UNIVERSIDADE FEDERAL DO MARANHÃO

Curso de Ciência da Computação

Arthur Azevedo Lima

Uma Proposta de Metodologia para Segmentação Semântica de Pólipos em Imagens de Colonoscopia

São Luís

2023

Arthur Azevedo Lima

Uma Proposta de Metodologia para Segmentação Semântica de Pólipos em Imagens de Colonoscopia

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Profa. Dra. Simara Vieira da Rocha

São Luís

2023

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Azevedo Lima, Arthur.
Uma Proposta de Metodologia para Segmentação Semântica de Pólipos em Imagens de Colonoscopia / Arthur Azevedo Lima. - 2023.
58 f.

Orientador(a): Simara Vieira da Rocha.
Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís, 2023.

1. Câncer de cólon. 2. Deep learning. 3. Redes neurais artificiais. 4. Segmentação semântica. I. Vieira da Rocha, Simara. II. Título.

Arthur Azevedo Lima

Uma Proposta de Metodologia para Segmentação Semântica de Pólipos em Imagens de Colonoscopia

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Trabalho _____ em São Luís, 16 de dezembro de 2023:

Profa. Dra. Simara Vieira da Rocha
Orientadora

Prof. Dr. Geraldo Braz Júnior
Examinador

Prof. Dr. Anselmo Cardoso de Paiva
Examinador

São Luís
2023

Agradecimentos

Primeiramente, gostaria de agradecer aos professores do curso de Ciência da Computação da Universidade Federal do Maranhão, por me ensinar tudo o que eu sei sobre a minha vocação. E, principalmente, a professora Simara Vieira da Rocha, por sua paciência, seu apoio e seus conselhos, que foram cruciais para que eu pudesse completar este trabalho.

Mas, acima de tudo, gostaria de agradecer a minha família. Aos meus pais, Teonio do Carmo Lima e Odaisa Ericeira Azevedo. Todas as oportunidades e a capacidade de aproveitá-las, eu devo a eles.

Aos meus avós, Teresinha Lourdes do Carmo Lima, Antônio Pereira Lima, Odino Azevedo, e minha finada vó Maria Heloisa Ericeira Azevedo, por tudo que eles fizeram por mim, muito antes que eu pudesse sequer prometer algo em troca.

Aos meus irmãos, Heitor, Davi e o pequeno Caetano, que sempre me deixam maravilhado e orgulhoso com seu potencial incrível e motivado a ser o melhor que eu posso ser.

Aos meus primos, Miguel, Valentina, Leandro, Alexandre, e Maria Heloísa e aos meus tios, Tania Charlene, João Lucas, Antônio, e Odino Júnior.

A minha querida madrasta, Sofia, e sua família, que se juntou com a nossa com muita alegria e carinho.

Ao meu tio de consideração, Henrique, meu padrinho, Walmir Jr., minha madrinha, Piedade, por me mostrar que família não é somente parentesco.

Aos meus grandes amigos, Caio, Beatriz, André, Andreia, Marcus, Davi, Vitor Hugo, e tantos outros, por me lembrar que eu sempre posso achar um lugar no mundo para pertencer.

A esta família que eu amo, agradeço por me amar de volta e nunca ter me deixado esquecer disso.

"Mas eu o tentarei,
como ele próprio aconselhava,
pois o importante é tentar,
mesmo o impossível."

Jorge Amado, em *"A morte e a morte de Quincas Berro D'água"*

Resumo

O câncer de cólon e reto é o quarto tipo de câncer mais comum no Brasil e resulta do desenvolvimento descontrolado de células no intestino grosso que formam um tumor maligno e, se não tratado, se espalha para outras partes do corpo, possivelmente causando a morte. No ano de 2021, no Brasil, ele foi responsável por 21.260 óbitos. As chances de cura dessa doença são muito maiores no estágio inicial e, portanto, o diagnóstico precoce é muito importante. O diagnóstico é realizado a partir da biópsia de lesões suspeitas encontradas no intestino grosso, pois o câncer normalmente se desenvolve a partir delas. O principal exemplo destas lesões são os pólipos. A detecção de lesões geralmente ocorre através da colonoscopia, um exame que permite visualizar o interior do cólon através da introdução de um endoscópio. Além de avaliada durante o exame, a colonoscopia pode ser gravada para análise posterior. Em ambos os casos, é possível aplicar sistemas computacionais de auxílio na detecção (CADe) de indícios de doença, como é o caso dos pólipos. Este trabalho propõe uma metodologia utilizando redes neurais artificiais para separar as regiões de imagens de colonoscopia ocupadas por pólipos do restante da imagem. Esta tarefa é conhecida como segmentação semântica de pólipos em imagens de colonoscopia e possui múltiplas bases de imagens dedicadas para permitir o desenvolvimento e avaliação de soluções para ela. A metodologia proposta foi aplicada e testada em uma destas bases, a Kvasir-SEG, alcançando 93,37% de coeficiente de Dice, 88,91% de índice de Jaccard, 94,41% de precisão e 93,96% de revocação.

Palavras-chave: segmentação semântica, *deep learning*, redes neurais artificiais, câncer de cólon.

Abstract

Colon and rectal cancer is the fourth most common type of cancer in Brazil, and results from the uncontrolled development of cells in the large intestine that form a malign tumor and, if untreated, can spread to other parts of the body, possibly causing death. In 2021, in Brazil, it was responsible for 21,260 deaths. The chance of curing this disease are much higher in the early stages and, therefore, early diagnosis is very important. This diagnosis is performed with the biopsy of suspicious lesions found in the large intestine, as the cancer usually develops from them. The main example of these are the polyps. Lesion detection is normally done through colonoscopy, a medical exam that allows internal visualization of the colon by introducing an endoscope. Besides being evaluated during the procedure, this footage can be recorded for later analysis. In both cases, it is possible to apply computer systems to assist in detection (CADe) of signs of disease, as is the case for polyps. This work proposes a methodology using artificial neural networks to separate colonoscopy image regions occupied by polyps from the rest of the image. This task is known as semantic polyp segmentation in colonoscopy images and features multiple image datasets created specifically to allow development and evaluation of solutions to it. The proposed methodology was applied and tested in one of these datasets, the Kvasir-SEG dataset, attaining 93.37% Dice coefficient, 88.91% Jaccard index, 94.41% precision and 93.96% recall.

Keywords: semantic segmentation, *deep learning*, artificial neural network, colon cancer.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

CC	*Ciência da Computação*
UFMA	*Universidade Federal do Maranhão*
INCA	*Instituto Nacional do Câncer*
CADe	*Computer Assisted Detection*
IA	*Inteligência Artificial*
ML	*Machine Learnig*
DL	*Deep Learning*
CNN	*Convolutional Neural Network*
FCN	*Fully Convolutional Network*
ViT	*Vision Transformer*
MiT	*Mix Transformer*
FCB	*Fully Convolutional Branch*
PVT	*Pyramid Vision Transformer*
GD	*Gradient Descent*
SGD	*Stochastic Gradient Descent*
AHE	*Adaptive Histogram Equalization*
CLAHE	*Contrast Limited Adaptive Histogram Equalization*
BCE	*Binary Cross Entropy*
ReLU	*Rectified Linear Units*
SiLU	*Sigmoid Linear Units*
SCSE	*Spatial and Channel Squeeze and Excite*
VP	*Verdadeiros Positivos*
FP	*Falsos Positivos*
FN	*Falsos Negativos*

Sumário

1 Introdução

O câncer de cólon, também conhecido como câncer de intestino ou câncer colorretal, é uma doença que acomete o intestino grosso e o reto de um indivíduo, caracterizada pela multiplicação descontrolada de células que agridem as células circundantes, se não tratada, pode se espalhar para outras regiões e órgãos do corpo e, eventualmente, causar a morte (INCA, 2022a).

Segundo o Instituto Nacional de Câncer (INCA, 2022b), no Brasil, se não levarmos em consideração o câncer de pele, o câncer de cólon é o segundo tipo de câncer mais comum tanto entre homens, com estimados 21.970 novos casos em 2023, ficando atrás apenas do câncer de próstata, quanto em mulheres, com estimados 23.660 novos casos em 2023, ficando atrás apenas do câncer de mama. Além disso, também possui a terceira maior letalidade em ambos os sexos, sendo responsável por 10.662 óbitos entre homens e 10.598 entre mulheres no ano de 2021. Dentre os tipos de câncer, apenas o de mama, de próstata, e de pulmão foram responsáveis por maiores quantidades de óbitos (INCA, 2022b).

No Maranhão, o câncer de cólon e reto é o quarto mais comum entre homens e mulheres, sendo superado pelo câncer de pulmão e de estômago, além do de próstata, na população masculina, e pelo câncer de colo de útero e de glândula tireóide, além do de mama, na população feminina. Ao todo, estimam-se 520 novos casos no estado em 2023 (INCA, 2022c).

Este tipo de câncer é tratável e, na maioria dos casos, curável se detectado precocemente, antes de se desenvolver e espalhar para outras partes do corpo. A presença de certos sintomas pode ser um indício de câncer de cólon, como mudanças drásticas no hábito intestinal, fezes com presença de sangue, dores abdominais, anemia, fraqueza e perda de peso, entre outros. Porém, estes sintomas são compartilhados com outras doenças do sistema digestivo. A confirmação do diagnóstico de câncer de cólon envolve a detecção e subsequente análise de lesões suspeitas no reto ou no intestino grosso a partir de uma colonoscopia (INCA, 2022a).

Devido ao impacto da precocidade do diagnóstico na chance de sobrevivência dos pacientes, existe muito interesse em desenvolver formas de agilizar esse processo. Uma destas formas são os sistemas de detecção auxiliada por computador, computer aided detection ou CADe, que visam automatizar parcialmente a detecção de pólipos. Este tipo de aplicação não é uma exclusividade do câncer de cólon, o processamento de imagens biomédicas para auxílio à detecção e diagnóstico sendo um grande motivador de trabalhos na área de visão computacional (RONNEBERGER; FISCHER; BROX, 2015).

Nesta área de visão computacional, assim como na maior parte da área de inteligência artificial, o estado da arte atual é dominado pelas redes neurais artificiais. Objeto de estudo do campo de *deep learning*, as redes neurais artificiais são modelos matemáticos que tentam extrair e armazenar conhecimento abstrato para aplicação no mundo real através de tentativa e erro (GOODFELLOW; BENGIO; COURVILLE, 2016).

1.1 Objetivos

O objetivo geral desse trabalho é propor uma metodologia utilizando rede neural artificial capaz de desempenhar a tarefa de segmentação semântica de pólipos em imagens de colonoscopia.

1.1.1 Objetivos Específicos

Destaca-se como objetivos específicos deste trabalho:

- analisar técnicas de realce de imagens que possibilitem uma melhor descrição do objeto de interesse (pólipos);
- estabelecer uma rede neural que seja capaz de realizar a segmentação de pólipos em imagens de colonoscopia;
- avaliar a arquitetura proposta por meio de experimentos.

