagens de pele saudável, dentre as imagens com lesão uma subdivisão foi feita para criar 4 classes referentes a cada grau de severidade avaliado pelo sistema. No primeiro módulo um modelo vgg-16 modificado foi utilizado e teve como resultado uma acurácia média de 99,08%. Na segunda etapa, no procedimento de segmentação um modelo Unet

modificado foi usado e garantiu acurácia de 94,76%. Por fim, no módulo de avaliação de severidade o mesmo modelo da primeira etapa foi utilizado, alterado para classificação multi classes, e obteve acurácia média de 92,64%. No último módulo foram realizados testes classificando as imagens segmentadas e não segmentadas, e o melhor desempenho foi obtido utilizando imagens segmentadas automaticamente pelo segundo módulo.

A partir de um sistema composto por 2 etapas, Zhao et. al [77] desenvolveram um sistema de classificação de lesões dermatológicas. A primeira etapa têm como foco classificar imagens dentre 9 classes de diferentes lesões de pele. Na segunda etapa a saída da primeira predição é utilizada, adicionando as probabilidades de 8 classes os autores compararam com a probabilidade da classe psoríase, obtendo assim um sistema de classificação entre psoríase e não psoríase. As arquiteturas DenseNet, InceptionV3, InceptionResnetV2 e Xception foram utilizadas e treinadas a partir de um dataset de aproximadamente 8000 imagens contendo 9 lesões dermatológicas diferentes, sendo 900 delas imagens de psoríase. O modelo InceptionV3 foi o que obteve maior desempenho a partir da métrica Area Under The Curve (AUC), com um resultado de 0,981 $\pm$ 0,015. Em uma outra abordagem, os autores desenvolveram outros dois datasets para testes, ambos com 50 imagens de diversas doenças dermatológicas, porém um com imagens semelhantes a psoríase e o outro com imagens relativamente diferentes da psoríase, nestes testes o modelo atingiu 88% de acurácia média.

Na pesquisa de Padilla et. al [78], um sistema para classificação de pele foi desenvolvido com o objetivo de classificar imagens entre as classes psoríase, dermatite atópica e desconhecido, para imagens como pele sem lesão. Utilizando a arquitetura MobileNet e um dataset composto por 6264 imagens de treino, sendo 2656 de dermatite atópica, 2524 de psoríase e 1084 da classe desconhecido, e 30 imagens capturadas diretamente de um Raspberry Pi para testes, os autores atingiram uma acurácia de 88% para dermatite atópica e 90% para psoríase.

Capítulo 3

Abordagem e Análise

A seguir serão apresentados diagramas, fluxogramas e modelagens do funcionamento da arquitetura desenvolvida. Partindo do geral ao específico do sistema proposto, são utilizadas ferramentas UML para apresentar e simplificar o entendimento do eco sistema da aplicação em geral.

3.1 Levantamento de requisitos

Neste tópico são apresentados os requisitos funcionais e os requisitos não funcionais que foram levantados para desenvolvimento do sistema proposto.

3.1.1 Requisitos Funcionais

Segundo Sommerville [79], um Requisito Funcional (RF) refere-se às funções que deverão ser realizadas pelo sistema. A Tabela 3.1 apresenta os requisitos funcionais estipulados para o projeto.

3.1.2 Requisitos Não Funcionais

Um Requisito Não Funcional (RNF), de acordo com Sommerville [79], não está relacionado de forma direta aos serviços específicos proporcionados pelo sistema a seus usuários. A

Tabela 3.1: Requisitos funcionais do sistema.

Requisito	Descrição
RF[001] Acessar a câmera	A aplicação deverá solicitar o acesso à câmera do dispositivo móvel para que o utilizador efetue a aquisição das imagem.
RF[002] Capturar fotografias	A aplicação deverá ser capaz de capturar fotografias por meio da câmera nativa do dispositivo.
RF[003] Classificação binária	O sistema deverá ser capaz de inferir sobre uma imagem, retornando uma classificação binária (pele com psoríase ou pele sem psoríase) e os dados fornecidos pelo classificador.
RF[004] Segmentação de imagem	O sistema deverá ser capaz de segmentar a imagem submetida.

Tabela 3.2 apresenta os requisitos não funcionais levantados para o sistema proposto.

Tabela 3.2: Requisitos não funcionais do sistema.

Requisito	Descrição
RNF[001] Biblioteca TensorFlow	O sistema deverá utilizar a biblioteca TensorFlow para realizar o processamento e a inferência das imagens.
RNF[002] Plataforma de funcionamento	O aplicativo deverá funcionar em dispositivos móveis que utilizam a plataforma Android como sistema operacional.
RNF[003] Fácil manuseio	O sistema deve fornecer facilidade para manuseio por parte dos utilizadores.
RNF[004] Tempo de análise	O sistema deve realizar a análise da imagem submetida em até 800ms.

3.2 Escopo geral do projeto

O sistema proposto tem como objetivo auxiliar pessoas no rápido diagnóstico e tratamento da doença psoríase, por meio de uma aplicação móvel que sugere uma possível presença

da doença psoríase. Desta forma, os módulos básicos necessários ao sistema são: uma aplicação móvel, um servidor web e uma plataforma baseada em aprendizado profundo. A Figura 3.1 apresenta uma visão geral do sistema proposto.

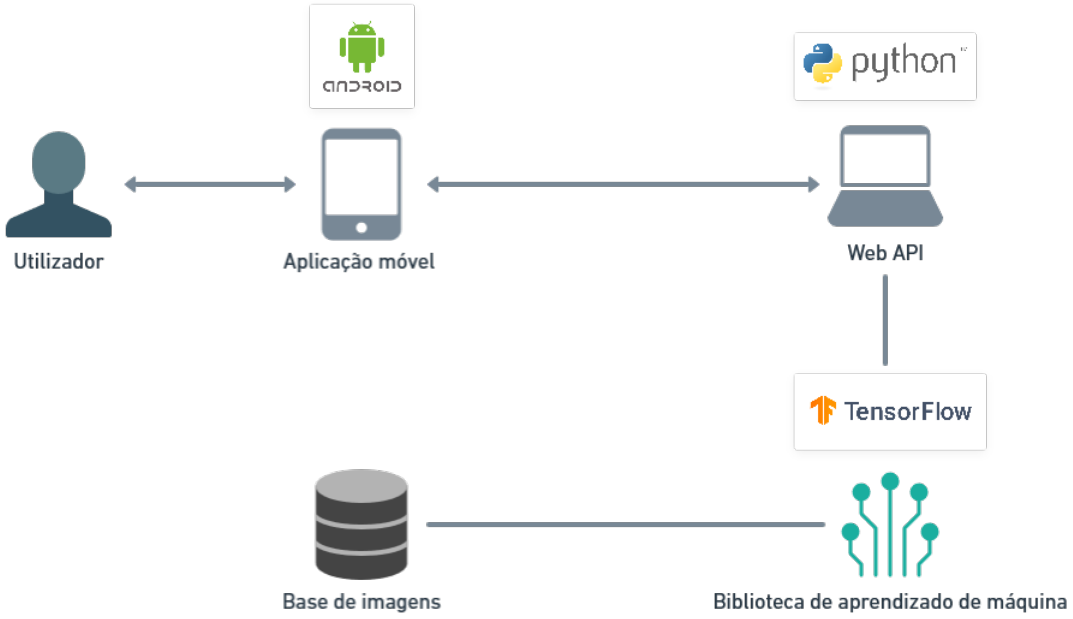

Figura 3.1: Arquitetura cliente servidor utilizada para classificação.

Como apresentado na Figura 3.1 a arquitetura pode ser caracterizada como cliente servidor, sendo os maiores componentes: o utilizador, a aplicação móvel, o servidor web, a biblioteca de aprendizado de máquina e a base de imagens utilizada para treino e teste dos modelos de rede neural. A Figura 3.2 apresenta os casos de uso da aplicação, que representam as interações do utilizador no sistema.

Assim como apresentado na Figura, para efetuar a análise de imagens da aplicação o utilizador precisa permitir que a aplicação acesse o hardware de câmera nativa do dispositivo. No caso de uso capturar imagem, estão as ações de aquisição e recorte das fotografias. No caso de uso analisar imagem são efetuadas as tarefas de classificação, segmentação e apresentação dos dados de resultado.

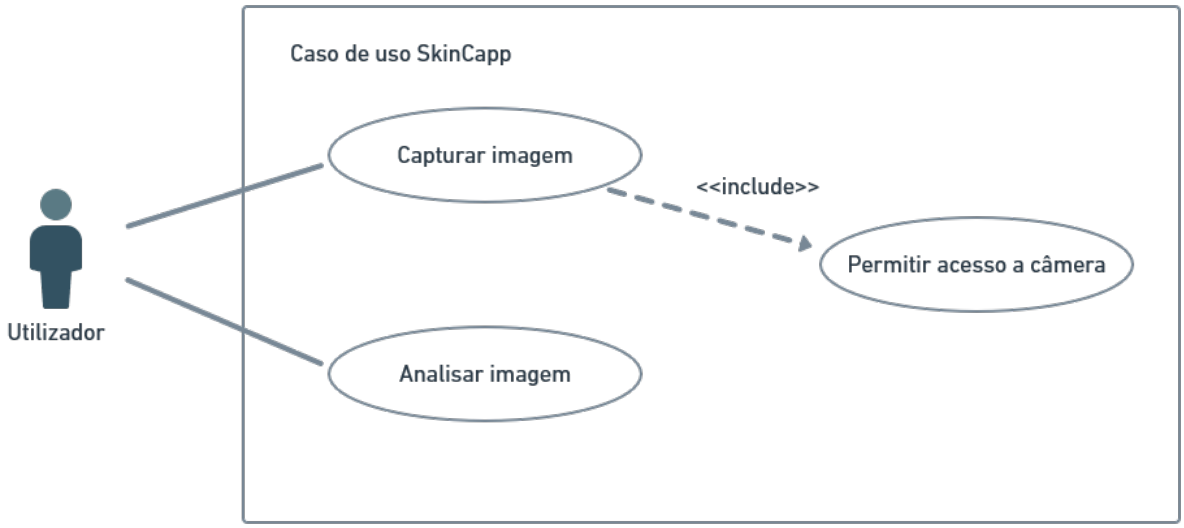

Figura 3.2: Diagrama de caso de uso do sistema.

3.2.1 Diagramas de Atividades

A seguir serão apresentados os diagramas de atividades desenvolvidos para planejamento e documentação dos processos e arquiteturas do sistema.

Aplicação móvel

Optou-se por manter o fluxo de navegação entre interfaces na aplicação móvel o mais simples possível, focando efetivamente no processo de captura e análise da imagem submetida. Desta forma, a aplicação conta com 3 interfaces, sendo uma delas a principal. Ao iniciar a aplicação existe um circuito de navegação entre as 3 interfaces, começando pela interface inicial e terminando na interface com os resultados da análise, a Figura 3.3 apresenta este fluxo de navegação.

Como apresentado na imagem existe a possibilidade de cancelamento e retorno à interface principal em qualquer interface. O circuito inicia e termina na interface principal pois os resultados da captura e da análise são inseridos na mesma interface de forma dinâmica, ou seja, caso exista uma imagem capturada ou um resultado de análise, o layout da interface se altera para apresentar tais informações.

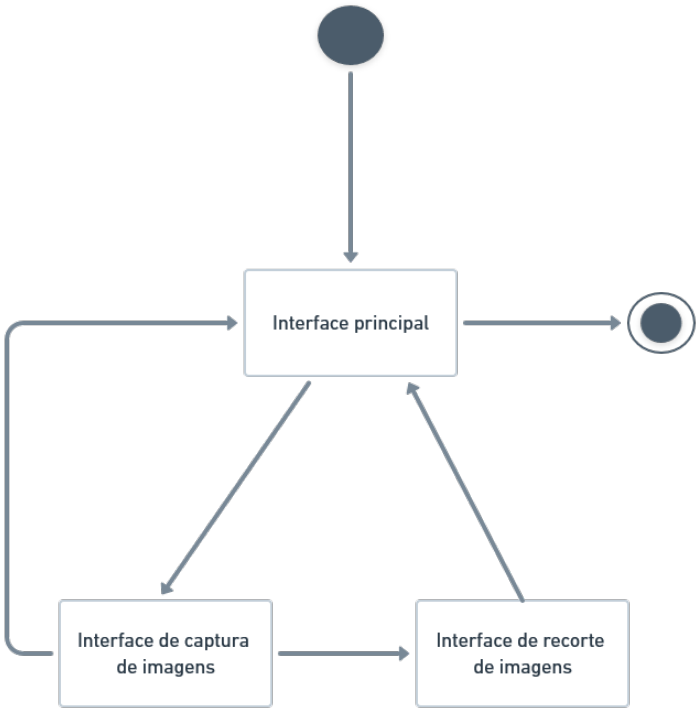

Figura 3.3: Diagrama de atividades do fluxo de navegação entre interfaces da aplicação.

Web API

O servidor web é composto por um conjunto de atividades relativamente simples, o único ponto de decisão é após a classificação, pelo qual a próxima ação a ser executada depende do resultado da análise da imagem, desta maneira, caso o resultado da classificação seja psoríase, são adicionadas as atividades de segmentação e codificação da imagem antes da devolução do resultado. A Figura 3.4 apresenta este funcionamento.