1.2 Trabalhos Relacionados

A DUCK-Net (DUMITRU; PETELEAZA; CRACIUN, 2023) é uma arquitetura de codificador-decodificador similar à Unet (RONNEBERGER; FISCHER; BROX, 2015) que apresenta um bloco convolucional inovador, o DUCK (*Deep Understanding Convolutional Kernel*). Este bloco consiste de múltiplos ramos paralelos com diferentes estruturas e operações convolucionais, visando permitir que a rede selecione melhor as características que deve priorizar. Na base Kvasir-SEG, a DUCK-Net atingiu 95,02% de coeficiente de Dice e 90,51% de índice de Jaccard.

O Meta Polyp (TRINH, 2023) é uma aplicação do paradigma Meta Former para a segmentação de pólipos. O Meta Former é um conceito introduzido por Yu et al. (2022), afirmando que o sucesso dos Transformers não decorre especificamente da operação de auto atenção, mas sim da estrutura dos seus blocos de arquitetura. O Meta Polyp é uma arquitetura decodificador-codificador que possui, como *backbone* do codificador, um CA Former (YU et al., 2023) pré-treinado. Na base Kvasir-SEG, ele alcançou 95,9% de coeficiente de Dice e 92,1% de índice de Jaccard.

A ESFPNet foi desenvolvida por Chang et al. (2023) para segmentação de lesões em imagens de broncoscopia, um exame para detecção de câncer de pulmão. Porém, o trabalho que a propôs também avaliou o modelo em outros domínios, entre eles, a segmentação de pólipos em colonoscopia. A abordagem é um modelo codificador-decodificador, cujo segmento codificador consiste em um Mix Transformer (MiT) pré-treinado e o segmento decodificador é uma rede em formato de pirâmide, chamada de *efficient stage-wise feature pyramid* ou ESFP. A ESFP é um segmento de rede neural, também proposto por Chang et al. (2023), que visa agregar as características de diferentes níveis do codificador de forma eficiente e balanceada. Na base Kvasir-SEG, a metodologia alcançou 93,1% de coeficiente de Dice e 88,7% de índice de Jaccard.

Hung et al. (2023) propôs a UGCANet (*Unified Global Context Aware Network*) para múltiplas tarefas relacionadas a endoscopia do sistema digestivo, incluindo a segmentação de pólipos em colonoscopia. A arquitetura da UGCANet para segmentação de colonoscopia também é do tipo codificador-decodificador e utiliza o MiT como *backbone* do codificador. O segmento decodificador do modelo é uma rede de pirâmide proposta por Huang et al. (2021), chamada de *Feature-aligned Pyramid Network* ou FaPN. Além disso, as conexões entre o codificador e decodificador passam por um módulo que agrega informação sobre o contexto global, composto por um bloco CGNL (*Compact Generalized Non-Local*), proposto por Yue et al. (2018), seguido por um bloco de *squeeze and excite* (ROY; NAVAB; WACHINGER, 2018). Na base Kvasir-SEG, a UGCANet alcançou 92,8% de coeficiente de Dice e 88,1% de coeficiente de Jaccard.

O Decodificador de Atenção por Convolução de Grafo em Cascata, *Cascaded Graph Convolutional Attention Decoder* ou G-CASCADE (RAHMAN; MARCULESCU, 2023), foi desenvolvido como um módulo decodificador para arquiteturas codificador-decodificador com *backbones* baseadas em atenção (Transformers). Seu desempenho foi avaliado em múltiplas tarefas de segmentação de imagens biomédicas: órgãos abdominais, órgãos cardíacos, pólipos, lesões de pele, e vasos retinais. Assim como a ESFP, o G-CASCADE visa decodificar as características multinível extraídas pelo codificador Transformer aproveitando tanto as características locais quanto globais. O PVT G-CASCADE é uma aplicação do decodificador em conjunto com um codificador baseado PVT v2 (*Pyramid Vision Transformer v2*). Na base Kvasir-SEG, o PVT G-CASCADE alcançou 92,74% de coeficiente de Dice e 87,90% de índice de Jaccard.

Sanderson e Matuszewski (2022) apresentou o conceito de arquiteturas de fusão FCN-ViT, visando, assim como outros trabalhos previamente citados, aproveitar a informação extraída por *backbones* de Vision Transformer mas sem perder os benefícios das FCNs. Para efetuar isto, as arquiteturas de fusão utilizam duas arquiteturas codificador-decodificador em paralelo, onde uma é baseada numa *backbone* de ViT e a outra é inteiramente convolucional, e combina as características produzidas por ambas para

produzir a segmentação. A instância deste tipo de modelo apresentada é chamada de FCB Former e utiliza o PVT v2 como codificador no seu módulo ViT. Na base Kvasir-SEG, o FCB Former alcançou 93,85% de coeficiente de Dice e 89,03% de coeficiente de Jaccard.

O FCB Swin V2 Transformer é uma instância de arquitetura de fusão FCN-Transformer apresentada por Fitzgerald e Matuszewski (2023). A principal diferença entre ele e o FCB Former é o módulo de ViT proposto, que utiliza o Swin Transformer V2 (LIU et al., 2022) como *backbone*. Na base Kvasir-SEG, ele alcançou 94,20% de coeficiente de Dice e 89,73% de Jaccard.

A Tabela 1 resume os trabalhos relacionados, bem como os resultados obtidos por eles.

Tabela 1 – Resultados reportados pelos trabalhos relacionados usando a base Kvasir-SEG

Métrica	**Coeficiente de Dice**	**Índice de Jaccard**
Meta-Polyp	95,90%	92,10%
DUCK-Net	95,02%	90,51%
ESFPNet	93,10%	88,70%
UGCANet	92,80%	88,10%
PVT-GCASCADE	92,74%	87,90%
FCB Former	93,85%	89,03%
FCB Swin V2 Transformer	94,20%	89,73%

1.3 Organização do Trabalho

Esta monografia está organizada em cinco capítulos. No Capítulo 2, será apresentada toda a fundamentação teórica usada no desenvolvimento deste trabalho.

Em seguida, no Capítulo 3, serão descritos os procedimentos realizados para segmentação semântica de pólipos em imagens de colonoscopia.

O Capítulo 4 irá apresentar e discutir os resultados obtidos por meio dos experimentos realizados. Finalmente, no Capítulo 5, serão feitas algumas conclusões, bem como apresentadas sugestões de trabalhos futuros.

2 Fundamentação Teórica

Neste capítulo será apresentada a fundamentação teórica usada no desenvolvimento desta monografia, importante para compreensão dos métodos utilizados para alcançar os objetivos esperados. Para tanto, abordaremos o câncer de cólon, o processamento de imagens digitais, a inteligência artificial, e o método de validação dos resultados.

2.1 Câncer de Cólon

O termo câncer refere-se a um conjunto de doenças malignas caracterizadas pela multiplicação descontrolada de células, que se tornam agressivas, podendo formar tumores e invadir outros tecidos e órgãos, se espalhando pelo corpo. Estas doenças são classificadas de acordo com o tipo de célula e região do corpo ou órgão onde se origina o tumor (INCA, 2022d). O câncer colorretal agrupa aqueles que surgem no intestino grosso, o cólon, ou no segmento final do intestino, o reto (INCA, 2022a).

Grande parte dos tumores deste tipo se origina a partir de lesões benignas na parede do intestino grosso chamadas de pólipos, podendo-se confirmar o diagnóstico de câncer de intestino através da biópsia de tecido retirado da lesão suspeita. Além disso, a remoção precautória de um pólipo previne a eventual ocorrência de um tumor a partir dele. O método preferencial de diagnóstico é a colonoscopia, um exame endoscópico que permite a avaliação de todo o intestino grosso, bem como a detecção, remoção ou biópsia de tumores (INCA, 2022a).

O câncer de intestino é o objeto de estudo deste trabalho, especificamente o uso de técnicas de processamento de imagens digitais e aprendizado de máquina para auxiliar no desempenho do exame de colonoscopia, através da segmentação de pólipos em imagens deste tipo de exame.

2.2 Processamento de Imagens Digitais

Para entender o funcionamento das técnicas de processamento de imagens digitais, é preciso compreender como os computadores “entendem” uma imagem. Uma imagem monocromática pode ser representada matematicamente como uma função $f(x, y) = z$, onde (x, y) é qualquer ponto da imagem e z é a intensidade da luz naquele ponto. Para que um computador possa armazenar e processar uma imagem, assim como qualquer tipo de dado, é preciso representá-la digitalmente. Uma imagem monocromática nada mais é do que a mesma função f, onde x, y, e z são discretizadas, podendo ser representada como

uma matriz. Dada uma imagem de altura h e largura w, convenciona-se que o ponto $(0, 0)$ é o canto superior esquerdo da imagem, o ponto $(h, 0)$ é o inferior esquerdo, $(0, w)$ é o superior direito, e (h, w) é o inferior direito, com a função sendo indefinida para qualquer ponto fora da área retangular definida por eles. O par de dimensões $h \times w$ da imagem é chamado da resolução da imagem e cada ponto (x, y) em uma imagem digital é chamado de pixel. Imagens coloridas são representadas como múltiplas imagens monocromáticas sobrepostas, estas camadas são chamadas de canais e cada uma representa a intensidade de um tipo específico de luz que ao se combinar com os outros canais produz uma imagem colorida (GONZALEZ; WOODS, 2000).

Técnicas de processamento de imagens são computações específicas sobre uma imagem digital visando obter alguma informação automaticamente a partir da imagem ou visando alterá-la para se adequar melhor a um determinado contexto, como a observação humana ou algum outro processamento seguinte (GONZALEZ; WOODS, 2000). O universo de técnicas desenvolvidas e aplicáveis é vasto e variado, mas para os propósitos deste trabalho destacam-se: fatiamento (ou *thresholding*), a equalização de histograma, a equalização adaptativa de histograma limitada pelo contraste (*contrast limited adaptive histogram equalization* ou CLAHE), e a convolução.

O fatiamento de brilho, ou *thresholding*, consiste em destacar os pixels que estejam dentro de uma determinada faixa de brilho e/ou ocultar os que estejam fora dela. Esta técnica é muito comumente utilizada para binarizar uma imagem, maximizando o brilho de um pixel que esteja acima de um determinado nível de brilho e zerando o brilho daqueles que estejam abaixo (GONZALEZ; WOODS, 2000).