Redes Neurais

O processo de treinamento, análise e seleção das redes neurais de classificação e segmentação foi desenvolvido de forma iterativa e adaptativa. Para cada modelo testado existem 5 principais atividades, que são executadas iterativamente até que um resultado satisfatório seja obtido. Iniciando pela seleção das imagens, passando por estruturação da base de dados, ajuste de parâmetros, re-treinamento do modelo e finalizando na aplicação de teste

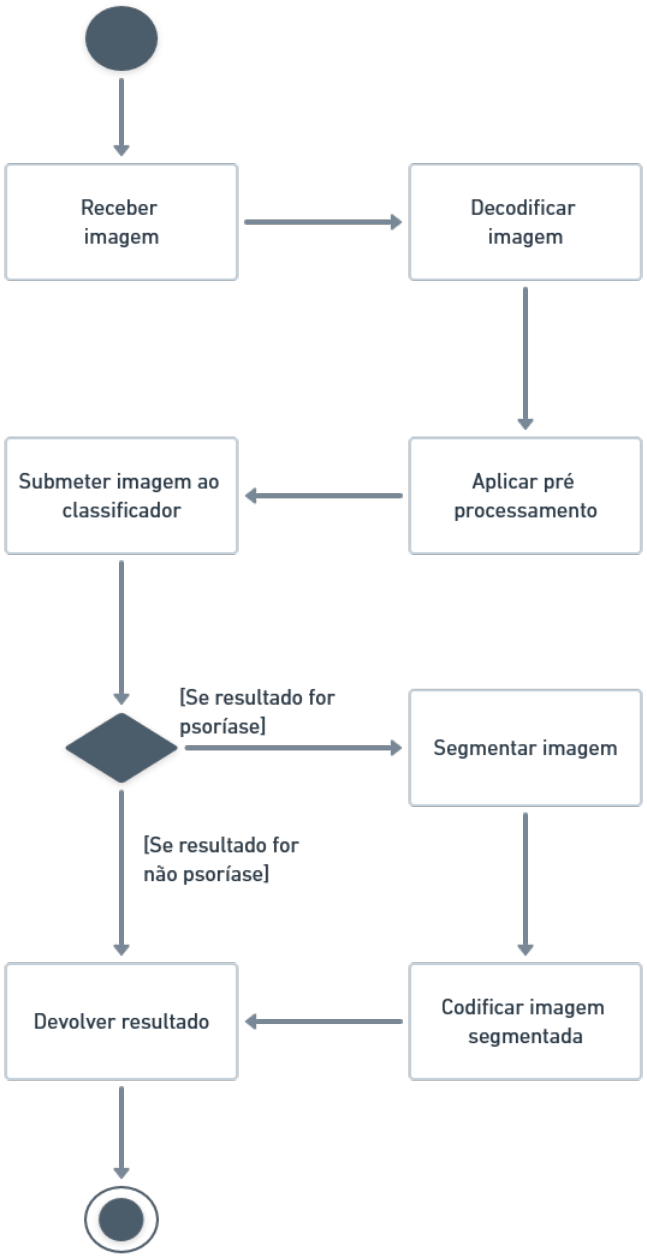

Figura 3.4: Diagrama de atividades do servidor web.

e medidas de eficiência. A Figura 3.5 apresenta este circuito.

A atividade de estruturação da base de dados engloba desde a seleção das imagens para treino e teste até a aplicação das operações de data augmentation.

3.2.2 Fluxogramas

A seleção do melhor modelo foi dada a partir dos resultados obtidos de uma sequência de testes realizados, cada etapa desta sequência de testes tem como objetivo apontar qual a melhor arquitetura, recurso ou método a se utilizar. Ao todo foram mapeadas 6 etapas, todas elas visando a melhor abordagem na resolução do problema com o melhor desempenho possível. A Figura 3.6 apresenta a relação dos testes realizados e a respectiva sequência empregada.

Todas as setas conectadas entre as caixas de atividades representam os testes feitos, cada caixa de atividade representa uma arquitetura, recurso ou método que foi testado. Desta maneira, ao chegar no final desta sequência de testes é obtida a melhor combinação para atingir o objetivo do projeto.

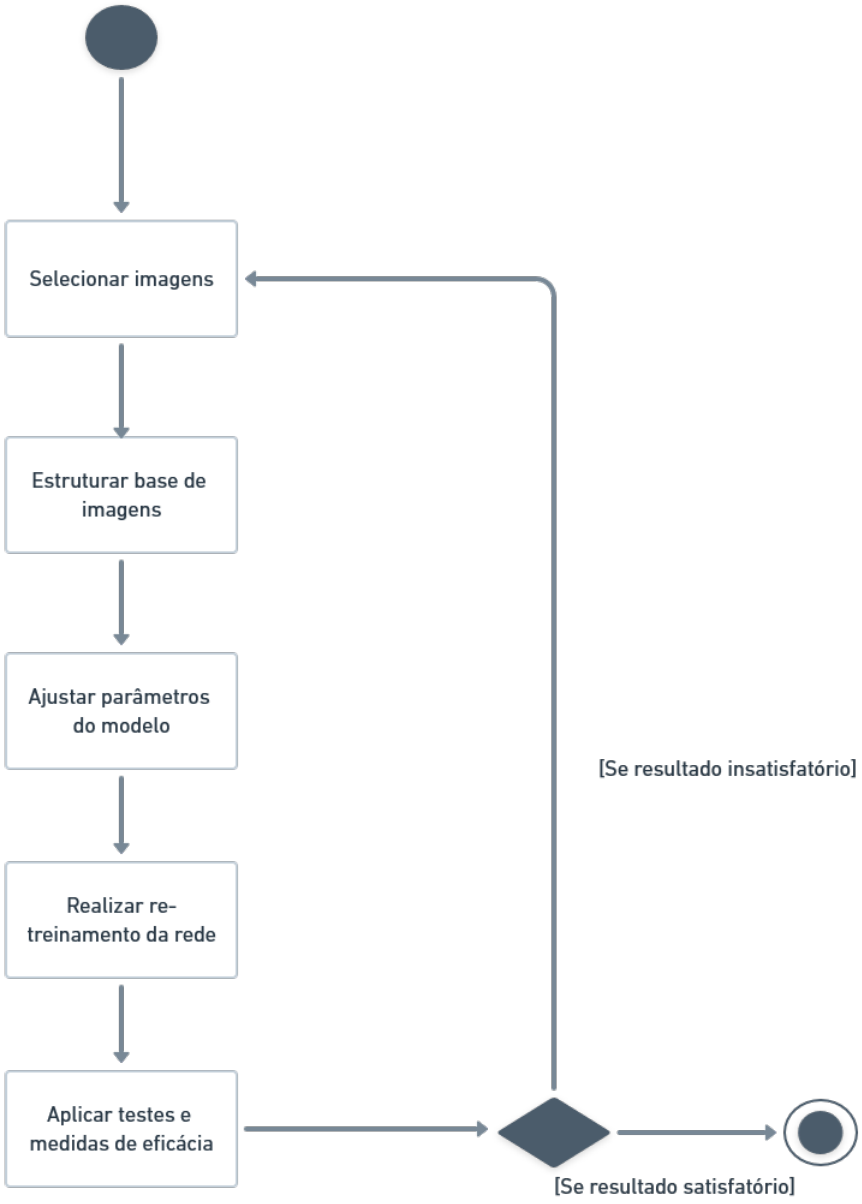

Figura 3.5: Diagrama de atividades do treinamento dos modelos.

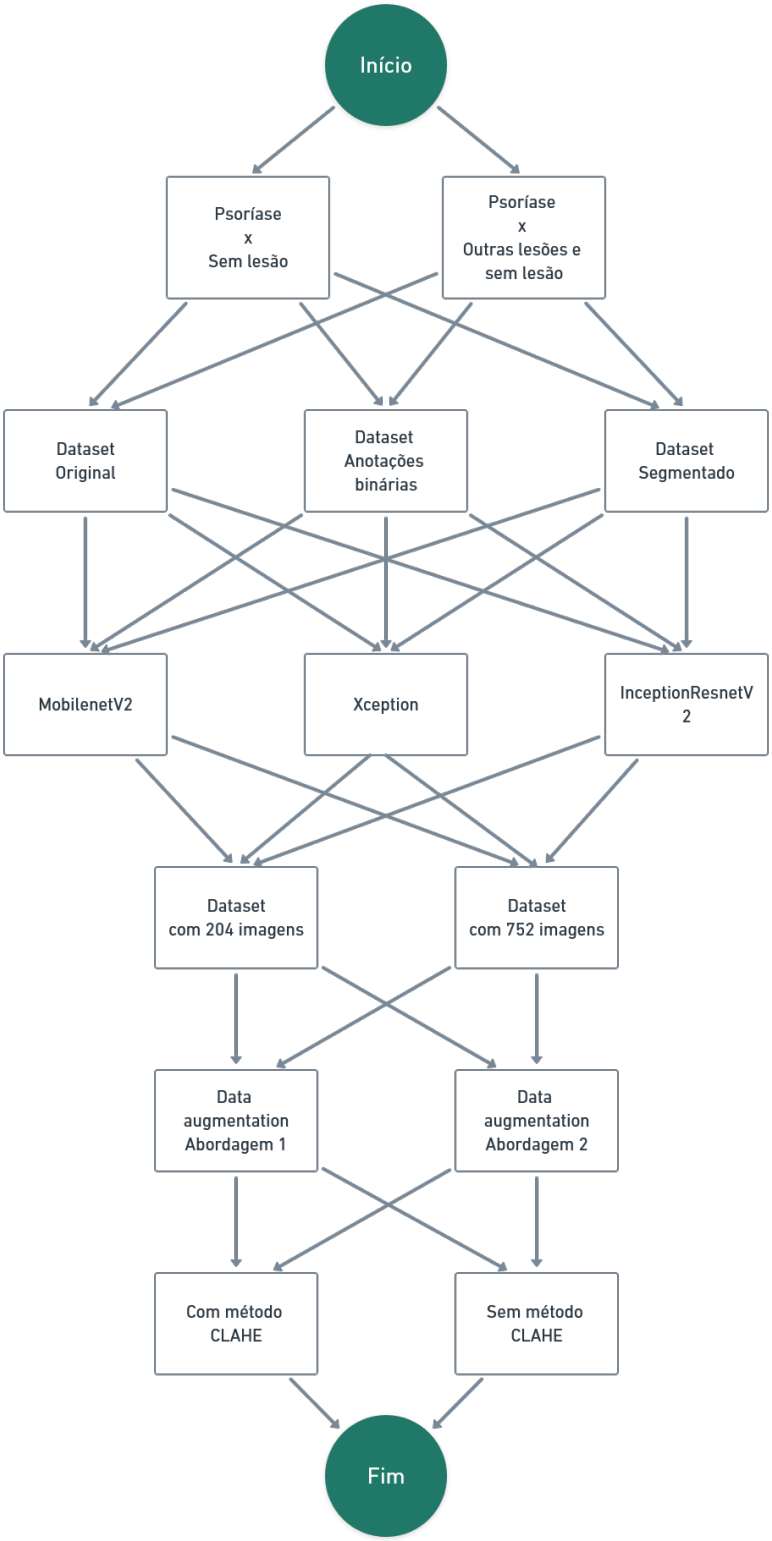

Figura 3.6: Fluxograma de testes mapeados.

Capítulo 4

Implementação

A seguir serão descritas as metodologias empregadas em cada um dos módulos desenvolvidos neste trabalho de forma detalhada, apresentando as partes mais importantes e as dificuldades encontradas em cada etapa.

4.1 Base de Imagens

A obtenção de imagens relacionadas a área médica é geralmente um grande desafio. Imagens dermatológicas especificamente possuem uma grande limitação devido a exposição de pessoas, o que necessita de autorizações e consentimento das pessoas fotografadas. Foram realizados contatos com diversos autores de pesquisas relacionadas a doença psoríase para solicitação das bases de imagens utilizadas em suas pesquisas, contudo poucas respostas foram obtidas. Os autores que forneceram uma resposta afirmaram que não tinham permissão para compartilhar as imagens utilizadas. A base mais promissora encontrada é denominada como Xian-gyaDerm [80] e possui 107.565 imagens clínicas de diversas lesões dermatológicas, sendo 67.066 imagens de psoríase. Uma solicitação de acesso foi enviada aos autores, porém, sem retorno.

Dada tal dificuldade de obtenção de imagens, a base de imagens utilizada para treinamento e validação na etapa de segmentação é composta por imagens obtidas nas seguintes

plataformas de dermatologia: International Psoriasis Council (IPC)[1], DermNetNZ[2], DermIS[3], DanDerm[4] e Hellenic atlas[5], listadas na Tabela 4.1. A utilização das imagens é permitida pelas plataformas citadas.

Tabela 4.1: Fontes das imagens utilizadas.

Fonte	**Quantidade de imagens**
International Psoriasis Council (IPC)	376
DermnetNZ	370
DermIS	193
DanDerm	170
Hellenic derm. atlas	97

Imagens de couro cabeludo, unhas e genitálias não foram utilizadas pois possuem características especificas que poderiam confundir o modelo. Para criação das anotações foram selecionadas 82 imagens com diferentes tons de pele, severidade da lesão e localização no corpo, bem como, diferentes posições, ângulos e condições de claridade. As anotações de segmentação foram criadas a partir do preenchimento de toda a área de lesão da imagem em pixeis brancos, e o restante em pixeis pretos. A Figura 4.1 apresenta uma amostra das imagens utilizadas e suas respectivas anotações binárias. As imagens originais são apresentadas na primeira linha (Figura 4.1 (a)) e as anotações binárias estão localizadas na segunda linha (Figura 4.1 (b)).

Uma abordagem utilizando apenas imagens com a lesão para treinamento da rede de segmentação foi efetuada, porém não se obteve um resultado satisfatório uma vez que a rede treinada fazia demasiadas predições de falso positivo, apontando áreas de lesões em imagens que não possuíam lesões. Desta forma, a base de imagens principal conta com 164 imagens de treinamento, no qual, metade é composta por imagens com lesão e a outra metade por imagens sem lesão, para realização da etapa de teste 40 imagens foram selecionadas, também com metade das imagens sendo de lesão e a outra metade

[1] www.psoriasiscouncil.org
[2] www.dermnetnz.org
[3] www.dermis.net
[4] www.danderm-pdv.is.kkh.dk/atlas/index.html
[5] www.hellenicdermatlas.com

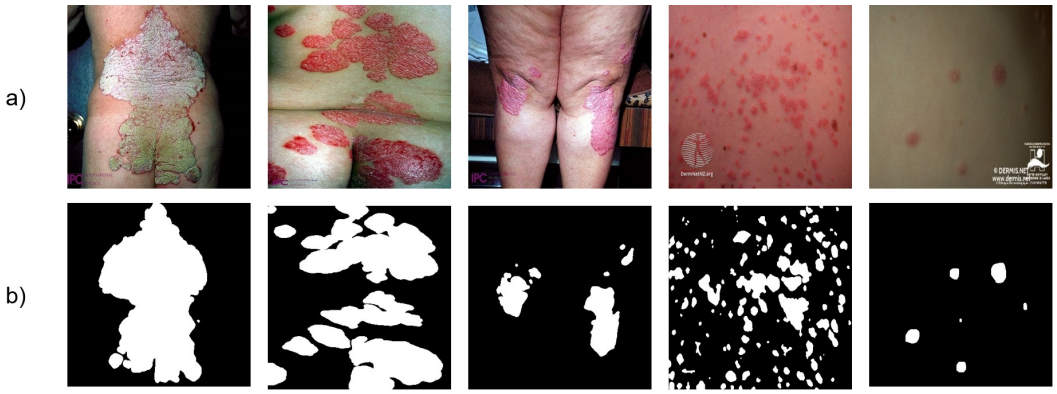

Figura 4.1: Amostra de imagens contidas no dataset.

sem lesão. Tais quantidades de imagens das bases de treino e testes referem-se a imagens sem operações de data augmentation, ou seja, os datasets utilizados efetivamente possuem uma quantidade maior de imagens, como apresentado na próxima seção.

Na etapa de classificação três modelos de dataset são propostos, com o intuito de comparar e selecionar a melhor abordagem. O primeiro dataset é o mesmo utilizado na etapa de segmentação, constituído apenas pelas imagens RGB originais. A segunda abordagem utiliza somente as máscaras geradas pela rede de segmentação. A última abordagem utiliza o resultado das predições da rede de segmentação multiplicadas pelas imagens originais, o que produz uma espécie de corte, apresentando somente as regiões de lesão e o restante representado em pixels pretos, sendo assim, imagens sem lesão são imagens totalmente pretas. Os 3 datasets possuem a mesma quantidade de imagens, dimensões e com a mesma subdivisão por imagens com e sem lesão. A Figura 4.2 apresenta amostras das imagens de cada dataset utilizado.

4.2 Aumento de dados

Como mencionado em capítulos anteriores, tarefas que envolvem estruturas de *deep learning* necessitam de uma ampla disposição de dados, para que o modelo alcance alto desempenho e evite o *overfitting*.

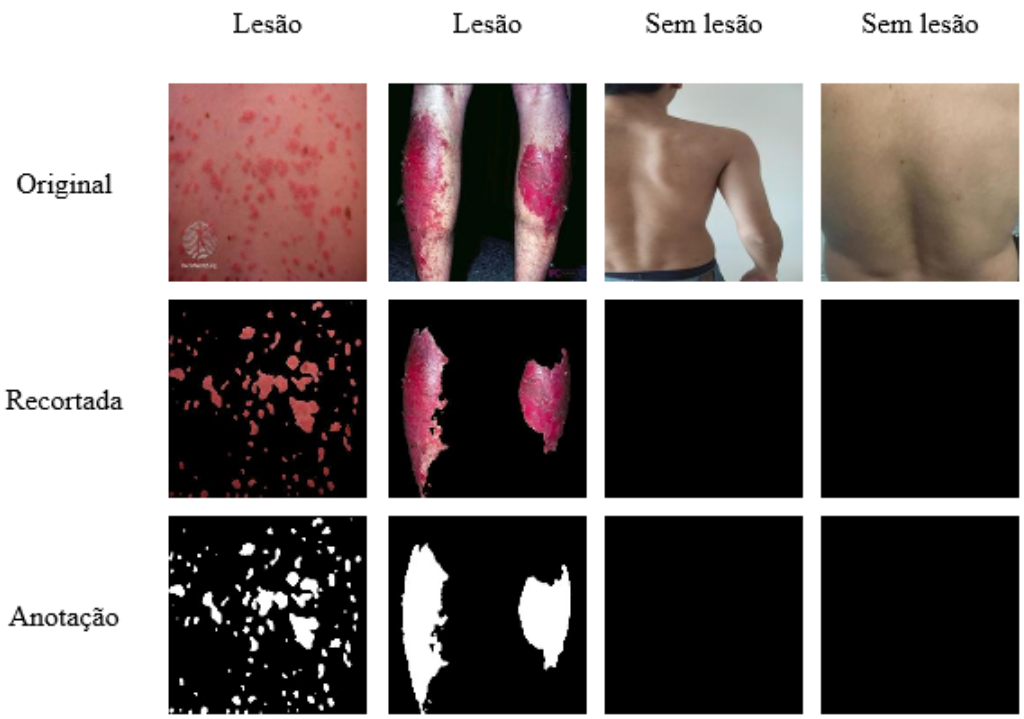

Figura 4.2: Exemplos de imagens dos 3 datasets utilizados para classificação.

Uma vez que o objetivo do modelo proposto é realizar inferências em imagens capturadas diretamente por usuários, ruídos e anomalias durante o processo de aquisição podem interferir no processo de análise do modelo. Por exemplo, excesso de exposição ao brilho no momento da captura ou alteração no espaço de cores da imagem devido ao ângulo de captura do ponto alvo, dentre muitos outros ruídos. Além das características intrínsecas da psoríase, que pode manifestar-se de diversas maneiras, com cores, dimensão, localização e texturas diferentes dependendo do tipo, da região e da severidade em que a doença se apresenta, tal variação pode dificultar a capacidade de generalização do modelo.

Com o intuito de ensinar ao modelo propriedades de invariância e robustez, suavizar o impacto de ruídos e interferências, bem como, para ampliar a quantidade de imagens do dataset, operações de *flip* horizontal, *flip* vertical, rotação e alteração do espaço de cores foram aplicadas às imagens. A Figura 4.3 apresenta um exemplo do resultado obtido após aplicação das operações mencionadas.

O primeiro bloco de imagens (Figura 4.3 (a)) refere-se as operações de *flip* e rotação,

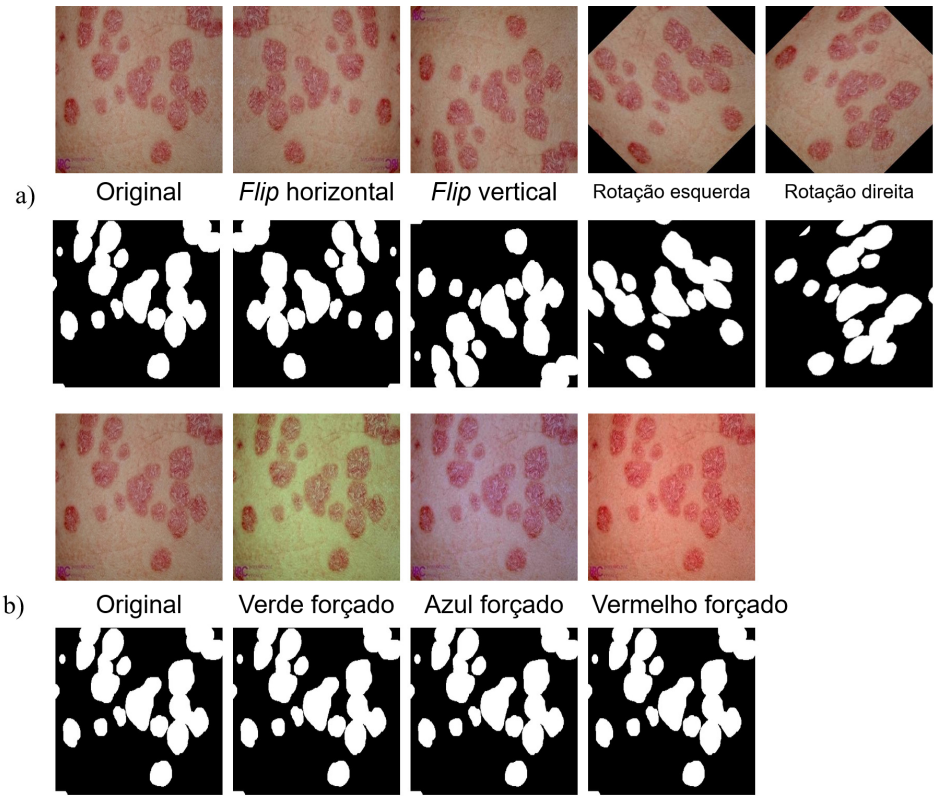

Figura 4.3: Operações de *data augmentation* aplicadas ao dataset.

e o segundo bloco (Figura 4.3 (b)) apresenta as operações de modificação do espaço de cores, no qual são aplicados aumentos nos valores de cada pixel em cada uma das dimensões RGB, fazendo com que a imagem assuma diferentes aspectos de cores. Como apresentado na Figura 4.3 as mesmas operações aplicadas nas imagens originais foram também efetuadas nas imagens ground truth, para que a semântica das máscaras não ficassem inconsistentes com a imagem. Após a aplicação de todas as operações, o dataset foi incrementado e passou de 204 imagens para 22848 imagens, sendo 18368 para treino e 4480 para teste. O mesmo dataset foi utilizado para os modelos de classificação, porém sem aplicação do *data augmentation* no conjunto de testes, mantendo somente as 40 imagens originais. A Tabela 4.2 apresenta as operações utilizadas e as proporções aplicadas em cada uma.

Tabela 4.2: Relação entre operações e proporções aplicadas no aumento de dados.

Operação	**Proporção**
flip horizontal	1
flip vertical	1
rotação esquerda	45º, 90º, 135º
rotação direita	45º, 90º, 135º
alteração do espaço de cores	1.1

4.3 Segmentação

A segmentação é proposta nesta pesquisa como uma ferramenta de apoio, pela qual é responsável por detectar zonas da imagem que possam conter lesão de psoríase e extrair todo o restante. Tal operação permite a remoção de objetos e planos de fundo de uma imagem que podem confundir ou atrapalhar o modelo no momento do treinamento para classificação. Por outro lado, uma vez que o objetivo deste trabalho é fornecer uma aplicação móvel para detecção de psoríase, a partir da segmentação semântica é possível apresentar ao utilizador final qual região da imagem submetida foi analisada para realização da predição final.

Para realização da segmentação semântica o modelo U-net [52] é utilizado como base. Três modelos foram testados, todos eles baseados na arquitetura encoder-decoder do modelo U-net. O primeiro modelo testado é semelhante a arquitetura tradicional da U-net, porém, para que as dimensões da imagem de entrada sejam iguais as de saída, todas as convoluções possuem padding zero, desta forma, zeros são adicionados nas bordas da imagem, garantindo que a imagem de saída terá o mesmo tamanho que a imagem de entrada. Camadas de *batch normalization* também foram inseridas para que o treinamento seja optimizado e para reduzir a *Internal Covariate Shift*. A segunda arquitetura proposta utiliza um modelo VGG-16 [81] como encoder e a estrutura de decoder da U-net. Nesta abordagem as camadas totalmente conectadas do modelo foram removidas, e a ultima camada convolucional do modelo foi conectada ao mecanismo de decoder, para que a anotação fosse gerada. A última camada de cada bloco do modelo VGG-16 fornece como saída um mapa de recursos, que é utilizado para concatenação com uma camada do caminho de subida. O terceiro modelo utiliza uma rede MobileNetV2, implementada em [82], para extrair características, assim como o modelo anterior camadas foram selecionadas para fornecer mapas de recursos à concatenação efetuada ao decoder para recuperar dados de localização. A Tabela 4.3 apresenta os parâmetros que foram utilizados para testes e seleção do melhor modelo.

Tabela 4.3: Parâmetros utilizados para comparação dos modelos.

Parâmetro	**Valor**
Otimizador	Adam
Função de perda	Binary cross entropy
Taxa de aprendizagem	0,001
Épocas	40
Tamanho do dataset de treino	18368
Tamanho do dataset de testes	4480

A mesma base de imagens foi utilizada para cada um dos modelos. As arquiteturas que utilizam o VGG-16 e o MobileNetV2 são pré-treinadas no dataset ImageNet e têm como entrada o formato 224x224x3, pois tais modelos possuem limitação da dimensão de entrada. O modelo que não utiliza outra rede como extrator possui entrada de

256x256x3. Para treinamento, os mesmo parâmetros de passos por época foram atribuídos aos modelos que utilizam VGG-16 e MobileNetV2, pois estes modelos necessitam menos processamento, uma vez que a resolução de entrada é menor e os pesos são iniciados a partir do ImageNet, desta forma, neste caso um *batch* de 28 imagens foi utilizado, já o modelo Unet modificado utiliza-se um *batch* de 20 imagens. O cálculo para atribuição do número de passos por época é realizado pela razão entre a quantidade de amostras do dataset e o tamanho do *batch*, sendo assim, para treinamento foram utilizados 656 passos por época no treinamento e 160 passos por época na etapa de testes. Para o modelo Unet modificado foram utilizados 918 passos por época para treinamento e 224 passos para etapa de testes, devido a necessidade de *batch* menor do que os demais modelos.

Embora o modelo com a MobileNetV2 tenha obtido valor de acurácia 97,69 na base de teste e 99,10% nos treinos, o VGG-16 como extrator de característica apresentou maior desempenho nos testes, com 97,92% de acurácia na validação e 99,19% em teste. Desta forma, o modelo com maior taxa de acurácia em imagens inéditas a rede é selecionado, pois mostra maior aptidão de generalização. O modelo U-net modificado obteve uma acurácia relativamente baixa em relação aos demais modelos, com acurácia de 96,93%, este valor pode estar relacionado ao treinamento do zero que é realizado neste modelo, diferentemente dos demais que utilizam um modelo pré-treinado na base ImageNet como encoder. Os resultados de acurácia obtidos nos testes podem ser visto na Figura 4.4.

4.3.1 Arquitetura Proposta

Assim como em [83], uma rede neural VGG é utilizada como encoder, contudo o modelo utilizado é semelhante ao implementado por [84] e [85], pelo qual se utiliza uma versão VGG-16 para o caminho de contração. Para que seja possível a utilização de tal modelo para *downsampling*, as camadas totalmente conectadas são retiradas, tornando-o uma FCN. O modelo proposto para segmentação utiliza uma arquitetura encoder-decoder, sendo o caminho de contração composto por um modelo VGG-16 e o caminho de expansão composto por uma sequência de camadas de convolução, convolução transposta e

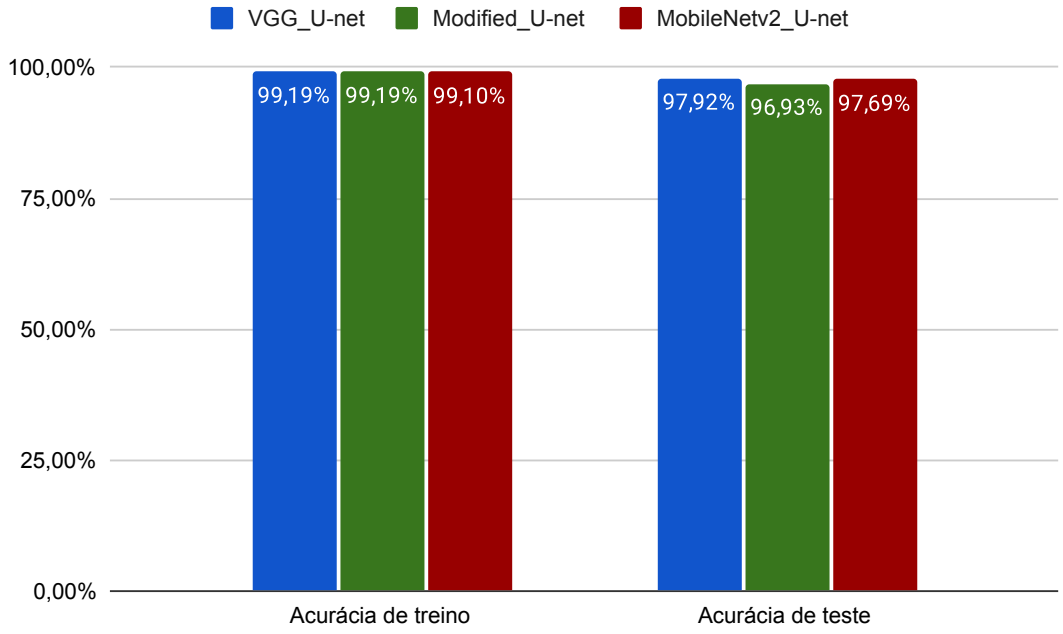

Figura 4.4: Resultados dos testes realizados nos modelos de segmentação.

concatenações, semelhante a estrutura utilizada na rede U-net. Como saída um mapa de probabilidades é fornecido por meio da função de ativação sigmoid (Equação 4.1) disponibilizada pela biblioteca Keras.

$$f(x) = \frac{1}{1 + \exp(-x)} \tag{4.1}$$

em que,

x = tensor de entrada

A partir de um tensor de entrada x a função sigmoid retorna valores entre 0 e 1, que são as probabilidades de cada pixel pertencer a uma classe ou a outra. Para que as saídas sejam atribuídas de forma binária, um valor de *threshold* é atribuído, convertendo todos os valores acima deste valor para 1 e os valores abaixo em 0. Desta forma, como saída é fornecida uma máscara de 224x224x1, com a zona da lesão destacada em pixeis brancos e o fundo em pixeis pretos. A Figura 4.5 apresenta a arquitetura do modelo de

segmentação.