O histograma de uma imagem é a distribuição da frequência dos seus níveis de brilho, e é uma forma de visualizar algumas informações úteis sobre ela, como o nível de contraste. Ele é definido como a função $h(x)$ descrita na Equação 2.1, onde x é um número inteiro entre 0 e o nível máximo de brilho da imagem representando um nível de brilho específico e n_x é o número de pixels da imagem que possuem nível de brilho x.

$$h(x) = n_x \tag{2.1}$$

Um histograma normalizado é semelhante ao histograma absoluto, porém, ao invés de trazer a frequência absoluta de cada nível de brilho, ele traz a frequência relativa, ou seja, conforme definido pela função $p(x)$ na Equação 2.2, onde n é o número total de pixels. Este tipo de histograma também pode ser lido como a distribuição de probabilidades do nível de brilho de um pixel escolhido aleatoriamente (GONZALEZ; WOODS, 2000).

$$p(x) = \frac{n_x}{n} \tag{2.2}$$

O histograma cumulativo, descrito na Equação 2.3, é outra forma importante de visualizar uma distribuição. Ele indica o número de pixels que estão abaixo de um

determinado nível de brilho. E também pode ser normalizado, indicando a probabilidade do nível de brilho de um pixel escolhido aleatoriamente estar abaixo de x.

$$c(x) = \sum_{i=0}^{x} n_i \tag{2.3}$$

A equalização de histograma (*histogram equalization*), também conhecida com linearização de histograma, é uma técnica de de processamento de imagens que redistribui os níveis de brilho da imagem de entrada de tal forma que a imagem resultante tem um histograma é uniforme mas mantém o brilho relativo dos pixels. Matematicamente, ela consiste em obter o histograma normalizado cumulativo da imagem original e produzir uma nova imagem onde o nível de brilho de cada pixel é igual ao produto entre o nível máximo de brilho $L - 1$ e a frequência cumulativa do valor do pixel correspondente na imagem original, conforme descrito na Equação 2.4 (GONZALEZ; WOODS, 2000).

$$E_{h \times w} = hist.eq.(X_{h \times w}) \implies e_{i,j} = \lceil c(x_{i,j})(L-1) \rfloor \tag{2.4}$$

A equalização de histograma adaptativa, *adaptive histogram equalization* ou AHE, é uma forma variante da equalização de histograma comum que determina o novo valor de cada pixel com base apenas no histograma de uma determinada área que o cerca, a sua "área contextual". Esta técnica visa preservar detalhes locais que a equalização de histograma comum não leva em conta devido à sua natureza global (PIZER et al., 1987). Uma das formas de facilitar o cálculo da AHE é dividir a imagem em regiões fixas de mesmo tamanho, e realizando a equalização de histograma usual sobre elas, mas esta solução gera um problema: um efeito "xadrez" sobre a imagem nas fronteiras entre as regiões (GONZALEZ; WOODS, 2000). Porém, é possível de remover estes artefatos indesejáveis utilizando interpolação (OPENCV, 2017).

Outro problema que resulta da aplicação da AHE é que a equalização do histograma de uma área muito pequena pode magnificar o ruído ali contido. Para solucioná-lo, pode-se aplicar a técnica de limitação de contraste para diminuir a intensidade do aumento do contraste dentro de cada área. Esta variação da técnica é chamada de equalização adaptativa de histograma com contraste limitado, ou CLAHE (*contrast limited adaptive histogram equalization*). A aplicação da CLAHE é similar à AHE, porém, após computar o histograma da área mas antes de obter o histograma cumulativo a partir dele, adiciona-se um novo passo. Este passo consiste em identificar todas as classes do histograma com contagem acima de um limite de contraste pré-definido e redistribuir este excedente igualmente ao longo de todas as classes. A partir daí, obtém-se o histograma cumulativo e o algoritmo procede normalmente (PIZER et al., 1987).

A convolução é uma operação matemática aplicada sobre entradas organizadas em grade, como uma série temporal ou uma imagem. A operação é definida tal que a saída S, também chamada de mapa de características ou *feature map*, da convolução entre o

argumento X, chamado de entrada, e o argumento W, chamado de filtro ou *kernel*, é uma grade onde cada elemento de s de S é a soma ponderada do elemento correspondente x em X com todos os elementos de X, cujos pesos são definidos por W centrado em x (GOODFELLOW; BENGIO; COURVILLE, 2016).

Se todos os dados são discretos, X e W são unidimensionais, e W possui módulo igual a $2p$, a saída S pode ser definida nos termos da Equação 2.5.

$$s_i = \sum_{m=0}^{2p} x_{(i-p+m)} w_m \tag{2.5}$$

Em casos bidimensionais, onde W é uma matriz $2p \times 2q$, pode-se definir S com a Equação 2.6.

$$s_i = \sum_{m=0}^{2p} \sum_{n=0}^{2q} x_{(i-p+m),(j-q+n)} w_{m,n} \tag{2.6}$$

Esta operação possui várias aplicações no processamento de imagens digitais, a maioria das quais são exemplos de filtragem, onde utilizam-se filtros previamente definidos para realizar convolução sobre uma imagem, produzindo diferentes efeitos dependendo das características do filtro (GONZALEZ; WOODS, 2000). É possível modificar a operação além dos valores do filtro, como através da alteração do tamanho do filtro ou através da mudança do passo ou *stride* (t) da operação, fazendo com que a operação "pule" $t - 1$ elementos após cada elemento processado de X, resultando em uma saída t vezes menor do que a entrada.

2.3 Inteligência Artificial

Inteligência Artificial é o nome dado à área da computação que visa criar aplicações capazes de resolver problemas que requerem a posse e aplicação de conhecimento, ou seja, que requerem inteligência, sem a necessidade de um algoritmo fixo projetado e implementado por humanos. Existem diferentes abordagens dentro desta área que normalmente podem ser divididas entre abordagens de "base de conhecimento" (*knowledge base*) e "aprendizado de máquina" (*machine learning* ou ML) (GOODFELLOW; BENGIO; COURVILLE, 2016).

Abordagens de base de conhecimento visam conferir inteligência a uma aplicação através do fornecimento de uma base abrangente de predicados formalizados sobre o domínio no qual ela deverá operar, a partir dos quais o programa poderá tirar conclusões usando regras de inferência lógica. Algoritmos deste tipo são, em geral, inviáveis, devido não só ao custo de produzir uma base como esta com também à dificuldade de criar um sistema de predicados formais com complexidade o suficiente para representar a realidade (GOODFELLOW; BENGIO; COURVILLE, 2016).

O aprendizado de máquina, por sua vez, visa permitir que sistemas de inteligência artificial produzam seu próprio conhecimento através da experiência. Algoritmos deste tipo, mesmo os mais simples, são muito mais amplamente aplicáveis do que os do tipo anterior, porém sua eficácia ainda depende da representação dos dados fornecidos. A representação é a forma que situação-problema é apresentada para o algoritmo e, normalmente, consiste em um conjunto de informações consideradas relevantes, denominadas de características. Infelizmente, em muitas tarefas, especialmente aquelas do cotidiano, é difícil definir quais características devem ser extraídas e, além disso, como elas devem ser extraídas sem a assistência de um operador humano. Isso se deve ao fato de que cenários reais estão sob a influência de muitos fatores de difícil observação, como as condições de iluminação ou o ângulo de observação em algoritmos que trabalham com fotos (GOODFELLOW; BENGIO; COURVILLE, 2016).

O aprendizado profundo (*deep learning* ou DL) é um tipo de abordagem de aprendizado de máquina que consegue, automaticamente, descobrir quais são e como extrair as características que melhor representam um problema e como elas se relacionam à solução. O poder e flexibilidade deste tipo de ML, bem como sua denominação de "profundo", se deve à sua capacidade de compreender o mundo através de uma hierarquia aninhada de conceitos, onde conceitos mais complexos e mais abstratos são definidos nos termos de conceitos mais simples e menos abstratos. Soluções de DL normalmente se organizam em várias camadas interconectadas que executam diferentes operações matemáticas sobre o produto das camadas que lhe antecedem, cada camada sendo composta por múltiplas unidades que realizam a mesma operação sobre a mesma entrada com pesos diferentes. Ao projeto de organização das camadas de um modelo, é dado o nome de arquitetura (GOODFELLOW; BENGIO; COURVILLE, 2016).

Estes modelos são comumente chamados de redes neurais artificiais, uma referência às ambições de modelar do comportamento de cérebros biológicos que motivaram o desenvolvimento dos primeiros algoritmos de aprendizado profundo em meados do século passado, bem como ao fato de que a neurociência é fonte de inspiração para o campo até os dias atuais, a exemplo das redes convolucionais (LECUN et al., 1989) e dos mecanismos de atenção (VASWANI et al., 2017). A popularidade do campo de *deep learning* na comunidade de pesquisa flutuou intensamente ao longo das décadas que seguiram sua origem, porém a conjuntura atual proporcionou uma "era de ouro" na área, devido principalmente à disponibilidade de grandes volumes de dados para treinamento e ao significativo avanço do poder computacional que permitiu o aumento da complexidade dos modelos (GOODFELLOW; BENGIO; COURVILLE, 2016).

Uma vez que a arquitetura do modelo está definida, ele aprende através do ajuste de seus pesos, ou seja, é a configuração de pesos de um modelo que contém seu aprendizado. Os pesos são ajustados visando maximizar a eficácia do modelo, ou, em outras palavras,

minimizar os seus erros. Matematicamente, a função L que computa e quantifica a incorreção da inferência de um modelo M operando sobre um conjunto de exemplos T é a função de perda ou função de custo, chamada primariamente de função *loss* neste trabalho (GOODFELLOW; BENGIO; COURVILLE, 2016).

O algoritmo que faz o ajuste dos pesos e visa reduzir o máximo possível a função de perda é chamado de otimizador. A base para grande parte dos otimizadores utilizados em DL é a descida do gradiente, também conhecido como *gradient descent* ou GD, um algoritmo que realiza a procura pelo valor mínimo da *loss* utilizando como guia a sua derivada em relação ao modelo M no ponto atual. O resultado desta derivada é o gradiente $\frac{\delta L(M,T)}{\delta M}$, ou seja, ele indica a variação da função *loss* de acordo com a variação da configuração de pesos de M. A partir disto, é feito o ajuste dos pesos de M, cuja intensidade pode ser controlada através do parâmetro "taxa de aprendizado" ou "*learning rate*". Desta maneira, o algoritmo atualiza os pesos ao longo de vários passos muito pequenos até encontrar um ponto onde o gradiente seja zero ou próximo de zero, o que indica a presença de um mínimo local ou ponto de inflexão (GOODFELLOW; BENGIO; COURVILLE, 2016).

O gradiente das várias camadas é calculado através do algoritmo de retropropagação, também conhecido como *back propagation* ou backprop. Para aplicação do algoritmo é necessário representar o modelo como um grafo direcionado representando o fluxo de informação ao longo do processamento, onde cada nodo é uma operação simples, com as camadas do modelo sendo um conjunto de um ou mais nodos. Assim, após uma unidade de informação percorrer toda a rede e a função de perda e seu gradiente serem calculados para a última camada, o algoritmo viaja no sentido oposto do grafo, calculando o gradiente para cada nodo ao aplicar a regra da cadeia de cálculo (Equação 2.7). A partir destes gradientes, pode-se aplicar o GD para ajustar os pesos (GOODFELLOW; BENGIO; COURVILLE, 2016).

$$\frac{\delta z}{\delta x} = \frac{\delta z}{\delta y}\frac{\delta y}{\delta x} \iff y = f(x) \quad \text{e} \quad z = g(y) \tag{2.7}$$

Apesar de muito importante, o algoritmo de descida do gradiente puro não é viável para o treinamento em *deep learning*, pois ele depende do cálculo da função *loss* em todo o conjunto de treinamento. Isso faz com que a complexidade de um único passo do algoritmo aumente linearmente com o tamanho do conjunto de treinamento, entrando em conflito com o fato de que a eficácia de algoritmos de DL, e ML em geral, depende de um grande volume de dados (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para resolver este problema, foi desenvolvida uma variação deste algoritmo: a descida do gradiente estocástica, também conhecido como *stochastic gradient descent* ou SGD. Este algoritmo funciona a partir da constatação de que as funções de perda sobre um conjunto de exemplos normalmente são compostas pelo somatório ao longo de todos os exemplos do resultado de uma função de perda calculada individualmente em cada

exemplo. Sabendo disso, concluiu-se que, a cada passo, é possível estimar o gradiente sobre o conjunto de treino T como um todo a partir de um conjunto menor de exemplos obtidos aleatoriamente de T. Este subconjunto T' é comumente chamado de minilote, ou simplesmente lote (*minibatch* ou *batch*), possui tamanho fixo mas parametrizável, normalmente entre 1 e algumas centenas. Este algoritmo foi muito importante para viabilizar a área de *deep learning*, permitindo que os modelos se beneficiem do treinamento com descida de gradiente dentro de um tempo de execução hábil, independente do tamanho de T. Até hoje, os otimizadores mais comumente usados, como RMSprop e Adam, são variações do SGD (GOODFELLOW; BENGIO; COURVILLE, 2016).

Uma variação importante do SGD é o algoritmo chamada de *momentum*, ou quantidade de movimento, que trata o modelo como uma partícula em movimento sobre a superfície determinada pela função de perda e, portanto, leva em conta os gradientes anteriores ao ajustar os pesos, tal qual a energia cinética de um objeto. A velocidade acumulada decai de acordo com um hiperparâmetro α que representa algo similar ao arrasto viscoso operando sobre uma partícula em movimento (GOODFELLOW; BENGIO; COURVILLE, 2016).

A taxa de aprendizado é um dos hiperparâmetros mais difíceis de escolher, pois afeta fortemente o desempenho do modelo. Ao longo da década passada, introduziram-se alguns algoritmos otimizadores com minilotes capazes de adaptar as taxas de aprendizado dos parâmetros do modelo (GOODFELLOW; BENGIO; COURVILLE, 2016).

O primeiro destes, o algoritmo "AdaGrad", reajusta a taxa de aprendizado de cada parâmetro de acordo com seu histórico de gradiente, diminuindo a taxa de parâmetros que afetam a *loss* drasticamente (alto gradiente) e mantendo a taxa daqueles com efeito mais sutil. Na prática, o "AdaGrad" pode resultar em uma diminuição prematura das taxas devido aos valores de gradiente altos no início do treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016).

O algoritmo RMSProp tenta resolver esse problema descartando o histórico do passado extremo ao introduzir um declínio exponencial à média dos gradientes históricos. Ao fazer isso, o algoritmo introduz um novo hiperparâmetro ρ, a taxa de declínio do gradiente acumulado (GOODFELLOW; BENGIO; COURVILLE, 2016). O RMSProp é um dos otimizadores mais utilizados em trabalhos de DL até hoje.

O Adam é outro algoritmo otimizador muito usado na literatura. Ele é similar ao RMSProp, porém se distingue de 2 formas principais. Primeiro, entendendo que o gradiente do lote em um determinado passo é apenas uma estimativa do gradiente completo, ele tenta fazer uma estimativa mais sofisticada incorporando o *momentum* diretamente no sua estimativa do gradiente. Segundo, ressalta que as técnicas introduzidas no RMSProp e *momentum* introduzem variáveis de acumulação que precisam ser inicializadas e, portanto,

introduzem vieses. Devido a isso adiciona uma correção matemática do viés de cada uma ao calcular o ajuste de pesos (GOODFELLOW; BENGIO; COURVILLE, 2016).

Um dos maiores obstáculos a serem superados no desenvolvimento de um algoritmo de *machine learning* é o *overfitting*, ou sobreajustação, onde o algoritmo se adequa às especificidades dos exemplos no conjunto de treinamento ao invés das características gerais da tarefa, criando uma diferença entre seu desempenho durante o treino e seu desempenho na validação ou em cenários reais. Mudanças aplicadas sobre algoritmos de ML visando melhorar especificamente o desempenho de teste e validação ao invés do desempenho em treino, ou seja, aumentar a generalizabilidade do algoritmo, são chamadas de "regularização" (GOODFELLOW; BENGIO; COURVILLE, 2016).

Um dos alvos comuns da regularização é a complexidade do modelo, conforme indicada pelo valor absoluto dos pesos, ou seja, um modelo com pesos de maior módulo é mais complexo do que um com pesos com menor valor absoluto. O regularizador $L2$, comumente chamado de *weight decay*, ou declínio de pesos, busca limitar a complexidade do modelo adicionando um novo termo à função objetivo, que representa a complexidade do modelo e é calculado através da soma dos quadrados de todos os pesos (GOODFELLOW; BENGIO; COURVILLE, 2016). Porém, Loshchilov e Hutter (2017) propõe que a regularização $L2$ não é equivalente a um real declínio de pesos quando aplicada ao algoritmo otimizador Adam e, consequentemente, propõe um desacoplamento do declínio de pesos da função objetivo e aplicação direta no ajuste. O resultado desta proposta é o otimizador AdamW, que possui melhor generalizabilidade do que o Adam padrão.

Outra forma de regularização popular na literatura de ML, especialmente no campo da visão computacional, é a augmentação de dados, ou *data augmentation*. Esta técnica busca aumentar artificialmente o conjunto de dados de treino através da criação de novos exemplos falsos que ainda assim podem ser usados para aprender. Na área de processamento de imagens com *deep learning* este processo consiste em realizar aleatoriamente várias transformações geométricas e no domínio da cor para criar distorções e ruído. Normalmente, múltiplas combinações de transformações diferentes são realizadas em cada caso, mas mantém-se a limitação de apenas um uso de cada caso durante uma época de treinamento. Isto desestimula a adequação excessiva do modelo a características específicas reduzindo o *overfitting* (GOODFELLOW; BENGIO; COURVILLE, 2016).

A definição da função *loss* é outro aspecto importante no desenvolvimento de soluções de ML, visto que ela é o critério que guiará o aprendizado do modelo. Uma função de perda padrão muito utilizada em várias tarefas é a entropia cruzada, representada na Equação 2.8. Ela consiste na expectativa da verossimilhança logarítmica negativa da previsão do modelo em relação ao conjunto de treino.

$$H(P, Q) = - \mathop{\mathbb{E}}_{x \in P} \ln Q(x) \tag{2.8}$$

Em distribuições discretas, como é o caso em treinos de ML, esta expectativa pode ser calculada como o somatório, ao longo de todos os casos, do produto entre a probabilidade na distribuição de treino e o logaritmo da probabilidade prevista, conforme a Equação 2.9 (GOODFELLOW; BENGIO; COURVILLE, 2016).

$$H(P, Q) = -\sum_{x} P(x) \ln Q(x) \tag{2.9}$$

Em tarefas de classificação com classe única (binária) ou múltiplas classes não exclusivas, utiliza-se a entropia cruzada binária, *binary cross entropy* ou BCE, representada na Equação 2.10, que é a soma da entropia cruzada para casos positivos e com a para casos negativos.

$$BCE(P, Q) = -\{\sum_{x}\{P(x) \ln Q(x) + [P(x) - 1] \ln[Q(x) - 1]\}\} \tag{2.10}$$

Foi definido previamente que, em redes neurais artificiais, cada camada realiza uma operação matemática entre suas unidades e os dados de entrada, ambos estruturados como vetores multidimensionais também chamados de tensores. Normalmente, esta operação é uma multiplicação de matrizes, porém esta é apenas um tipo de operação possível. O tipo de camada é definido pela operação que ela realiza, por exemplo, as camadas que realizam a operação mencionada anteriormente são chamadas de densas ou lineares. O tipo da camada define a quantidade e organização de seus parâmetros e dos dados que ela espera receber (GOODFELLOW; BENGIO; COURVILLE, 2016).

Redes Neurais Convolucionais, *convolutional neural networks* ou CNN, são redes neurais que utilizam, em uma ou mais camadas, a operação de convolução. Em aprendizado de máquina, a entrada X são os dados para processamento e o *kernel* W são os parâmetros a serem adaptados. As CNNs são parte do estado da arte em visão computacional com *deep learning* até hoje pois é naturalmente adaptada à natureza espacial das imagens (GOODFELLOW; BENGIO; COURVILLE, 2016).

Em conjunto com a convolução, as CNNs frequentemente utilizam camadas de ativação não-linear e camadas de *pooling*, ambas são camadas não parametrizadas e tem como fim melhorar os mapas produzidos pela convolução. Camadas de ativação simplesmente aplicam uma função de ativação sobre a entrada, como a *rectified linear units* ou ReLU, e as camadas de *pooling* realizam uma operação de resumo estatístico, como média ou máximo, sobre os dados com limites temporais/espaciais pré-definidos (GOODFELLOW; BENGIO; COURVILLE, 2016). Além disso, múltiplas variações da operação de convolução também são comumente utilizadas em CNNs como a convolução por profundidade (*depthwise convolution*) ou a convolução transposta (*transposed convolution*).