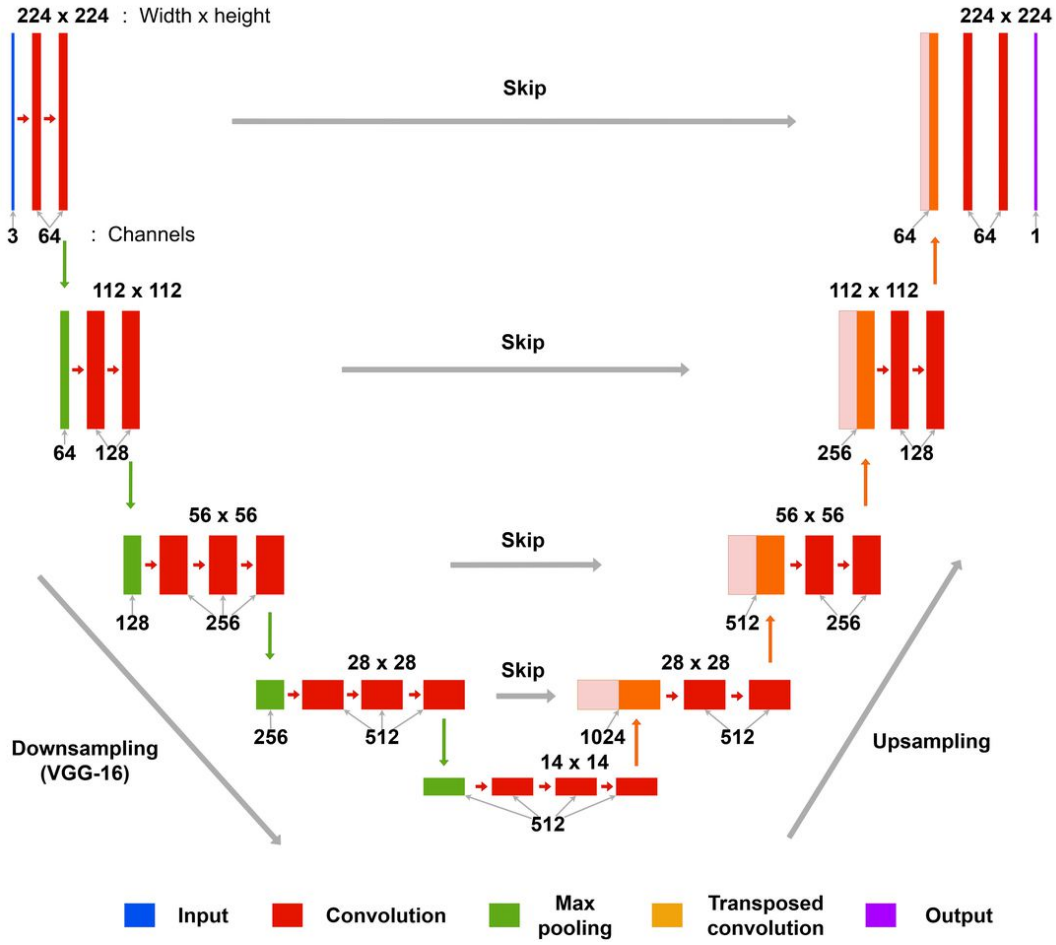

Figura 4.5: Arquitetura do modelo de segmentação proposto.

O bloco azul representa a camada de entrada, possuindo dimensão de 224x224 e 3 canais representando o espaço de cores RGB. Os números acima de cada caixa representam a dimensão (largura x altura) de cada camada e os números localizados abaixo de cada caixa referem-se a quantidade de canais que cada camada possui. As flechas de skip representam a concatenação entre as camadas do *downsampling* e do *upsampling*, utilizadas para recuperar características de localização, assim como na arquitetura U-net. As caixas verdes representam as camadas pooling, utilizadas para reduzir a dimensão das imagens por meio da operação de max pooling. Caixas em vermelho retratam operações de convolução e caixas laranjas operações de convolução transposta, utilizada para expandir a

dimensão da imagem. Por fim a caixa roxa representa o mapeamento final em pixels, que gera a máscara da imagem fornecida, possuindo a mesma dimensão de 224x224 da entrada e 1 canal de saída.

4.3.2 Implementação

O modelo foi implementado utilizando a linguagem de programação Python 3, por meio das bibliotecas de código aberto TensorFlow e Keras. Para construção do encoder o modelo VGG-16 disponível na biblioteca Keras foi utilizado, pré-treinado na base ImageNet e sem as camadas totalmente conectadas.

O encoder da arquitetura é composto totalmente pelo modelo VGG-16, o qual possui 5 blocos, sendo os 2 primeiros compostos por 2 camadas de convolução e 1 de pooling, e os últimos 3 blocos compostos por uma sequência de 3 camadas de convolução e 1 de pooling. Todas as camadas de convolução possuem kernel 3x3, stride 1, padding same e são seguidas de uma função ReLU. As camadas de pooling utilizam a função max pooling e possuem um kernel de 2x2 e stride 2x2. As saídas das últimas camadas de convolução de cada bloco foram selecionadas, exceto a do último bloco, para posterior concatenação com os mapas de recursos correspondentes do caminho de expansão.

A última camada de convolução do encoder foi conectada ao caminho de expansão, composto por um sequência de blocos com 1 convolução transposta e 2 convoluções com Batch Normalization (BN) e ReLU. Após cada convolução transposta o mapa de recursos expandido é concatenado ao mapa de recursos correspondente do caminho de contração. Após cada concatenação uma camada de dropout é aplicada para evitar o excesso de ajustes na rede. Todas as camadas de convolução possuem kernel de 3x3, stride 1 e padding same. As camadas de convolução transposta utilizam kernel de 3x3, stride 2x2 e padding same. A Tabela 4.4 apresenta as características de cada camada.

O caminho de contração possui 13 camadas de convolução que são responsáveis por reduzir a dimensão da imagem e ampliar a quantidade de mapas de recursos, tal caminho assim como na arquitetura U-net fornece características semânticas da área alvo. Por

Tabela 4.4: Relação de camadas do modelo proposto.

Arquitetura do modelo proposto

Contracting path

L. N.	Layers in contracting path	Filter size	Output Shape
L1	Input		(224,224,3)
L2	Convolution + ReLU	3 x 3	(224,224,64)
L3	Convolution + ReLU	3 x 3	(224,224,64)
L4	Max pooling	2 x 2	(112,112,64)
L5	Convolution + ReLU	3 x 3	(112,112,128)
L6	Convolution + ReLU	3 x 3	(112,112,128)
L7	Max pooling	2 x 2	(56,56,128)
L8	Convolution + ReLU	3 x 3	(56,56,256)
L9	Convolution + ReLU	3 x 3	(56,56,256)
L10	Convolution + ReLU	3 x 3	(56,56,256)
L11	Max pooling	2 x 2	(28,28,256)
L12	Convolution + ReLU	3 x 3	(28,28,512)
L13	Convolution + ReLU	3 x 3	(28,28,512)
L14	Convolution + ReLU	3 x 3	(28,28,512)
L15	Max pooling	2 x 2	(14,14,512)
L16	Convolution + ReLU	3 x 3	(14,14,512)
L17	Convolution + ReLU	3 x 3	(14,14,512)
L18	Convolution + ReLU	3 x 3	(14,14,512)

Expanding path

L. N.	Layers in contracting path	Filter size	Output Shape
L1	Transposed convolution	3 x 3	(28,28,512)
L2	Concatenate + dropout	3 x 3	(28,28,1024)
L3	Convolution + BN + ReLU	3 x 3	(28,28,512)
L4	Convolution + BN + ReLU	3 x 3	(28,28,512)
L5	Transposed convolution	3 x 3	(56,56,256)
L6	Concatenate + dropout	3 x 3	(56,56,512)
L7	Convolution + BN + ReLU	3 x 3	(56,56,256)
L8	Convolution + BN + ReLU	3 x 3	(56,56,256)
L9	Transposed convolution	3 x 3	(112,112,128)
L10	Concatenate + dropout	3 x 3	(112,112,256)
L11	Convolution + BN + ReLU	3 x 3	(112,112,128)
L12	Convolution + BN + ReLU	3 x 3	(112,112,128)
L13	Transposed convolution	3 x 3	(224,224,64)
L14	Concatenate + dropout	3 x 3	(224,224,128)
L15	Convolution + BN + ReLU	3 x 3	(224,224,64)
L16	Convolution + BN + ReLU	3 x 3	(224,224,64)
L17	Convolution + sigmoid	1 x 1	(224,224,1)

outro lado, o caminho de expansão é composto por 13 camadas convolucionais, sendo 4 delas convoluções transpostas, responsáveis por ampliar a resolução dos mapas de recursos. O caminho de expansão é responsável por recuperar a posição espacial da lesão detectada e projetá-la em dimensões maiores. Ao total o modelo possui 26 camadas convolucionais, com 28.029.377 parâmetros treináveis.

4.3.3 Treinamento

O treinamento do modelo foi realizado utilizando um processador AMD EPYC 7351 16-Core Processor com 8 núcleos de processamento e uma frequência de 2.4GHz, 16GB de memória RAM e uma GPU nvidia geforce RTX 2080 TI com 11GB de memória. A tecnologia CUDA versão 10.2 foi utilizada para extrair o máximo desempenho da GPU, utilizando o conceito de paralelismo por meio da execução de diversos núcleos simultaneamente.

Para desenvolvimento, compilação e treino do modelo, foi utilizada a biblioteca TensorFlow versão 2.2.0, que fornece suporte a tecnologia CUDA. A base de dados utilizada possui 18368 imagens para treino e 4480 para testes, todas com suas respectivas anotações binárias. Como pré processamento foram realizadas operações de redimensionamento, para garantir que as entradas tenham as dimensões necessárias e normalização das imagens. O dataset é então reordenado aleatoriamente e lotes de imagens são criados.

A função de perda binary cross-entropy foi utilizada, pois o problema se trata de uma classificação em pixel binária, ou seja, lesão ou não lesão. Como optimizador a função Adam foi adotada com taxa de aprendizado de 0,001. O modelo foi treinado por 40 épocas e a quantidade de passos por época é calculada pela razão entre o tamanho da base de treino e o tamanho de cada lote, neste caso 656 passos por época, para validação foram utilizados 160 passos.

Para acompanhamento e seleção dos melhores parâmetros, bem como análise de resultados, foi utilizada a biblioteca TensorBoard, que fornece dados essenciais à etapa de estruturação e treinamento do modelo.

4.4 Classificação

Neste secção serão apresentadas todas as abordagens adotadas para seleção da melhor estratégia para classificação binária de psoríase ou não psoríase.

O módulo de classificação é utilizado para efetivamente gerar uma predição ao utilizador final, retornando a possibilidade da imagem submetida conter ou não psoríase. Para isto, três modelos de CNN foram utilizados, todos os modelos estão contidos na biblioteca Keras e utilizam pesos iniciados a partir da base ImageNet. A fim de comparação e busca pelo melhor desempenho foram selecionados os modelos MobileNetV2, Xception [86] e InceptionResnetV2 [87]. Desta forma, é possível comparar a performance entre um modelo pequeno, leve e mais rápido como o MobileNetV2, um modelo maior e mais preciso como o InceptionResnetV2 e um modelo intermediário que provê um bom nível de precisão assim como o modelo Xception. A partir desta comparação é possível saber a viabilidade de integrar um modelo pequeno assim como o MobileNetV2 diretamente ao dispositivo móvel a partir do TensorFlow Lite, ou a necessidade de criação de um servidor backend para hospedar um modelo mais robusto e pesado como o InceptionResnetV2.

4.4.1 Arquitetura dos modelos

Todos os 3 modelos citados anteriormente: InceptionResnetV2, MobileNetV2 e Xception, utilizam suas estruturas originais, com exceção da camada totalmente conectada, que foi modificada para atender os requisitos da classificação binária em questão. Desta forma, modifica-se a dimensão de entrada para 224x224x3 para permitir que as imagens de saída do modelo de segmentação sejam utilizadas nos modelos de classificação e adiciona-se uma nova topologia de camada totalmente conectada para realizar a saída binária.

Como citado anteriormente os modelos utilizam pesos inicializados a partir da base de dados ImageNet, que são totalmente atualizados durante o processo de treinamento do modelo. Técnicas de *fine tuning* foram utilizadas, porém não aumentaram o desempenho dos modelos para esta abordagem.

Ao final das camadas de convolução e pooling os modelos aplicam um pooling global,

utilizado para calcular a média de cada mapa de recursos da camada anterior e produzir dados estruturados para realização da classificação final. Nos modelos MobileNetV2 e Xception estes dados são utilizados diretamente como entrada dos 2 neurônios de saída com ativação a partir da função SoftMax, por outro lado no modelo InceptionResnetV2 uma camada com 512 neurônios ativados pela função sigmoid e uma camada de dropout com uma taxa de 40% de perda é utilizada antes dos 2 neurônios de saída, tais camadas intermediárias foram adicionadas pois o modelo InceptionResnetV2 possui uma topologia maior que as outras redes, o que permitiu o incremento da estrutura totalmente conectada para prevenir uma possível falta de parâmetros para classificação. A Tabela 4.5 apresenta um resumo das 3 arquiteturas utilizadas.

Tabela 4.5: Características dos modelos utilizados [88].

Modelo	**Tamanho**	**Top 5 acurácia**	**Parâmetros**	**Profundidade**
MobileNetV2	14 MB	0.901	3,538,984	88
Xception	88 MB	0.945	22,910,480	126
InceptionResnetV2	215 MB	0.953	55,873,736	572

Estes dados foram retirados diretamente da plataforma da biblioteca Keras [88] e apresentam as diferentes características de cada modelo. Os dados de acurácia foram baseados em testes realizados na base de validação do ImageNet. Os dados de profundidade referem-se a profundidade da topologia de cada modelo, nesta contagem incluem-se as camadas de normalização de lotes, camadas ativação, etc [88].

4.4.2 Treinamento

Para o treinamento dos modelos de classificação foi utilizado o mesmo processador AMD EPYC 7351 usado no modelo de segmentação, uma memória RAM de 16GB e uma GPU nvidia TITAN V com 12GB de memória. As bibliotecas TensorFlow e Keras juntamente com a tecnologia CUDA 10.2 foram utilizadas para ajustes de parâmetros e realização do treinamento dos modelos. Como função de perda foi utilizada a binary cross-entropy, o optimizador selecionado foi o Adam com uma taxa de aprendizado de 0,0001. Os modelos

foram treinados por 30 épocas, sendo a quantidade de passos por época calculada de acordo com a quantidade de amostras do dataset dividido pelo tamanho do lote criado.

4.4.3 Psoríase x Pele Saudável

Em uma primeira abordagem um dataset com imagens de psoríase e imagens de pele sem lesões é utilizado, os resultados foram satisfatórios, pois as características de cada classe eram consideravelmente diversas, oque tornava fácil a predição pelo modelo. A Tabela 4.6 apresenta os resultados de acurácia obtidos em cada um dos modelos testados.

Tabela 4.6: Resultados do dataset com psoríase e pele saudável.

Modelo	**Acurácia**	**Perda**
MobileNetV2	1.0	3.3751e-07
Xception	1.0	1,4156 e-07
InceptionResnetV2	1.0	1,2995 e-05

Após obtenção de um resultado satisfatório, o foco é criar uma base de imagens que contenha imagens de psoríase em uma classe e imagens de outras lesões dermatológicas bem como imagens de pele sem lesões em outra classe. Assim é possível ensinar ao modelo o que é psoríase e o que não é, como queimaduras, picadas de insetos, outros tipos de lesões dermatológicas ou até mesmo tatuagens. A classe de outras lesões dermatológicas é composta por lesões como granuloma anular, doença de grover, picada de inseto, cicatriz queloide, Lúpus eritematoso sistêmico, dentre outras. Tais imagens foram selecionadas aleatoriamente das mesmas plataformas citadas na tabela 4.1.

Esta nova abordagem é capaz de realizar uma maior generalização de casos, uma vez que permite distinguir não somente psoríase e pele saudável, mas também diferenciar outras lesões de pele e a doença psoríase em específico.

4.4.4 Tipos de Base de Imagens

Para a realização dos testes foram utilizados 3 tipos de datasets, um formado pelas imagens originais, um contido somente das máscaras binárias das imagens e o último formado por

imagens cortadas, onde as lesões são destacadas e todo o restante que não se enquadra na classe lesão é apresentado na cor preta. Para criação de tais datasets o módulo de segmentação foi utilizado para geração das máscaras e uma posterior operação de matrizes foi aplicada para realizar o corte das regiões de lesão das imagens. Estes datasets foram criados para testar a eficiência dos modelos em imagens que não contenham o fundo, ou seja, objetos ou ambientes que não fazem parte do foco da classificação, por outro lado o dataset que utiliza somente as máscaras binárias foi utilizado para saber se o modelo seria capaz de classificar corretamente as imagens utilizando somente os dados de forma, tamanho e localização. A Tabela 4.7 apresenta os resultados obtidos nos testes realizados.

Tabela 4.7: Resultados dos testes com tipos de datasets diferentes.

Dataset \Model	**MobileNetV2**	**Xception**	**InceptionResnetV2**
Original	acc: 0,875 loss: 0,6330	acc: 0,875 loss: 0,4587	acc: 0,9 loss: 0,56555
Mask	acc: 0,7 loss: 0,666	acc: 0,7500 loss: 0,8622	acc: 0,8 loss: 0,71155
Cropped	acc: 0,85 loss: 0,2933	acc: 0,8500 loss: 1,0534	acc: 0,75 loss: 0,6693

A partir da análise dos dados obtidos dos testes é possível observar que embora o modelo MobileNetV2 com o dataset cropped tenha tido uma taxa de perda menor do que os outros datasets, o dataset original teve uma melhor média de desempenho em relação a acurácia e perda dentre os 3 datasets nos 3 modelos, ou seja, a segmentação proposta para este caso não se torna tão efetiva. Desta forma, é selecionado o dataset de imagens originais como base da pesquisa.

4.5 Bases de imagens e aumentos de dados

É possível observar que os resultados não foram satisfatórios após a adição das imagens de outras lesões na classe de imagens de não psoríase, tal fato provavelmente ocorre pela pouca quantidade de amostras e pela grande semelhança entre as lesões e a psoríase, o que faz com que o modelo não aprenda característica necessárias para distinguir entre as

classes corretamente.

Para testar a influência do número de imagens no dataset e das operações utilizadas no data augmentation nestes resultados uma nova etapa de teste foi realizada, pelo qual um incremento no número de imagens foi aplicado elevando o total de imagens de 204 para 752, sendo 376 de psoríase e 376 de não psoríase, destas, 188 imagens de pele sem lesões e 188 imagens de pele de outras lesões. Algumas das demais lesões utilizadas são: vitiligo, Eritema anular centrífugo de Darier, Dermatite herpetiforme de Duhring, Carcinoma basocelular nódulo-cístico, Acne conglobata, dentre outras. No segmento de data augmentation as operações de alteração do espaço de cores foram retiradas e uma operação de zoom foi adicionada, outra modificação foi a utilização de probabilidades, diferente da abordagem anterior as operações são aplicadas em conjunto em uma mesma imagem de acordo com uma probabilidade de ocorrência, gerando imagens que podem ser transformadas por mais de 1 operação. A Tabela 4.8 apresenta as operações, as proporções e as probabilidades de ocorrência de cada transformação.

Tabela 4.8: Novas operações de data augmentation utilizadas.

Operação	**Proporção**	**Probabilidade**
inversão horizontal	1	70%
inversão vertical	1	70%
rotação	90º, 180º, 270º	70%
rotação	25º, -25º	70%
zoom aleatório	0.6	70%

Nesta nova seleção de imagens 80% (602 imagens) foram utilizadas para treinamento e 20% (150 imagens) para teste. A partir desta nova abordagem de data augmentation foram produzidas 30.000 imagens para treino, sendo 15.000 para cada classe. Tais imagens foram utilizadas no treinamento dos 3 modelos para obtenção de um resultado preliminar do melhor desempenho entre as arquiteturas. Como pode ser visto na Tabela 4.9 um grande avanço de desempenho foi obtido utilizando ambas as técnicas de *data augmentation*, comprovando a influência da quantidade de amostras do dataset no desempenho dos modelos.

Tabela 4.9: Resultados dos testes com novo dataset e novo *data augmentation*.

Dataset \Model	**MobileNetV2**	**Xception**	**InceptionResnetV2**
Antigo DT e antigo DA	acc: 0,9250 loss: 0,3464	acc: 0,85 loss: 0,6371	acc: 0,9 loss: 0,5022
Antigo DT e novo DA	acc: 0,9 loss: 0,5190	acc: 0,925 loss: 0,2643	acc: 0,9250 loss: 0,2376
Novo DT e antigo DA	acc: 0,98 loss: 0,1412	acc: 0,98 loss: 0,1054	acc: 0,9467 loss: 0,1694
Novo DT e novo DA	acc: 0,9733 loss: 0,0723	acc: 0,9666 loss: 0,1266	acc: 0,9466 loss: 0,1404

O valores destacados em verde e vermelho correspondem ao melhor e pior resultado respectivamente, obtidos em relação a maior acurácia e menor taxa de perda neste teste. Sobre o data augmentation é possível observar que a mudança das operações afetou significamente o resultado dos modelos, em maioria positivamente. Desta forma, selecionou-se o novo dataset com 752 imagens e a nova abordagem de data augmentation, pois obtiveram maior acurácia e menor taxa de perda. As arquiteturas e parâmetros utilizados no teste da primeira linha da tabela 4.9 são os mesmo da tabela 4.7, com alteração apenas da taxa de aprendizado, de 0,001 para 0,0001.

4.6 Pré-processamento

Após a definição do modelo, do tipo do dataset e da técnica de *data augmentation* que será utilizada, uma nova técnica é proposta como possível fator no aumento de desempenho do modelo final. A proposta é utilizar o método CLAHE [89] para analisar a influência nos resultados. A expectativa neste teste é que após a otimização do contraste das imagens de entrada o modelo consiga identificar mais facilmente as regiões avermelhadas das lesões e as característica específicas de cada classe, elevando sua taxa de precisão.

A aplicação do CLAHE foi construída por meio da biblioteca de visão computacional OpenCV versão 4.3.0. O método CLAHE em geral é aplicado em imagens em escala de cinza, desta forma, houve a necessidade de conversão do espaço de cores das imagens para CIELAB, pois assim é possível aplicar o CLAHE no canal L que armazena valores de

luminosidade e manter os canais AB que representam as informações de cor, em seguida a imagem é convertida em RGB novamente. Nos parâmetros do CLAHE foram utilizados os valores 8x8 para o tamanho da grade, gerando 24 sub-regiões, e um limite de *clip* de valor 2. A Figura 4.6 apresenta um exemplo da aplicação do CLAHE em uma imagem da classe psoríase e uma da classe não psoríase.

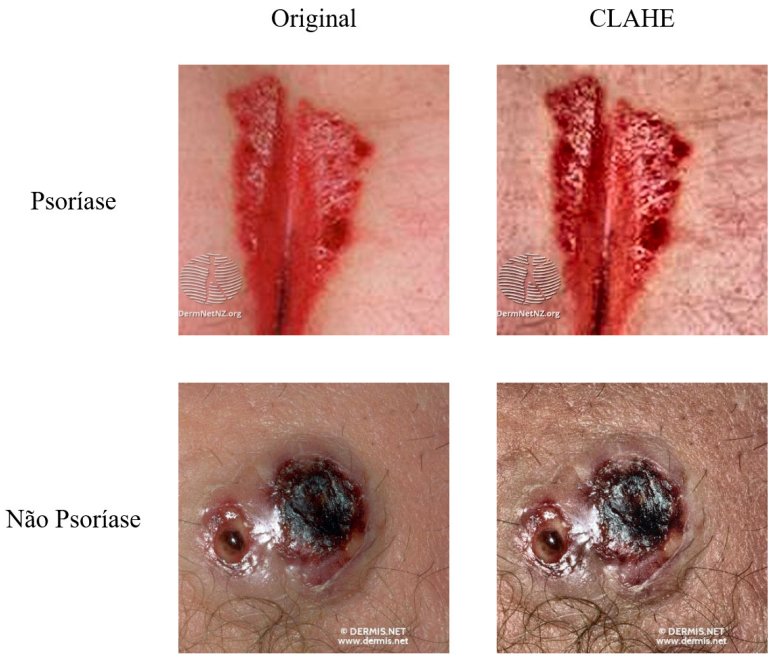

Figura 4.6: Resultado das imagens pós aplicação do método CLAHE.

As figuras da esquerda são as imagens originais e as da direita são as mesma imagens após a aplicação do método CLAHE. É possível observar que o CLAHE aumenta o contraste significativamente e faz com que as lesões se destaquem do restante das imagens, porém observou-se que em alguns casos a aplicação deste método fez com que imagens de psoríase e imagens de outras lesões ficassem mais semelhantes, podendo confundir o modelo.

4.7 Aplicação móvel e Web API

Para realizar a aplicação efetiva dos modelos propostos nas seções anteriores, foi desenvolvida uma arquitetura cliente servidor para realização da análise de imagens de pele. Esta topologia é composta por uma aplicação móvel, camada cliente, e uma web API, camada servidor. A função da aplicação móvel é essencialmente capturar a imagem utilizando a câmera do dispositivo, enviar a imagem capturada ao servidor e apresentar os resultados. Já o servidor têm como tarefa receber uma imagem e analisá-la, utilizando os modelos de classificação e segmentação. A Figura 4.7 apresenta este fluxo de comunicação entre o cliente e o servidor.

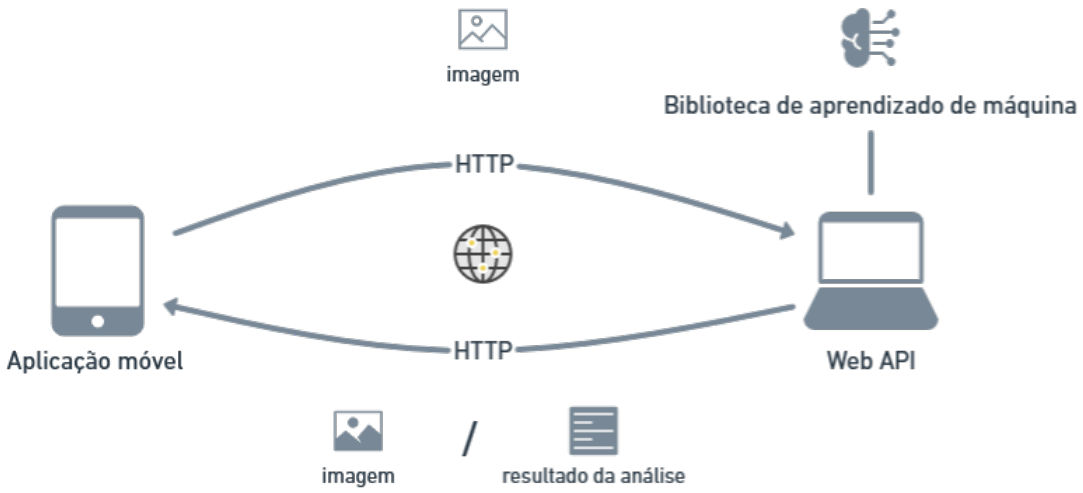

Figura 4.7: Arquitetura cliente servidor utilizada para classificação.