A segmentação semântica é uma tarefa de visão computacional cujo objetivo é destacar uma ou mais áreas de uma imagem onde haja presença de pelo menos um dentre

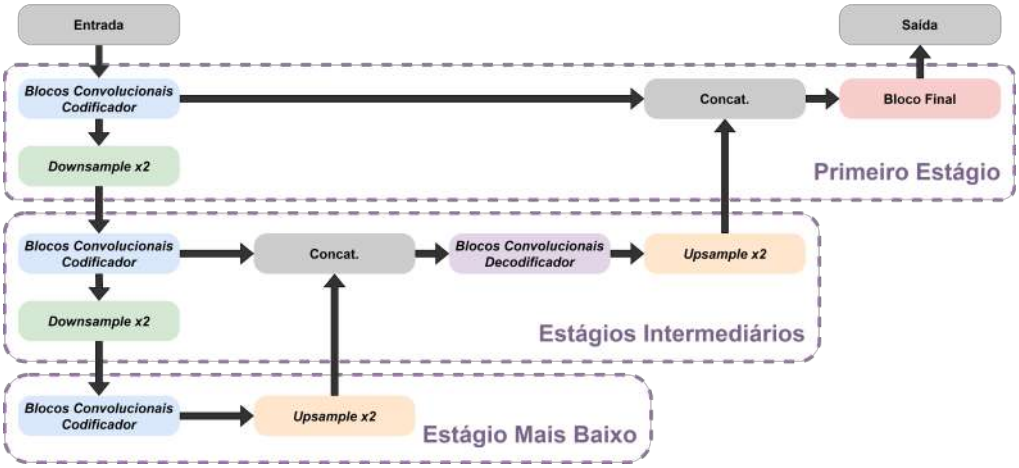
um dado grupo de objetos alvo. Em termos de representação, a segmentação é um problema de classificação onde deve se classificar cada pixel da imagem individualmente. O tensor ou a imagem que contém estas classificações é chamada de máscara de segmentação. A grande maioria das abordagens com redes neurais para tarefas de segmentação no estado da arte ao longo dos últimos anos são redes neurais completamente convolucionais, conhecidas, em inglês, como *fully convolutional networks* ou FCNs. Proposta por Long, Shelhamer e Darrell (2015), FCN é o conjunto de arquiteturas de rede neural que utilizam camadas convolucionais de ponta a ponta do modelo.

Além disso, Long, Shelhamer e Darrell (2015) também fez uso de outra técnica muito comum no estado da arte de segmentação: o *transfer learning* aproveitando o conhecimento obtido em tarefas de classificação. O *transfer learning*, ou transferência de conhecimento, refere-se ao reaproveitamento do que foi aprendido com uma tarefa na resolução de outra. Esse processo pode não só acelerar a convergência como melhorar a generalizabilidade (GOODFELLOW; BENGIO; COURVILLE, 2016). Em *deep learning*, este processo normalmente é feito reutilizando os pesos de uma ou mais camadas previamente treinadas em uma outra tarefa para inicializar os pesos de camadas no modelo a ser treinado na tarefa atual. No trabalho de Long, Shelhamer e Darrell (2015) foram selecionados modelos com notável desempenho em tarefas de classificação gerais, como a ImageNet (DENG et al., 2009), cujos segmentos convolucionais foram, então, reaproveitados como a base de uma FCN, e chamados de *backbones*.

Em tarefas de segmentação semântica de imagens biomédicas, o estado da arte é amplamente definido pela U-Net (RONNEBERGER; FISCHER; BROX, 2015) desde sua publicação até hoje, com a maioria das arquiteturas propostas sendo iterações ou variações sobre ela. A U-Net é uma arquitetura de FCN com algumas características notáveis: a divisão "horizontal", entre codificador e decodificador, e "vertical", entre estágios, e a presença de conexões de salto entre o codificador e o decodificador. O modelo U-Net, representado na Figura 1, se inicia no primeiro estágio do codificador. Este estágio, assim como aqueles que lhe sucedem, é composto por alguns blocos convolucionais e, ao final, uma camada de *downsampling* que reduz as dimensões espaciais dos mapas de características pela metade e as transmite para o estágio seguinte. Este processo é repetido mais 3 vezes até chegar no estágio final, totalizando 5 estágios (este número pode variar de acordo com a arquitetura), além disso, cada estágio possui um número maior de filtros em seus blocos convolucionais, normalmente o dobro do anterior. O quinto estágio é construído similarmente, porém ao invés da camada de *downsampling*, ele possui uma camada de *upsampling* que dobra as dimensões espaciais dos mapas. Isto dá início ao decodificador, que percorre os estágios na ordem inversa. Cada estágio do decodificador, recebe a saída do estágio anterior concatenada com a saída do estágio correspondente do codificador antes do *downsampling*, recebendo a conexão de salto, e é composto por um ou mais blocos

convolucionais seguidos pelo *upsampling*. Isto se repete até retornar ao primeiro estágio onde é produzida a máscara de segmentação.

Figura 1 – Representação de uma arquitetura U-Net genérica (RONNEBERGER; FISCHER; BROX, 2015).



Fonte: acervo do autor.

A área do processamento de linguagem natural, e de *deep learning* como um todo, foi revolucionada recentemente pelo advento dos Transformers (VASWANI et al., 2017), modelos de DL cuja arquitetura é composta primariamente de camadas de mecanismos de atenção, mais especificamente, auto-atenção utilizando uma variação de atenção proposta neste mesmo trabalho, a *multi-head attention*.

A atenção é outro tipo de operação matemática que pode ser usada em camadas de rede neural e baseia-se no sistema de buscas em bancos de dados. A atenção recebe três entradas: um conjunto de buscas (*query* ou Q), um conjunto de valores (*value* ou V), e um conjunto de chaves (*key* ou K), todas são vetores e normalmente V é igual a K. Então, é computada a compatibilidade entre as buscas e as chaves, normalmente através do produto escalar entre cada par de busca e chave ou através da multiplicação de matrizes entre Q e a transposição de K que é equivalente. Estes valores de compatibilidade são usados para produzir uma distribuição softmax, as pontuações de atenção ou *attention scores*. Esta matriz de pontuação é multiplicada pela matriz de valores, produzindo o resultado da operação, que seria equivalente ao retorno de cada busca. Esta operação já era utilizada em DL antes dos Transformers, para ajudar na modelagem em casos onde fosse necessário modelar as relações entre todas as unidades sem considerar a distância entre elas (VASWANI et al., 2017).

A auto-atenção, por sua vez, nada mais é do que uma aplicação da operação de atenção onde os valores também são usados como buscas além de chave, ou seja, $Q = K = V$. Esta técnica é frequentemente utilizada para produzir uma nova representação de uma mesma série de dados (VASWANI et al., 2017).

A *multi-head attention* (MHA), ou atenção de múltiplas cabeças, foi proposta por Vaswani et al. (2017) como uma variação da atenção tradicional com maior poder de modelagem, permitindo que os Transformers se afastassem de paradigmas sequenciais e mais custosos de modelagem de sequências sem grandes perdas de resolução. Uma camada de MHA, separa as buscas, chaves, e valores em várias partes menores que têm suas atenções calculadas paralelamente e cujas saídas são então combinadas novamente para produzir a atenção completa. O número de partes é o número de cabeças, frequentemente referido como h. A forma como as matrizes Q, K, e V de cada cabeça são calculadas a partir das originais, bem como a forma que as atenções são combinadas para produzir a saída final, é através de combinações lineares inteiramente aprendíveis. Isso permite com que cada cabeça aprenda a "prestar atenção" em diferentes características (VASWANI et al., 2017).

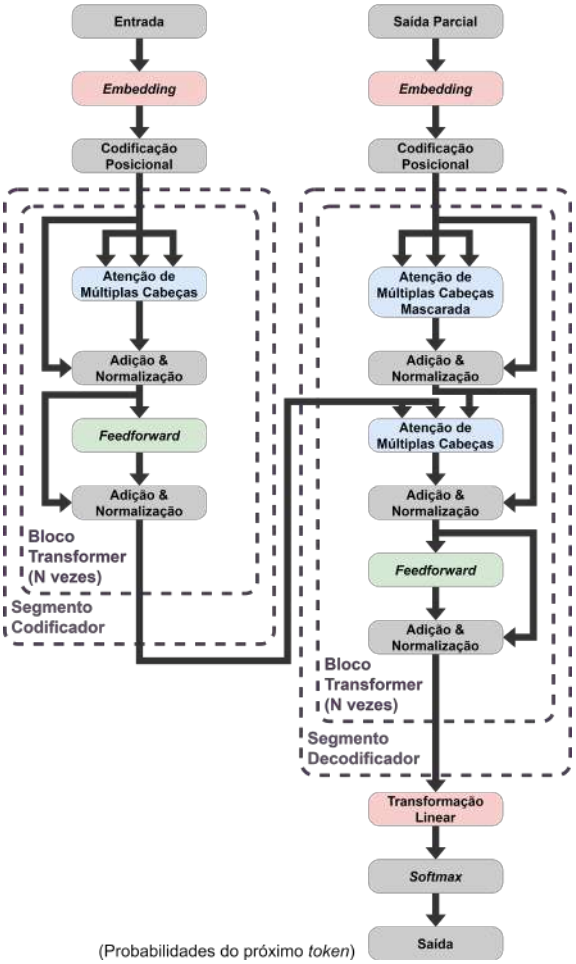
Além dos conceitos de atenção e auto atenção, outro conceito importante para compreender o funcionamento dos Transformers são os *embeddings*. Os *word embeddings* (que podem ser adaptados do inglês como "raízes das palavras") são a solução para um problema na área de processamento de linguagem natural: a representação de palavras e frases. Conforme previamente estabelecido, a representação dos dados é um passo fundamental em aprendizado de máquina e, mesmo que aprendizado profundo permita trabalhar com dados mais brutos, ainda assim é preciso representar os dados matematicamente para que o modelo possa processá-los. No caso de visão computacional, a forma usual de representação digital de imagens é satisfatória, mas, no caso de linguagens, uma sequência de caracteres não é suficiente para transmitir informações sobre a semântica das palavras. Uma outra representação consiste em definir um vocabulário contendo todas as palavras, ou *tokens*, que o modelo "entende" e representar cada palavra como um vetor *one-hot* (um vetor de tamanho igual ao tamanho do vocabulário onde todos os valores são zero, exceto o valor no índice associado à palavra que é igual a 1), mas essa representação é muito esparsa, dificultando o processamento. Os *embeddings* são uma forma de "aprendizado de representação" (*representation learning*) que utilizam técnicas de *machine* ou *deep learning* para produzir uma representação mais densa das palavras de um determinado vocabulário. Eles também são úteis como uma forma de *transfer learning*.

Além de transformar as sentenças em sequências de vetores de *embeddings*, os Transformers também precisam codificar a posição das palavras na frase, visto que a atenção não possui nenhuma capacidade inata de levar em consideração a posição das palavras. É possível utilizar codificações posicionais fixas ou aprendíveis. Nos Transformers apresentados por Vaswani et al. (2017), a posição foi codificada fixamente através de funções senoidais com diferentes frequências adicionadas à sequência de *embeddings*.

A arquitetura de um Transformer, representada na Figura 2, é composta por um segmento codificador e um segmento decodificador. Ambos os segmentos recebem, como

entrada, sentenças devidamente codificadas: o segmento codificador recebe a sentença de entrada do modelo, e o segmento decodificador recebe como entrada a estado atual da sentença de saída, cujo próximo *token* ele deverá prever. A entrada do bloco decodificador é deslocada para a direita, ou seja, tem um novo *token* inserido no início da frase. Os segmentos são compostos por um mesmo número N de blocos, onde o primeiro bloco recebe a entrada do segmento e cada bloco consecutivo recebe, como entrada, a saída do bloco anterior (VASWANI et al., 2017).