Como é possível observar na figura, para comunicação entre o cliente e o servidor é utilizado o protocolo Hypertext Transfer Protocol (HTTP), através de dados em formato JavaScript Object Notation (JSON). Tal protocolo permite que o aplicativo móvel envie imagens e dados via rede e vice-versa.

4.7.1 Aplicação Móvel

A aplicação móvel foi desenvolvida para a plataforma Android por meio do Android Software Development Kit (SDK), fornecido pelo Google, e a linguagem de programação JAVA. Este kit de ferramentas fornece diversos recursos e classes necessários para desenvolvimento de aplicações móveis.

Para o acesso a câmera foi utilizada a classe Intent, que utiliza outra aplicação de câmera nativa do próprio dispositivo para capturar imagens, basicamente esta classe é chamada com um conjunto de instruções, ela então seleciona a aplicação elegível à tarefa designada, que realiza a tarefa e retorna os dados à aplicação solicitante, neste caso a imagem capturada. A vantagem do uso de uma Intent para a captura da imagem é a possibilidade de utilização de todos os recursos nativos de configuração da câmera do dispositivo, como foco, aspecto, etc.

Em seguida, a imagem é enviada a uma segunda Activity responsável pelo corte da imagem, a região de corte é limitada em uma região quadrada para que não haja distorções no momento do redimensionamento, necessário para entrada nas redes neurais, esta Activity de corte permite ainda a aplicação de zoom e rotações na imagem para melhor enquadramento da região de interesse.

Uma vez que a imagem esteja capturada e recortada, encontra-se em uma dimensão quadrada e esta pronta para envio ao servidor. Desta forma, a imagem é codificada para o formato base64 e inserida em uma mensagem JSON. Para a comunicação com o servidor foi utilizado a biblioteca HTTP Volley, que de maneira integrada ao Android permite chamadas HTTP, desta forma, uma requisição do tipo POST é feita ao IP do servidor, enviando uma mensagem em formato JSON com a imagem. Como retorno, este método recebe um JSON com uma imagem e 2 campos de texto, um com o resultado da análise, psoríase ou não psoríase, e o segundo recebe a precisão desta classificação, estes dados são então apresentado ao utilizador como resultado da análise. A aplicação é suportada por dispositivos Android versão 5.0 (API 21) em diante e tem como versão alvo a 10.0 (API 29), segundo a própria plataforma de desenvolvimento Android Studio esta configuração

é suportada por aproximadamente 94,1% dos dispositivos.

4.7.2 Web API

Para desenvolvimento do servidor web foi utilizada a linguagem Python versão 3.6.10 e o framework Flask versão 1.0.3. Estas tecnologias foram escolhidas pois a versão do Python é a mesma utilizada no treinamento das redes neurais, oque facilita a integração e evita conversões dos modelos, bem como pela maior facilidade de operar sobre imagens que o Python proporciona.

Assim como descrito anteriormente, uma conversão é feita de base64 para imagem, esta imagem é submetida a um processo de redimensionamento e passa a ter tamanho 224x224, possibilitando sua utilização como entrada das redes neurais. Em seguida, a imagem é convertida em uma matriz numpy e os valores dos pixeis são dimensionados no *range* de 0 a 1 e transformados para o tipo float32 para alimentar os modelos.

O primeira modelo é alimentado com a imagem e a classifica como psoríase ou não psoríase, retornando a acurácia desta classificação, se o resultado for não psoríase uma mensagem JSON é estruturada e retorna o resultado e a acurácia da classificação, por outro lado, se o resultado da classificação for psoríase, o modelo de segmentação é alimentado com a imagem de entrada e cria uma anotação, pintando cada pixel de acordo com a sua probabilidade de conter psoríase. Ao final temos uma matriz numpy de probabilidades, para criar uma anotação normalizada e retirar ruídos um *threshold* de decisão é definido com o valor 0.8, ou seja, pixeis com intensidade iguais ou maiores que o *threshold* são definidos como 1 e valores menores são definidos como 0.4. Uma multiplicação de matriz é realizada entre a imagem original e a anotação gerada, produzindo uma nova imagem, pela qual as regiões onde o modelo classificou como psoríase são apresentadas com sua intensidade original e as regiões que não são de interesse são apresentada com menor intensidade, destacando assim as regiões de possível lesão. Esta imagem é convertida novamente ao formato base64 e adicionada ao JSON de resposta.

Capítulo 5

Testes e Discussão

A seguir são apresentados de maneira detalhada os testes e resultados de cada etapa do projeto, são elas: módulo de segmentação, módulo de classificação e aplicação móvel. No capítulo 4 foram apresentados resultados preliminares, utilizados no testes para seleção das melhores arquiteturas, parâmetros e técnicas. Os resultados apresentados neste capítulo são feitos com base nas arquiteturas e parâmetros que apresentaram maior desempenho.

5.1 Modelos de classificação

Para avaliação dos modelos de classificação foram utilizadas as métricas: acurácia, taxa de perda, matriz de confusão, sensibilidade, especificidade e precisão, por meio da biblioteca de aprendizado de máquina Scikit Learn [90] versão 0.23.1. Desta forma, por meio dos valores de Verdadeiro Positivo (VP), Verdadeiro Negativo (VN), Falso Positivo (FP) e Falso Negativo (FN) obtidos da matriz de confusão, é possível calcular as métricas citadas e obter uma medida de desempenho para cada modelo avaliado. A Tabela 5.1 apresenta uma descrição de cada uma das medidas.

Valores de acurácia (Equação 5.1) correspondem a fração das classificações corretas em todas as classificações realizadas [91].

Tabela 5.1: Valores utilizados no cálculo das métricas aplicadas.

Expressão	Descrição
VP	São amostras que são da classe positiva e o modelo classifica como pertencente a classe positiva.
FP	São amostras que não são da classe positiva e o modelo classifica como pertencente a classe positiva.
VN	São amostras que são da classe negativa e o modelo classifica como pertencente a classe negativa.
FN	São amostras que não são da classe negativa e o modelo classifica como pertencente a classe negativa.

$$Acurácia = \frac{(VP + VN)}{(VP + FP + FN + VN)} \tag{5.1}$$

em que,

VP = Verdadeiro Positivo;

FP = Falso Positivo;

VN = Verdadeiro Negativo;

FN = Falso Negativo.

A taxa de perda refere-se ao quão longe as classificações estão dos rótulos esperados, para determinação deste valor utiliza uma função de perda [92].

A matriz de confusão é utilizada para avaliar o desempenho do classificador, por meio de cada par de classes <c1, c2> esta métrica verifica qual a proporção de documentos c1 foram incorretamente associados a c2, a matriz de confusão pode auxiliar na identificação de oportunidades de aumento da precisão dos modelos [91].

A métrica sensibilidade (Equação 5.2) aponta a capacidade do modelo de classificar amostras positivas [93].

$$Sensibilidade = \frac{(VP)}{(VP + FN)} \tag{5.2}$$

A especificidade (Equação 5.3) é utilizada para calcular a capacidade do classificador

na predição de amostra negativas [93].

$$Especificidade = \frac{(VN)}{(FP + VN)} \tag{5.3}$$

A precisão (Equação 5.4) avalia o quão certo está o modelo na classificação de amostras positivas em todas as amostras que foram rotuladas pelo sistema como positivo [94].

$$Precisão = \frac{(VP)}{(VP + FP)} \tag{5.4}$$

Os modelos MobileNetV2, Xception e InceptionResnetV2 foram testados e avaliados com o mesmo dataset de 752 imagens, no qual 150 imagens foram utilizadas para esta etapa de validação, com as mesmas técnicas de aumento de dados e com os mesmo parâmetros de treinamento. Foram utilizadas 30 épocas no treinamento, 32 imagens por lote para treinamento, 30 imagens por lote para teste, uma taxa de aprendizado de 0,0001 e a função de perda *Binary Cross Entropy*. A Tabela 5.2 apresenta os resultados obtidos nos testes para cada modelo.

Tabela 5.2: Relação dos resultados obtidos na classificação.

Métricas	**MobileNetV2**	**Xception**	**InceptionResnetV2**
Acurácia	97.33%	96.66%	94.66%
Taxa de perda	7.23%	12.66%	14.05%
Sensibilidade	96%	96%	92%
Especificidade	98.66%	97.33%	97.33%
Precisão	98.63%	97.29%	97.18%

Como pode ser observado, o modelo MobileNetV2 apesar de possuir a menor estrutura e menor profundidade obteve maior acurácia e menor taxa de perda, outro ponto relevante é que todos os modelos tiveram maior valor de especificidade do que sensibilidade, o que sugere um maior êxito na classificação de imagens em classes negativas, neste caso imagens sem lesão. Ainda com estes dados é possível concluir que modelos maiores e mais robustos não são exatamente a melhor opção para esta abordagem, com esta base de imagens,

Tabela 5.3: Matrizes de confusão dos modelos de classificação.

MobileNetV2

	Valor previsto		
		Psoríase	Não psoríase
Valor real	Psoríase	72	3
	Não psoríase	1	74

Xception

	Valor previsto		
		Psoríase	Não psoríase
Valor real	Psoríase	72	3
	Não psoríase	2	73

InceptionResnetV2

	Valor previsto		
		Psoríase	Não psoríase
Valor real	Psoríase	69	6
	Não psoríase	2	73

possivelmente pela quantidade limitada de dados.

A seguir serão apresentadas as matrizes de confusão (Tabela 5.3), utilizadas para analisar o comportamento de cada modelo e identificar seus pontos fortes e fracos.

A partir da análise das matrizes de confusão é possível comprovar oque a sensibilidade nos apresentava, os modelos têm mais facilidade na classificação de imagens de não psoríase, em geral as redes estão errando em algumas predições de imagens de psoríase, classificando-as como não psoríase, tal fato pode se dar pela semelhança que existe entre algumas imagens de outras lesões e a psoríase em si. Contudo, em uma perspectiva geral os modelos obtiveram um resultado satisfatório, com o melhor desempenho apresentando somente 4 classificações errôneas e o pior modelo classificando 8 imagens de maneira equivocada, em um total de 150 imagens avaliadas. Para que seja possível compreender quais tipos de imagens os modelos têm dificuldade de classificar corretamente, a Figura 5.1 apresenta imagens e suas respectivas classificações do modelo MobileNetV2, que obteve melhor desempenho nos testes.

As 4 primeiras imagens da figura 5.1 referem-se as amostras de falso positivo e falso

Figura 5.1: Classificações realizadas pelo modelo MobileNetV2.

negativo apresentadas na matriz de confusão do modelo MobileNetV2. Como pode ser visto, o modelo teve dificuldade em classificar imagens de psoríase com bordas sem uma definição clara e com uma coloração não tão avermelhada, no caso de imagens de não psoríase apenas 1 imagem foi classificada como falso positivo, provavelmente por sua coloração que assemelha-se as imagens de psoríase. Nas últimas 4 imagens é possível observar que o modelo é capaz de classificar corretamente imagens de ambas as classes que sejam semelhantes, mas que tenham bordas bem definidas.

5.2 Modelos de Segmentação

Para avaliação dos modelos de segmentação são utilizadas as métricas: acurácia, índice Jaccard e pontuação DICE. Tais métricas proporcionam uma visibilidade real do desempenho dos modelos na tarefa de segmentação.

O índice Jaccard (Equação 5.5) é uma métrica para medição da similaridade ou dissimilaridade entre duas amostras de dados. Assim, para quaisquer conjuntos finitos A e B, esta métrica é definida como a razão do tamanho da intersecção sobre a união. Com valores entre 0 e 1, quanto maior este valor, maior a similaridade entre as amostras [95].

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{5.5}$$

em que,

A = Conjunto finito 1;

B = Conjunto finito 2.

De maneira similar ao índice Jaccard, o coeficiente DICE (Equação 5.6) também calcula a similaridade entre 2 quaisquer conjuntos finitos A e B. Com valores entre 0 e 1, o maior valor indica maior similaridade entre os conjuntos [95].

$$DICE(A, B) = \frac{2|A \cap B|}{|A| + |B|} \tag{5.6}$$

Tabela 5.4: Resultados dos modelos de segmentação.

Métricas	**VGG16-Unet**	**MobileNetV2-Unet**	**Modified-Unet**
Acurácia:	97.92%	97.69%	96.93%
Taxa de perda	6.51%	65.05%	8.43%
Jaccard	87.30%	86.17%	82.87%
DICE	91.13%	89.76%	88.10%

O módulo de segmentação utilizou um dataset contido por imagens de psoríase e não psoríase. A Tabela 5.4 apresenta os resultados das métricas citadas.

Para aplicação das métricas foi utilizado um valor de *threshold* de 0,8 para enquadrar cada pixel em um cenário binário, psoríase ou não psoríase. A partir da análise destes dados é possível compreender efetivamente qual é o nível de desempenho dos modelos. Embora todos os modelos tenham obtido uma acurácia relativamente elevada e semelhante, a partir da taxa de perda é possível ver que o modelo MobileNetV2 não é o mais estável para esta abordagem, por outro lado os demais modelos obtiveram uma taxa aceitável. As métricas DICE e Jaccard apresentam uma representação mais clara, da porcentagem de acerto dos modelos e mostra que o modelo VGG16-Unet possui maior capacidade de classificação correta em relação aos outros modelos. A seguir, na Figura 5.2 são apresentados exemplos de imagens segmentadas pelo modelo VGG16-Unet.

Todas as imagens possuem lesões de psoríase. Em geral, imagens que possuem lesões com bordas mais aparentes são segmentadas de maneira mais eficiente, por outro lado, imagens que possuem lesões com bordas não definidas geralmente são segmentadas com áreas de falha, geralmente estas áreas possuem lesão, porém o modelo não as classifica como lesão. Apesar de não conseguir cobrir todos os pontos de lesão em imagens que possuem múltiplos focos da doença, o modelo é capaz de segmentar a maior parte da imagem de forma correta, restando apenas pequenos pontos de lesão, geralmente os menores.