Figura 2 – Arquitetura do Transformer (VASWANI et al., 2017).



Fonte: acervo do autor.

Cada bloco do segmento codificador do Transformer é composto por dois sub blocos consecutivos: o primeiro contém a auto atenção de múltiplas cabeças, e o segundo é composto por uma rede *feedforward* pontual de duas camadas com uma ativação ReLU entre elas. Ambos os blocos são contornados por conexões residuais, onde a entrada do bloco é somada à sua saída, e o resultado é normalizado antes de prosseguir. Os blocos do decodificador são semelhantes, possuindo apenas duas diferenças notáveis, a atenção do primeiro sub bloco é mascarada para impedir que o modelo leve em conta *tokens* futuros (ainda não previstos), e é adicionado um novo sub bloco imediatamente após o primeiro

que recebe a saída deste como *query* e a saída do segmento codificador como valor e chave (VASWANI et al., 2017).

Esta versão particular da arquitetura é adaptada para tarefas que requeiram a produção de uma outra frase como resposta, mas também é possível adaptar os Transformers para outros tipos de problema, como classificação (DEVLIN et al., 2018).

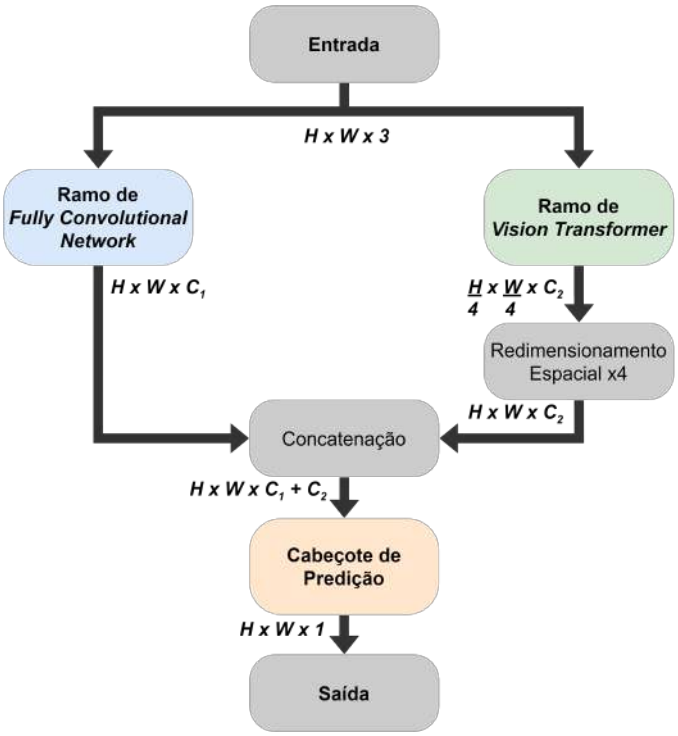
O Transformer e suas variações tornaram-se rapidamente o estado da arte em tarefas de processamento de linguagem natural, e despertaram o interesse em aplicações de mecanismos de atenção em outras áreas de *deep learning*. Em visão computacional, a atenção era relegada a um papel auxiliar à convolução, até a apresentação de uma abordagem puramente Transformer para processamento de imagens por Dosovitskiy et al. (2020). Neste trabalho, foi introduzido o conceito de *patch embeddings* (adaptado liberalmente do inglês como raízes de fragmento), onde a imagem é particionada em um número de fragmentos 2D de tamanho fixo que são achatados em um vetor 1D, e tratados como uma sequência de *tokens* para processamento por um Transformer. Além disso, utilizam-se *embeddings* posicionais aprendíveis unidimensionais. O chamado Vision Transformer, ou ViT, alcançou ou superou o estado da arte em várias tarefas de classificação de imagens, apesar de ser altamente dependente de pré-treinamento.

O FCB Former é uma arquitetura que combina redes inteiramente convolucionais e Vision Transformers, visando mitigar a principal fraqueza de modelos para segmentação semântica com ViTs: o fato de que eles só conseguem produzir máscaras de segmentação com dimensões 4 vezes menores do que as da imagem de entrada, devido ao processo de tokenização e *embedding* da imagem. Esta combinação é feita através de um sistema de ramos, representado na Figura 3. A entrada é passada simultaneamente para um ramo de Vision Transformer (ViT) e um ramo de *Fully Convolutional Network* (FCN), cujas saídas são combinadas e processadas por um terceiro módulo, o cabeçote de predição. Desta forma, o modelo utiliza a segmentação produzida pela FCN para refinar a segmentação produzida pelo ViT e vice-versa, aproveitando as vantagens de ambos (SANDERSON; MATUSZEWSKI, 2022).

O FCB - Swin V2 Transformer, por sua vez, utiliza a mesma estrutura geral do FCB Former bem como as estruturas internas do ramo de FCN e do cabeçote de predição. Porém, ele propõe um ramo de ViT completamente novo, utilizando o Swin V2 Transformer (LIU et al., 2022), e alguns outros pequenos ajustes que serão detalhados a seguir.

Os três módulos propostos por Sanderson e Matuszewski (2022) utilizam, repetidas vezes em sua construção, o Bloco Residual. Conforme representado na Figura 4.a, o bloco residual proposto por Sanderson e Matuszewski (2022) é composto por uma camada de normalização de grupo com número de grupos 32, uma ativação SiLU, e uma camada de convolução 2D 3x3, seguida por uma repetição destas 3 camadas, e, por fim, uma conexão residual de salto (*skip connection*) por adição que pula toda a

Figura 3 – A estrutura geral do FCB Former, com dois ramos paralelos que se unem para produzir a segmentação (SANDERSON; MATUSZEWSKI, 2022).



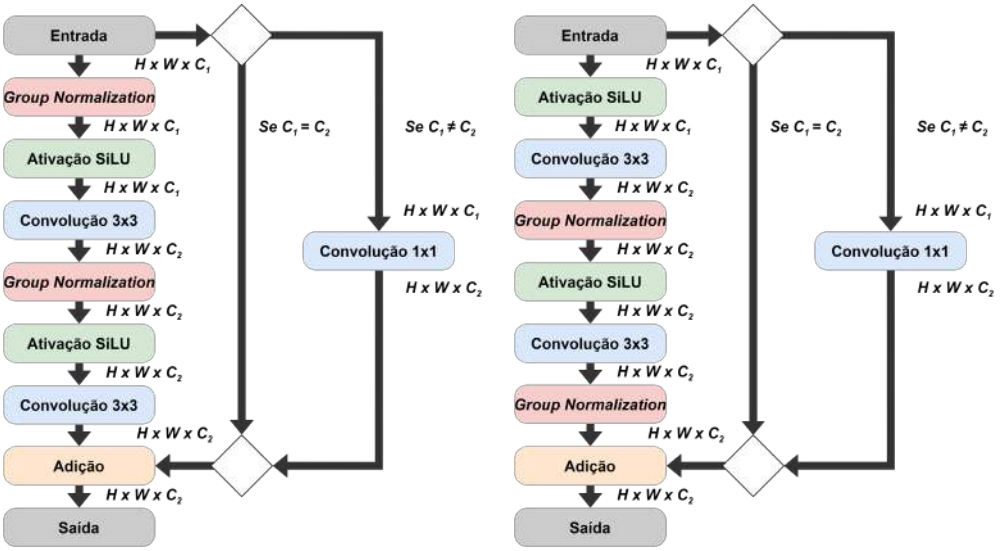
Fonte: acervo do autor.

estrutura. Fitzgerald e Matuszewski (2023), por sua vez, propôs uma reestruturação deste componente (representada na Figura 4.b), de tal forma que ele seja consistente com a abordagem de *residual post normalization* usada pelo Swin Transformer V2 (LIU et al., 2022). A variável C_2 é o número de filtros do bloco e varia de acordo com a instância de Bloco Residual em questão.

A arquitetura do ramo de ViT utilizado por Sanderson e Matuszewski (2022) (representada na Figura 5) é primariamente baseada na arquitetura do SSFormer, abordagem para segmentação em tamanho reduzido com ViT, porém com atualizações no segmento decodificador, o *Progressive Locality Decoder* (PLD), e seus componentes: *Local Emphasis* (LE) e *Stepwise Feature Aggregation* (SFA).

Como codificador, foi utilizado o Pyramid Vision Transformer v2 (PVTv2), que retorna um pirâmide de características em 4 níveis e utiliza patch *embeddings* sobrepostos ao invés de *embeddings* posicionais rígidos. O PVTv2 foi utilizado em sua variante B3 e com pesos pré-treinados na ImageNet. Os níveis possuem dimensões espaciais decrescentes e dimensão de canais crescente de acordo com sua profundidade: a cada nível, a altura e largura são reduzidos pela metade, e o número de filtros é dobrado (com exceção do terceiro nível que possui 320 filtros ao invés de 256). Mesmo o nível mais alto, possui apenas um quarto das dimensões espaciais, sendo 16 vezes menor que a imagem original.

Figura 4 – O bloco residual e suas camadas, conforme descrito por Sanderson e Matuszewski (2022) (a), e a reorganização das camadas proposta por Fitzgerald e Matuszewski (2023) (b).



(*a*) (*b*)

Fonte: acervo do autor.

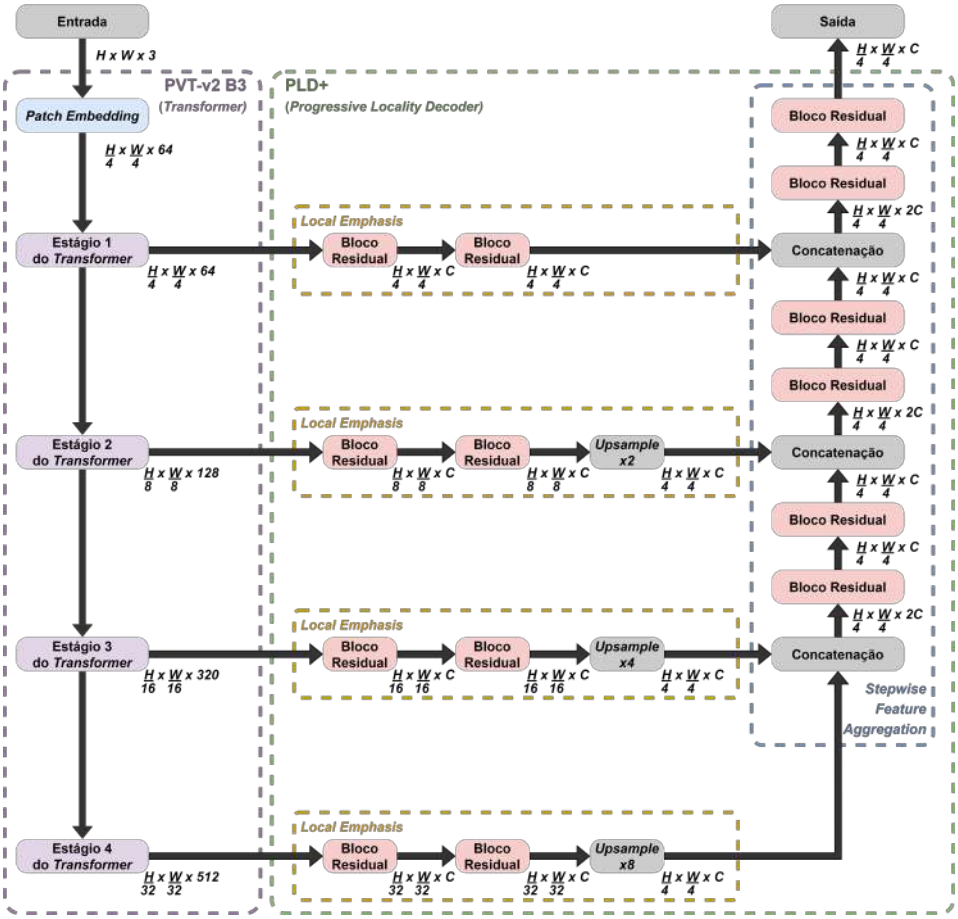
Isso se deve ao processo de *patch embedding* e é motivo do SSFormer e o ramo de ViT só poderem produzir segmentações em tamanho reduzido.