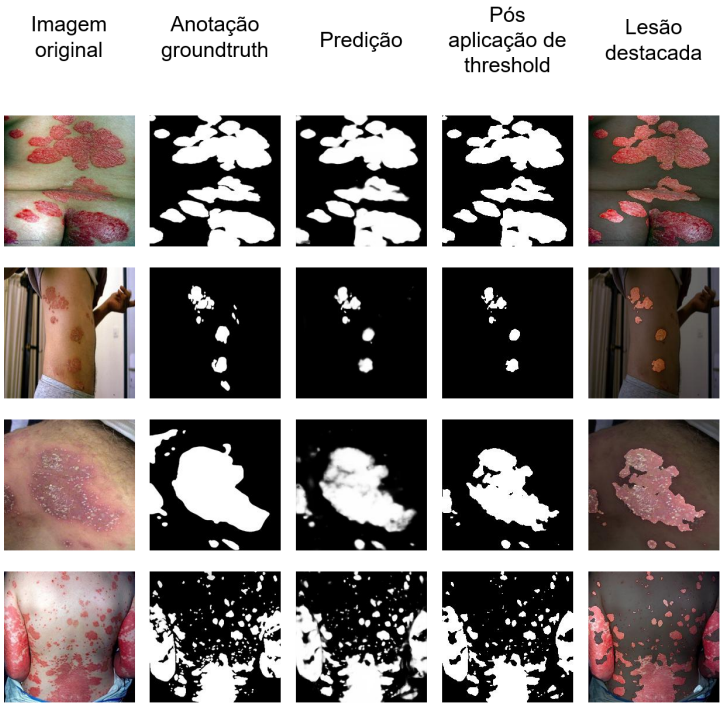

Figura 5.2: Classificações realizadas pelo modelo Vgg16-Unet.

5.3 Aplicação móvel

A aplicação móvel desenvolvida segue o planejamento e a documentação do capítulo 3. Suas interfaces são simples e intuitivas, pelas quais a partir de diversas cores, ícones, textos de instrução e diferentes tamanhos de fontes permite que o utilizador tenha uma experiência intuitiva e de fácil manuseio. Optou-se por uma interface simples, que foque na classificação e apresentação dos dados. A Figura 5.3 apresenta a interface principal (Figura 5.3(a)), interfaces de captura (Figura 5.3(b,c)) e interface de recorte (Figura 5.3(d)) .

Após o recorte, a imagem capturada é apresentada na interface principal para avaliação do utilizador e o botão de análise é habilitado (Figura 5.4 (a)). Ao clicar no botão de análise uma mensagem de carregamento é apresentada até que o servidor retorne os dados da classificação. Assim que a classificação é recebida os dados são apresentados por meio de duas mensagens, uma contendo a classificação, psoríase ou não psoríase, e a outra apresentando a probabilidade da classificação (Figura 5.4(b,c,d)). Para melhor

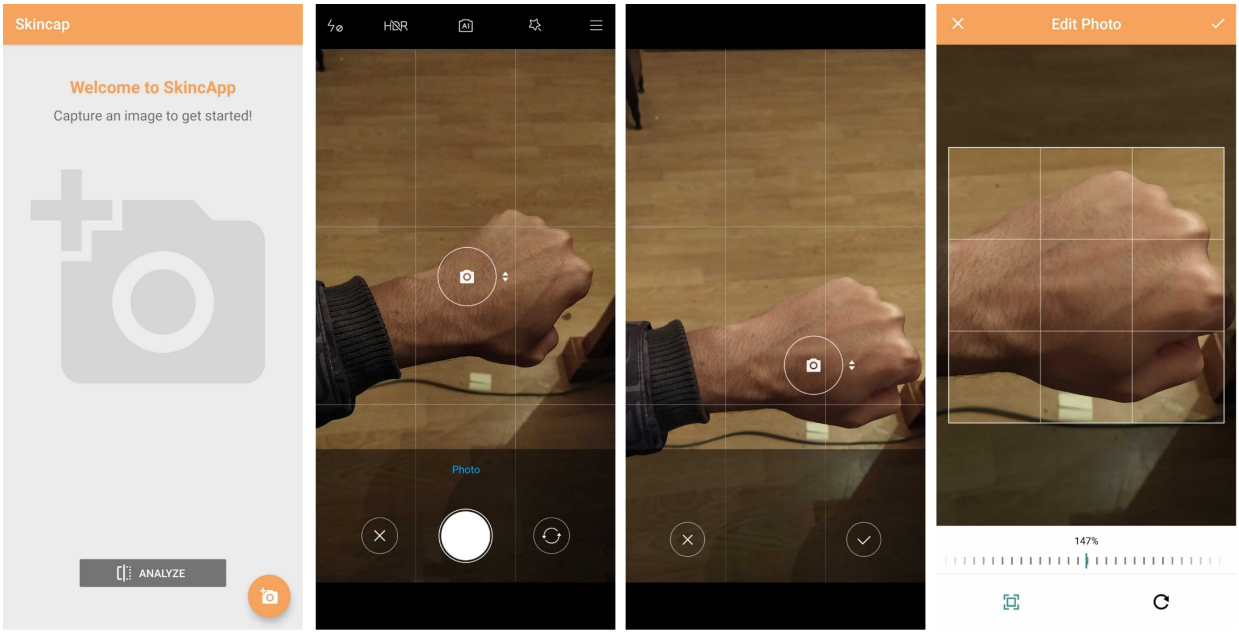

Figura 5.3: Interface principal, interfaces de captura e interface de recorte da aplicação móvel.

apresentação da informação, a probabilidade é mapeada em 5 medidas: muito baixa, baixa, média, alta e muito alta, que são definidas de acordo com a acurácia de 0% a 100% retornada pelo classificador.

Apesar de não fazer parte do objetivo principal, uma função de carregamento de imagem diretamente do armazenamento interno do dispositivo poderia ser adicionada, permitindo ao utilizador usar imagens armazenadas no dispositivo.

Como pode ser visto na Figura 5.4(c) o módulo de segmentação é utilizado para apresentar a possível área de lesão quando a classificação é psoríase, caso contrário a imagem original é apresentada.

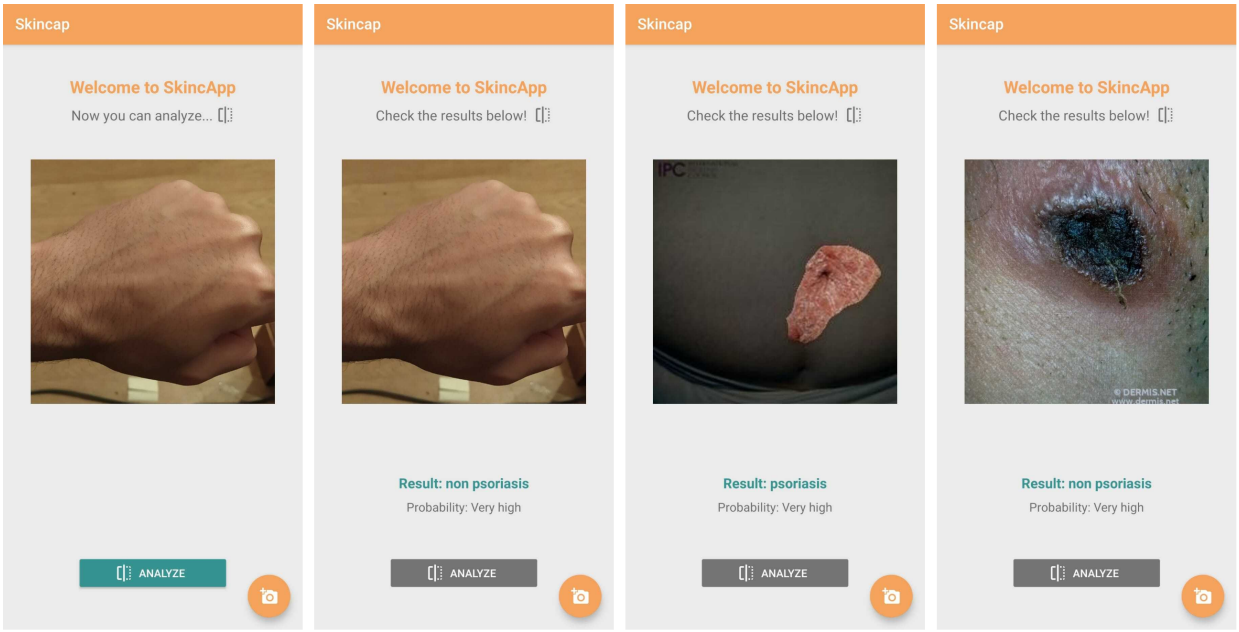

Figura 5.4: Interface de apresentação de resultados da aplicação móvel.

Capítulo 6

Conclusões

Considerando a diversidade de lesões dermatológicas existentes, diagnosticar corretamente uma doença dermatológica específica não é uma tarefa simples. Este é um dos motivos pelos quais o emprego de ferramentas computacionais tem sido cada vez mais integradas ao método de diagnóstico como uma ferramenta de auxílio ao profissional de saúde, reduzindo o tempo necessário e aumentando a precisão dos diagnósticos efetuados.

De forma a integrar um sistema com inteligência artificial a mobilidade e praticidade necessária no cotidiano atual, este trabalho baseou-se no desenvolvimento de uma aplicação móvel para auxilio ao diagnóstico da lesão psoríase por meio de redes neurais convolucionais. A aplicação desenvolvida cumpre satisfatoriamente o objetivo de classificar imagens de pele como psoríase ou não psoríase, por meio de uma interface amigável e interativa.

Para desenvolvimento do sistema de classificação, devido a dificuldade de acesso a imagens de psoríase, foram criados 2 principais datasets, onde o melhor dataset possui 752 imagens clínicas de pele humana, sendo metade de psoríase e metade de não psoríase. Um dataset para segmentação semântica também foi criado manualmente, possuindo imagens de psoríases e suas respectivas imagens Ground truth. Este dataset foi criado para comparar e validar se imagens segmentadas fariam o modelo de classificação ter maior desempenho. A eficiência dos datasets foi comprovada pelos modelos de classificação e segmentação que obtiveram um desempenho satisfatório.

O sistema de classificação é capaz de classificar um conjunto de 150 imagens inéditas com uma acurácia de 97,33%, pelas quais somente 4 imagens foram classificadas equivocadamente. No modelo de segmentação obteve-se um resultado satisfatório em relação a quantidade limitada de imagens utilizadas, o modelo apresentou uma acurácia de 97,92% com 91,13% de pontuação DICE.

Como citado anteriormente, as bases de imagens foram criadas, porém com uma quantidade limitada de amostras, desta forma para trabalhos futuros é de grande relevância a ampliação dos datasets para um melhor desempenho dos modelos de rede neural.

A integração da aplicação móvel aos módulos de classificação e segmentação fornecem ao utilizador final uma experiência simples e rápida, capaz de auxiliá-lo em um primeiro contato com uma lesão dermatológica a partir da sugestão de uma possível presença de psoríase. Tal informação é essencial para que um utilizador procure um profissional dermatologista e informe a suspeita de psoríase.

Como trabalhos futuros sugere-se as seguintes abordagens: permitir que o utilizador utilize imagens armazenadas na memória interna do dispositivo móvel; disponibilizar o uso da aplicação em modo offline, a partir da utilização de uma modelo de rede neural integrado internamente ao dispositivo, é possível oferecer a classificação das imagens sem acesso a rede e, caso haja rede, o sistema utiliza um servidor, permitindo a aplicação de métodos de processamento de imagens e segmentação; aumento da base de imagens, oque permitirá uma maior generalização do modelo de classificação; Aplicar mais doenças dermatológicas na predição e, por fim, implementar um módulo para coleta das imagens submetidas pelos utilizadores, com a respectiva autorização de cada utilizador, permitindo assim a expansão da base de dados, após um profissional da área catalogar e classificar cada imagem.

Bibliografia

[1] C. B. De Psoríase, "Guias de avaliação e tratamento Sociedade Brasileira de Dermatologia", *Rio de Janeiro: Sociedade Brasileira de Dermatologia*, 2012.

[2] T. Martins, "Dificuldades de regulação emocional em doentes com psoríase", dissertação de mestrado, Instituto Superior de Ciências da Saúde – Norte, 2011.

[3] T. Torres, R. Sales, C. Vasconcelos e M. Selores, "Psoriasis and cardiovascular disease", *Acta médica portuguesa*, vol. 26, n.º 5, pp. 601–607, 2013.

[4] A. P. Rodrigues e R. Teixeira, "Desvendando a psoríase", *Revista Brasileira de Análises Clínicas*, vol. 41, n.º 4, pp. 303–9, 2009.

[5] G. Marques Pinto e P. Filipe, "Normas de boa prática para o tratamento da psoríase em placas em idade não pediátrica com biológicos", *Revista da Sociedade Portuguesa de Dermatologia e Venereologia*, pp. 531–553, 2011.

[6] C. M. S. Mota, M. C. C. Gon e A. dos Santos Gon, "Análise comportamental de problemas de interação social de indivíduos com psoríase", *Interação em Psicologia*, vol. 13, n.º 1, 2009.

[7] S. McKenna, S. Cook, D. Whalley, L. Doward, H. Richards, C. Griffiths e D. Van Assche, "Development of the PSORIQoL, a psoriasis-specific measure of quality of life designed for use in clinical practice and trials", *British Journal of Dermatology*, vol. 149, n.º 2, pp. 323–331, 2003.

[8] G. Marques Pinto, M. Gonçalo, C. Resende e A. Pereira, "Psoríase", *Acta Médica Portuguesa*, pp. 219–245, 2001.

[9] R. C. Gonzalez e R. E. Woods, *Processamento digital de imagens.* 3ª ed. São Paulo: Pearson Educación, 2009.

[10] M. E. Vianna, "Calibração de Sistemas de Visão Computacional para Aplicação em Automação e Robótica", dissertação de mestrado, Pontifíca Universidade Católica de Minas Gerais, Belo Horizonte, 2009.

[11] M. Marengoni e S. Stringhini, "Tutorial: Introdução à visão computacional usando opencv", *Revista de Informática Teórica e Aplicada*, vol. 16, n.º 1, pp. 125–160, 2009.

[12] O. Marques Filho e H. V. Neto, *Processamento digital de imagens*. Rio de Janeiro: Brasport, 1999.

[13] E. d. Santos, "O uso de visão computacional para o controle de um manipulador robótico", trabalho de conclusão de curso, Universidade Tecnológica Federal do Paraná, 2014.

[14] H. Pedrini e W. Schwarsz, "Digital image analysis-principles algorithms and applications", *Thomson Learning, São Paulo, Brazil*, 2008.

[15] W. M. MARTINS, "Estudo de Algoritmos de Visão Computacional para Identificação e Desvio de Objetos em Tempo Real: Uma Aplicação para Quadrotores", dissertação de mestrado, Universidade Federal de Itajubá, Itajubá, 2018.

[16] M. E. Stivanello, "Inspeção industrial através de visão computacional", *Monografia (Monografia)—Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau*, p. 33, 2004.

[17] C. Solomon e T. Breckon, *Fundamentos de processamento digital de imagens: uma abordagem prática com exemplos em Matlab*, 1ª ed. Rio de Janeiro: LTC, 2013.