O PLD+, versão do segmento decodificador proposta por Sanderson e Matuszewski (2022), recebe a pirâmide de características e a condensa em um grupo de C mapas de características, sendo C igual a 64. Primeiramente, cada nível da pirâmide é processado por um segmento de LE, que consiste em 2 blocos residuais consecutivos com número de filtros C e, por fim, uma camada de *upsampling* que iguala as dimensões espaciais da saída às do primeiro nível, sendo *upsampling* $\times 8$ no quarto nível, $\times 4$ no terceiro, $\times 2$ no segundo, e apenas uma operação de identidade (vazia) no primeiro. Após este processamento, os 4 níveis são agregados passo a passo no bloco de SFA, que consiste em 3 etapas idênticas de concatenação de canais seguida por uma sequência de 2 blocos residuais com número de filtros C. A última etapa produz a saída do ramo de ViT: um conjunto de mapas de características de resolução grosseiras que serão refinados pelo cabeçote de predição com a ajuda dos mapas de características produzidos pelo ramo de FCN.

O ramo de ViT apresentado por Fitzgerald e Matuszewski (2023), e utilizado neste trabalho, é mais semelhante a uma arquitetura U-Net tradicional, onde as dimensões espaciais aumentam gradualmente ao longo do decodificador, espelhando o codificador, ao invés do sistema de ênfase local utilizado por Sanderson e Matuszewski (2022). A Figura 6.a representa a arquitetura deste ramo.

Como codificador, utiliza-se o Swin Transformer V2 (LIU et al., 2022), variante B (base) com janela de tamanho 24. Assim como o PVTv2 B3, ele possui 4 estágios, cujas

Figura 5 – Ramo de ViT utilizado por Sanderson e Matuszewski (2022).



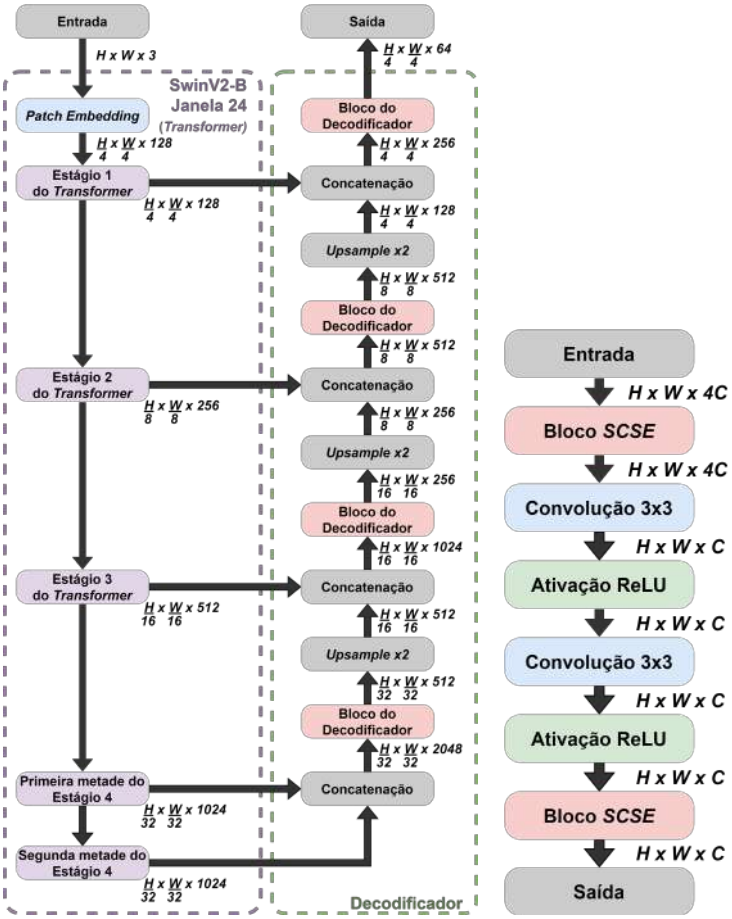
Fonte: acervo do autor.

dimensões espaciais reduzem-se pela metade, e a dimensão de canais dobra, a cada nível de profundidade. O primeiro estágio possui um quarto das dimensões espaciais da imagem e 128 filtros e o último possui um 32 avos das dimensões espaciais e 1024 filtros. A saída de cada estágio é utilizada como conexão entre o codificador e o decodificador, com exceção do último estágio, que possui uma conexão partindo de seu ponto intermediário (saída do primeiro bloco Transformer) além da conexão em sua saída que dá início ao decodificador.

O segmento codificador inicia-se com a saída do quarto estágio, e recebe cada uma das 4 outras conexões em ordem descendente de profundidade. Cada conexão é recebida através de concatenação de canais com saída da última etapa, seguida por um bloco de arquitetura especial apresentado por Fitzgerald e Matuszewski (2023) chamado de bloco do decodificador, que reduz o número de filtros a um quarto da entrada.

O bloco do decodificador, representado na Figura 6.b, é uma sequência de 4 etapas: um bloco SCSE (detalhado mais à frente), duas convoluções 2D com *kernel* 3 x 3, número de filtros igual a um quarto da entrada e ativação ReLU, seguidas por mais um bloco SCSE. Os mapas de características resultantes do bloco do decodificador têm suas dimensões espaciais dobradas via interpolação de vizinho mais próximo. Este processo de *upsampling*

Figura 6 – Estrutura geral do ramo de ViT que utiliza o Swin Transformer V2 (a) e a arquitetura do bloco do decodificador (b), ambos apresentados por Fitzgerald e Matuszewski (2023).



(*a*) (*b*)

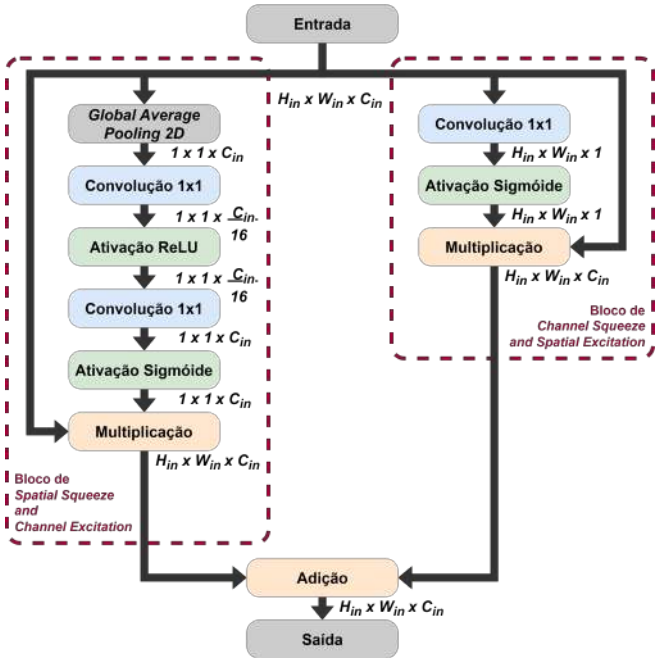
Fonte: acervo do autor.

não ocorre na última conexão, onde a saída do bloco decodificador é também a saída do ramo.

O bloco de *concurrent spatial and channel squeeze and excitation* (SCSE) utilizado neste ramo (representado na Figura 7) foi proposto por Roy, Navab e Wachinger (2018) e tem a função de recalibrar os dados ao longo dos eixos espaciais e do eixo de canais de acordo com ponderações calculadas de forma aprendível. Ele consiste em dois sub blocos paralelos: o *spatial squeeze and channel excitation* (cSE) que realiza a ponderação dos canais, e o *channel squeeze and spatial excitation* (sSE) que realiza a ponderação espacial. O bloco cSE faz a compressão espacial através global pooling e a excitação de canais através de uma convolução 1 x 1 com ativação ReLU que reduz drasticamente o número de canais (um dezesseis avos da quantidade original), seguida por outra que expande os canais de volta a sua dimensão original e utiliza ativação sigmóide, resultando em um vetor com um peso entre 0 e 1 para cada canal. Esta ponderação é aplicada aos dados via multiplicação. O bloco sSE consiste apenas em um convolução 1 x 1 com número de filtros

igual a 1 e ativação sigmóide, cujo resultado é uma matriz que contém um peso entre 0 e 1 para cada posição espacial, esta ponderação espacial também é aplicada aos dados via multiplicação. Ambas as reavaliações são combinadas via adição, gerando a saída do bloco SCSE.

Figura 7 – O bloco SCSE conforme utilizado por Fitzgerald e Matuszewski (2023).