[18] L. C. Lulio, "Técnicas de visão computacional aplicadas ao reconhecimento de cenas naturais e locomoção autônoma em robôs agrícolas móveis", dissertação de mestrado, Universidade de São Paulo, 2011.

[19] A. H. Knob, "Aplicação do processamento de imagens digitais para análise da anisotropia da massa de grãos", dissertação de mestrado, Universidade Regional do Noroeste do Estado do Rio Grande do Sul, Ijuí, RS, 2010.

[20] F. d. C. Prodossimo, "Metodologia para a captura, detecção e normalização de imagens faciais", dissertação de mestrado, Universidade Tecnológica Federal do Paraná, 2013.

[21] C. T. Picon, I. Rossi e M. P. Ponti Jr, "Análise da classificação de imagens por descritores de cor utilizando várias resoluções", *Instituto de Ciências Matemáticas e de Computação (ICMC)-USP*, 2011.

[22] R. d. Rocha et al., "Explorando abordagens evolutivas para recuperação de imagens baseada em conteúdo", dissertação de mestrado, Universidade Tecnológica Federal do Paraná, 2016.

[23] J. H. N. de Aquino, "Extração de características de imagens para classificação da qualidade de couro caprino usando padrão binário local", dissertação de mestrado, Universidade Federal do Ceará, 2017.

[24] O. A. B. Penatti, "Estudo comparativo de descritores para recuperação de imagens por conteudo na web", dissertação de mestrado, Universidade Estadual de Campinas, Instituto de Computação, Campinas, SP, 2009.

[25] I. C. Stacheski e O. Broch Junior, "Detecção e estimativa de velocidade veicular através de processamento de imagem", trabalho de conclusão de curso, Universidade Tecnológica Federal do Paraná, 2018.

[26] S. Haykin, *Redes neurais: princípios e prática. 2001.* Bookman, Porto Alegre, Rio Grande do Sul, Brasil, 2001.

[27] A. d. P. Braga, *Redes neurais artificiais: teoria e aplicações.* Livros Técnicos e Científicos, 2000.

[28] V. P. A. Barros et al., “Avaliação do desempenho de algoritmos de retropropagação com redes neurais artificiais para a resolução de problemas não-lineares”, dissertação de mestrado, Universidade Tecnológica Federal do Paraná, 2018.

[29] K. Faceli, A. C. Lorena, J. Gama, A. C. P. d. L. Carvalho et al., “Inteligência Artificial: Uma abordagem de aprendizado de máquina”, 2011.

[30] T. M. Mitchell, *Machine learning*, 1ª ed. New York, NY, USA: McGraw-hill, 1997.

[31] P. H. H. Faber, *Previsão de valores de ações utilizando Deep Learning*, 2016.

[32] L. Deng e D. Yu, *Deep learning: methods and applications. Foundations and Trends (R) in Signal Processing, 7 (3-4), 197-387*, 2014.

[33] C. A. R. Pacheco e N. S. Pereira, “Deep Learning Conceitos e Utilização nas Diversas Áreas do Conhecimento”, *Revista Ada Lovelace*, vol. 2, pp. 34–49, 2018.

[34] M. A. Ponti e G. B. P. da Costa, “Como funciona o deep learning”, *arXiv preprint arXiv:1806.07908*, 2018.

[35] F. J. Núñez Sánchez-Agustino, “Diseño de un sistema de reconocimiento automático de matrículas de vehículos mediante una red neuronal convolucional”, dissertação de mestrado, Universitat Oberta de Catalunya, 2016.

[36] S. A. B. Block et al., “Inspeção e classificação de picos em peças estampadas de metal utilizando rede neural convolucional”, dissertação de mestrado, Universidade Tecnológica Federal do Paraná, 2018.

[37] D. Silva et al., “Uso de aprendizado de máquina para estimar esforço de execução de testes funcionais”, dissertação de mestrado, Universidade Estadual de Campinas, Faculdade de Engenharia Eletrica e de Computação, 2009.

[38] F. MATEO JIMÉNEZ, “Redes neuronales y preprocesado de variables para modelos y sensores en bioingeniería”, tese de doutoramento, 2012.

[39] S. Russel e P. Norvig, *Inteligência Artificial*, 3ª ed. Rio de Janeiro: Editora Campus, 2013.

[40] J. D. P. Massucatto, “Aplicação de conceitos de redes neurais convolucionais na classificação de imagens de folhas”, trabalho de conclusão de curso, Universidade Tecnológica Federal do Paraná, 2018.

[41] A. d. S. Oliveira et al., “Uso de técnicas de aprendizagem profunda na classificação de configurações de mão de língua de sinais”, dissertação de mestrado, Universidade Federal do Amazonas, 2019.

[42] I. R. d. S. Teixeira, “Deep learning aplicado à automação de balanças comerciais de hortifrutis”, trabalho de conclusão de curso, Universidade Federal de Mato Grosso, Campus Universitário do Araguaia, Instituto de Ciências Exatas e da Terra, Barra do Garças, MT, 2018.

[43] A. C. G. Vargas, A. Paes e C. N. Vasconcelos, “Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres”, em *Proceedings of the XXIX Conference on Graphics, Patterns and Images*, vol. 1, 2016.

[44] D. C. d. Oliveira, “Uma abordagem para detecção de pessoas em imagens de veículos aéreos não-tripulados”, dissertação de mestrado, Universidade Tecnológica Federal do Paraná, 2016.

[45] D. d. O. Preto, “Reconhecimento de placas de trânsito por meio de deep learning”, trabalho de conclusão de curso, Universidade Federal de Santa Catarina, Florianópolis, SC, 2018.

[46] G. R. Silva, “Detecção de objetos em imagens utilizando técnicas de aprendizagem profunda”, trabalho de conclusão de curso, Universidade Federal de Santa Catarina, Florianópolis, SC, 2018.

[47] H. Zhao, X. Qi, X. Shen, J. Shi e J. Jia, “Icnet for real-time semantic segmentation on high-resolution images”, em *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 405–420.

[48] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez e J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation", *arXiv preprint arXiv:1704.06857*, 2017.

[49] H. Noh, S. Hong e B. Han, "Learning deconvolution network for semantic segmentation", em *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1520–1528.

[50] Y. Li, H. Qi, J. Dai, X. Ji e Y. Wei, "Fully convolutional instance-aware semantic segmentation", em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2359–2367.

[51] J. Long, E. Shelhamer e T. Darrell, "Fully convolutional networks for semantic segmentation", em *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[52] O. Ronneberger, P. Fischer e T. Brox, "U-net: Convolutional networks for biomedical image segmentation", em *International Conference on Medical image computing and computer-assisted intervention*, Springer, Quebec City, QC, Canada, 2015, pp. 234–241.

[53] T. Falk, D. Mai, R. Bensch, Ö. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald et al., "U-Net: deep learning for cell counting, detection, and morphometry", *Nature methods*, vol. 16, n.º 1, pp. 67–70, 2019.

[54] L.-C. Chen, G. Papandreou, F. Schroff e H. Adam, "Rethinking atrous convolution for semantic image segmentation", *arXiv preprint arXiv:1706.05587*, 2017.

[55] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy e A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs", *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, n.º 4, pp. 834–848, 2017.

[56] X. Zhang, Y. Zou e W. Shi, "Dilated convolution neural network with LeakyReLU for environmental sound classification", em *2017 22nd International Conference on Digital Signal Processing (DSP)*, IEEE, 2017, pp. 1–5.

[57] S. J. Pan e Q. Yang, "A survey on transfer learning", *IEEE Transactions on knowledge and data engineering*, vol. 22, n.º 10, pp. 1345–1359, 2009.

[58] E. S. Olivas et al., *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques: Algorithms, Methods, and Techniques*. New York: IGI Global, 2010.

[59] Y. Tang, L. Peng, Q. Xu, Y. Wang e A. Furuhata, "CNN based transfer learning for historical Chinese character recognition", em *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, IEEE, 2016, pp. 25–29.

[60] G. Wimmer, A. Vécsei e A. Uhl, "CNN transfer learning for the automated diagnosis of celiac disease", em *2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, IEEE, 2016, pp. 1–6.

[61] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li e L. Fei-Fei, "Imagenet: A large-scale hierarchical image database", em *2009 IEEE conference on computer vision and pattern recognition*, IEEE, 2009, pp. 248–255.

[62] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang e C. Liu, "A survey on deep transfer learning", em *International conference on artificial neural networks*, Springer, Rhodes, Grécia, 2018, pp. 270–279.

[63] L. Perez e J. Wang, "The effectiveness of data augmentation in image classification using deep learning", *arXiv preprint arXiv:1712.04621*, 2017.

[64] Z. Zhong, L. Zheng, G. Kang, S. Li e Y. Yang, "Random erasing data augmentation", *arXiv preprint arXiv:1708.04896*, 2017.

[65] A. Mikołajczyk e M. Grochowski, "Data augmentation for improving deep learning in image classification problem", em *2018 international interdisciplinary PhD workshop (IIPhDW)*, IEEE, 2018, pp. 117–122.

[66] C. Shorten e T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning", *Journal of Big Data*, vol. 6, n.º 1, p. 60, 2019.

[67] Tensorflow, *An end-to-end open source machine learning platform.* `https://www.tensorflow.org/`, jan. de 2020.

[68] M. S. d. Melo, "Dual scaling: uma implementação em GPU com o TensorFlow", trabalho de conclusão de curso, Universidade Federal Fluminense, Niterói, 2018.

[69] M. J. Rodríguez González, "Raspberry pi como plataforma de algoritmos de Machine Learning: reconocimiento de imágenes y datos financieros en streaming", trabalho de conclusão de curso, Universidad de Sevilla, 2018.

[70] H. d. Matos et al., "Desenvolvimento e validação de um sistema de identificação de emoções por visão computacional e redes neurais convolucionais com transferência de aprendizado", dissertação de mestrado, Universidade Presbiteriana Mackenzie, 2017.

[71] V. K. Shrivastava, N. D. Londhe, R. S. Sonawane e J. S. Suri, "Reliable and accurate psoriasis disease classification in dermatology images using comprehensive feature space in machine learning paradigm", *Expert Systems with Applications*, vol. 42, n.º 15-16, pp. 6184–6195, 2015.

[72] ——, "Exploring the color feature power for psoriasis risk stratification and classification: a data mining paradigm", *Computers in biology and medicine*, vol. 65, pp. 54–68, 2015.

[73] L.-s. Wei, Q. Gan e T. Ji, "Skin Disease recognition method based on image color and texture features", *Computational and mathematical methods in medicine*, vol. 2018, 2018.

[74] J. Velasco, C. Pascion, J. W. Alberio, J. Apuang, J. S. Cruz, M. A. Gomez, B. Molina Jr, L. Tuala, A. Thio-ac e R. Jorda Jr, "A Smartphone-Based Skin Disease Classification Using MobileNet CNN", *arXiv preprint arXiv:1911.07929*, 2019.

[75] M. Dash, N. D. Londhe, S. Ghosh, A. Semwal e R. S. Sonawane, "PsLSNet: Automated psoriasis skin lesion segmentation using modified U-Net-based fully convolutional network", *Biomedical Signal Processing and Control*, vol. 52, pp. 226–237, 2019.

[76] M. Dash, N. D. Londhe, S. Ghosh, R. Raj e R. S. Sonawane, "A cascaded deep convolution neural network based CADx system for psoriasis lesion segmentation and severity assessment", *Applied Soft Computing*, pp. 106–240, 2020.

[77] S. Zhao, B. Xie, Y. Li, X.-y. Zhao, Y. Kuang, J. Su, X.-y. He, X. Wu, W. Fan, K. Huang et al., "Smart identification of psoriasis by images using convolutional neural networks: a case study in China", *Journal of the European Academy of Dermatology and Venereology*, vol. 34, n.º 3, pp. 518–524, 2020.

[78] D. Padilla, A. Yumang, A. L. Diaz e G. Inlong, "Differentiating Atopic Dermatitis and Psoriasis Chronic Plaque using Convolutional Neural Network MobileNet Architecture", em *2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, IEEE, pp. 1–6.

[79] I. Sommerville, *Engenharia de Software*, 9ª ed. São Paulo, SP: Pearson Prentice Hall, 2011.

[80] B. Xie, X. He, S. Zhao, Y. Li, J. Su, X. Zhao, Y. Kuang, Y. Wang e X. Chen, "XiangyaDerm: A Clinical Image Dataset of Asian Race for Skin Disease Aided Diagnosis", em *Large-Scale Annotation of Biomedical Data and Expert Label Synthesis and Hardware Aware Learning for Medical Imaging and Computer Assisted Intervention*, Springer, 2019, pp. 22–31.

[81] K. Simonyan e A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[82] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov e L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks", em *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[83] V. Iglovikov e A. Shvets, "Ternausnet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation", *arXiv preprint arXiv:1801.05746*, 2018.

[84] A. A. Shvets, A. Rakhlin, A. A. Kalinin e V. I. Iglovikov, "Automatic instrument segmentation in robot-assisted surgery using deep learning", em *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2018, pp. 624–628.

[85] V. Badrinarayanan, A. Kendall e R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation", *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, n.º 12, pp. 2481–2495, 2017.

[86] F. Chollet, "Xception: Deep learning with depthwise separable convolutions", em *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[87] C. Szegedy, S. Ioffe, V. Vanhoucke e A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning", *arXiv preprint arXiv:1602.07261*, 2016.

[88] Keras, *Simple. Flexible. Powerful.* `https://keras.io/`, set. de 2020.

[89] P. Heckbert, *Graphics Gems IV (IBM Version)*. Elsevier, 1994.

[90] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot e E. Duchesnay, "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[91] C. D. Manning, H. Schütze e P. Raghavan, *Introduction to information retrieval*. New York: Cambridge university press, 2008.

[92] G. Developers, *Machine Learning Crash Course.* `https://developers.google.com/machine-learning/glossary#ł`, out. de 2020.

[93] M. Shafiq, X. Yu, A. K. Bashir, H. N. Chaudhry e D. Wang, "A machine learning approach for feature selection traffic classification using security analysis", *The Journal of Supercomputing*, vol. 74, n.º 10, pp. 4867–4892, 2018.

[94] N. Japkowicz, "Why question machine learning evaluation methods", em *AAAI workshop on evaluation methods for machine learning*, 2006, pp. 6–11.

[95] V. Verma e R. K. Aggarwal, "A comparative analysis of similarity measures akin to the Jaccard index in collaborative recommendations: empirical and theoretical perspective", *Social Network Analysis and Mining*, vol. 10, pp. 1–16, 2020.