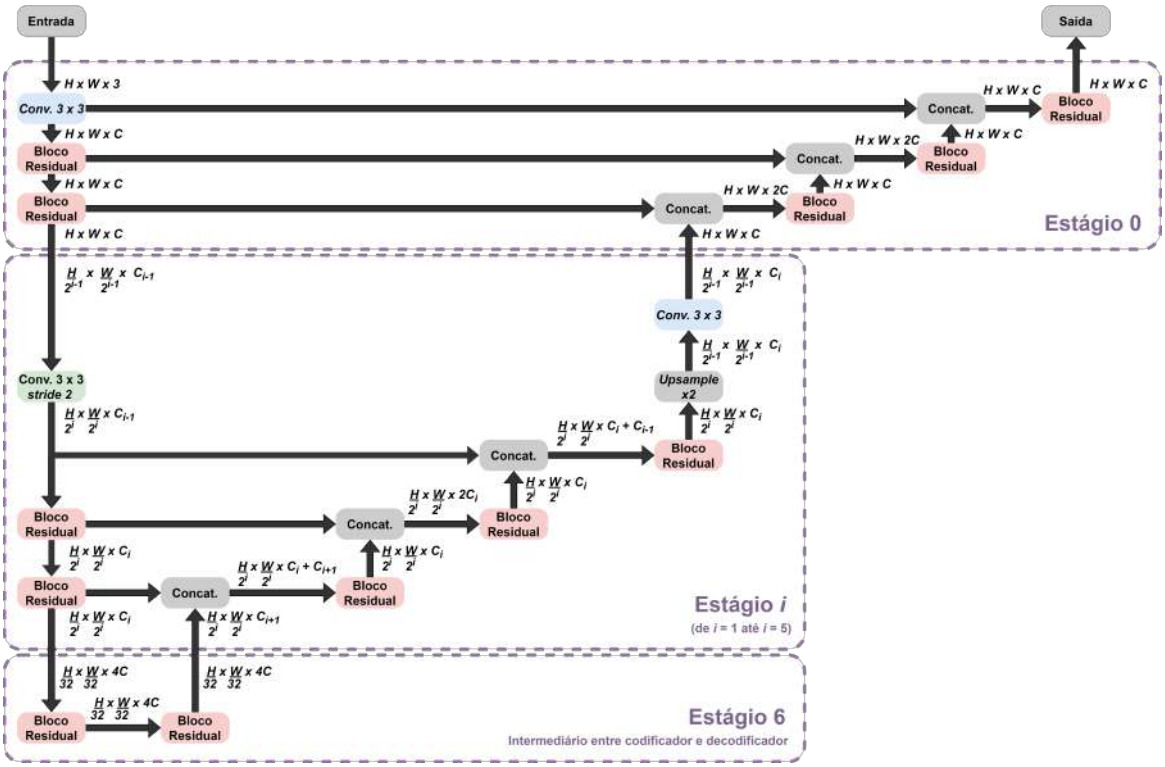


Fonte: acervo do autor.

O ramo de FCN representado na Figura 8, proposto por Sanderson e Matuszewski (2022) e reutilizado por Fitzgerald e Matuszewski (2023), possui uma estrutura que se assemelha a uma U-Net (RONNEBERGER; FISCHER; BROX, 2015), composta por um segmento codificador de múltiplos estágios, onde as dimensões espaciais diminuem a cada estágio enquanto a dimensão de canais aumenta, e um segmento decodificador, onde o oposto acontece, bem como conexões de salto entre os estágios correspondentes dos dois segmentos. Porém, enquanto a U-Net só possui uma conexão de salto por estágio, a arquitetura do ramo FCN possui 3, uma para cada sub bloco do estágio. O módulo FCN possui 7 estágios, o último dos quais é o estágio mais profundo, ponto intermediário entre os segmentos, totalizando 18 conexões de salto entre o codificador e o decodificador.

O lado codificador de cada estágio é composto por 3 sub blocos consecutivos: uma convolução 2D com *kernel* 3 x 3, e 2 blocos residuais. Em todos os estágios, exceto o primeiro, a convolução do primeiro sub bloco possui *stride* 2, reduzindo as dimensões espaciais de seu tensor de saída pela metade (*downsampling*). Todos os sub blocos do primeiro estágio possuem o mesmo número de filtros C, o número de filtros dos sub blocos seguintes aumenta para o dobro a cada 2 estágios, sendo C para o segundo, $2C$ para o terceiro e quarto, e $4C$ para o quinto e sexto. Porém os blocos de *downsampling* mantêm a

Figura 8 – Representação do ramo convolucional utilizado por Sanderson e Matuszewski (2022) e Fitzgerald e Matuszewski (2023).



Fonte: acervo do autor.

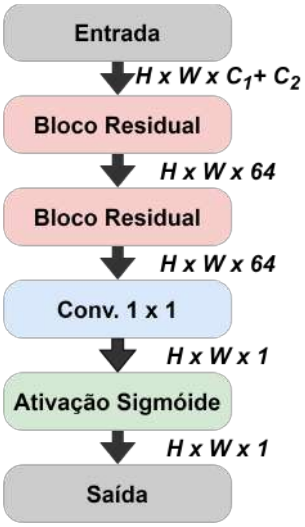
mesma quantidade de filtros do estágio anterior. O sétimo estágio é o ponto intermediário do fluxo principal da rede, sendo sucedido pelo lado decodificador do sexto estágio, e é composto por 2 blocos residuais em série, cujo número de filtros é igual ao do sexto estágio. Sanderson e Matuszewski (2022) definiu o número de filtros inicial $C = 32$, porém Fitzgerald e Matuszewski (2023) utilizou $C = 64$, a fim de ser consistente com o ramo ViT cuja saída possui 64 canais.

O lado decodificador de cada estágio é composto por 4 sub blocos consecutivos: 3 blocos residuais que recebem as conexões de salto dos blocos do lado codificador, e um bloco de *upsampling*. Os blocos residuais possuem o mesmo número de filtros ditado pelo estágio aos blocos do codificador e recebem, como entrada, o fluxo principal da rede e a conexão de salto vinda do codificador combinados através de concatenação ao longo do eixo dos canais. O bloco de *upsampling* é composto por uma interpolação de vizinho mais próximo com fator de escala 2 e uma convolução 2D com *kernel* 3 x 3 e o mesmo número de filtros que os blocos residuais que a antecederam. O primeiro estágio não possui um bloco de *upsampling* e a saída do seu terceiro bloco residual é repassada para o cabeçote de predição.

O ramo de FCN tem todos os seus pesos inicializados aleatoriamente antes do treinamento. Isso foi feito visando encorajá-lo a extrair as características necessárias para refinar a saída do ramo de ViT (SANDERSON; MATUSZEWSKI, 2022).

O cabeçote de predição, representado na Figura 9, é o terceiro e menor módulo da arquitetura FCB Former. Ele recebe como entrada as saídas de ambos os ramos, concatenadas ao longo do eixo dos canais, com a saída do ramo ViT tendo suas dimensões espaciais quadruplicadas via interpolação de vizinho mais próximo. O módulo consiste de 2 blocos residuais em série com número de filtros 64 seguidos por uma convolução 2D com *kernel* 1 x 1, número de filtros igual a 1 e ativação sigmóide, que produz uma matriz com dimensões $H \times W \times 1$ com valores entre 0 e 1. Esta é a saída final do modelo, onde cada item da matriz corresponde a um pixel da imagem original e contém a probabilidade, segundo a rede neural, de que este seja parte do alvo da segmentação, neste caso, um pólipo (SANDERSON; MATUSZEWSKI, 2022).

Figura 9 – Representação da arquitetura do cabeçote de predição (SANDERSON; MATUSZEWSKI, 2022).



Fonte: acervo do autor.

A função de *loss* utilizada por Fitzgerald e Matuszewski (2023) e vários outros trabalhos de segmentação é a soma da *binary cross entropy* (BCE) com a Dice *loss* (DL), definida na Equação 2.11. Onde P é a segmentação predita pelo modelo, V é a máscara verdadeira, e D é o coeficiente de Dice, definido na Seção 2.4 a seguir. Esta função é comumente utilizada em tarefas de segmentação por levar em conta tanto a acurácia da classificação pixel a pixel quanto a semelhança das áreas de segmentação.

$$DL(P, V) = 1 - D(P, V) \tag{2.11}$$

2.4 Validação dos Resultados

Após a etapa de segmentação, costuma-se fazer uma validação dos resultados produzidos, principalmente, para verificar o quanto o processo de segmentação conseguiu distinguir o objeto de interesse do restante da imagem. Neste trabalho, serão empregadas 4 métricas muito utilizadas na literatura no campo de segmentação de imagens biomédicas, que são: o coeficiente de Dice, o coeficiente de Jaccard, a precisão e a revocação.

Todas as métricas utilizadas trabalham com a noção de verdadeiros positivos (VP), falsos positivos (FP), e falsos negativos (FN). Onde VP é o número de pixels corretamente classificados como positivos, FP é o número de pixels erroneamente classificados como positivos, e FN é o número de pixels erroneamente classificados como negativos. Além disso, elas são definidas como funções com dois argumentos: a previsão do modelo P e a verdade fundamental fornecida pela base V.

O coeficiente de similaridade de Dice, definido na Equação 2.12, também conhecido como coeficiente de Sørensen ou *F1-score*, foi proposto por independentemente por Dice (1945) e Sorensen (1948) como forma de medir a associação entre espécies. Hoje em dia é uma das principais, se não a principal, métricas para avaliação de segmentação de imagens biomédicas, sendo utilizada por todos os trabalhos no estado da arte.

$$D(P,V) = \frac{2 * VP}{2 * VP + FP + FN} \tag{2.12}$$

O coeficiente de Jaccard, definido na Equação 2.13, também chamado de intersecção sobre união ou IoU (*intersection over union*), foi formulado por Jaccard (1912), também para estudo ecológico e, assim como o coeficiente de Dice, é padrão na avaliação de algoritmos de segmentação biomédica.

$$J(P,V) = \frac{VP}{VP + FP + FN} \tag{2.13}$$

Precisão e revocação (ou *precision* e *recall*) são termos originados na área de recuperação de informação, referindo-se a métricas de avaliação para algoritmos de RI. Porém, ambas as métricas são regularmente utilizadas para avaliação de segmentação de imagens biomédicas.

A precisão, definida na Equação 2.14, mede a relevância dos resultados recuperados.

$$Prec(P,V) = \frac{VP}{VP + FP} \tag{2.14}$$

Enquanto a revocação, descrita na Equação 2.15, quantifica a parcela dos resultados relevantes que foram recuperados de fato.

$$Revoc(P,V) = \frac{VP}{VP + FN} \tag{2.15}$$

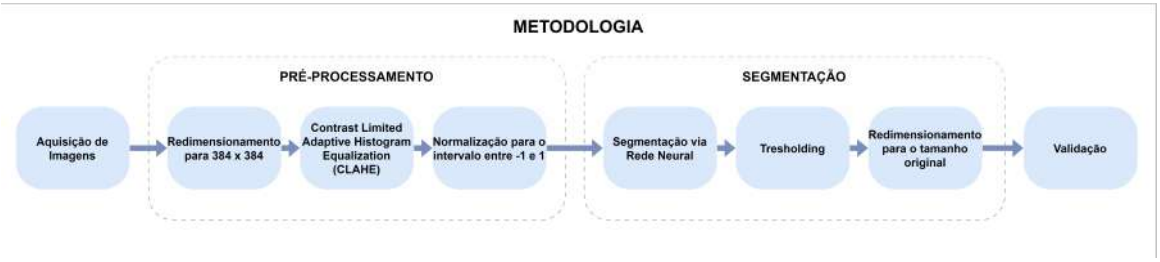
Logo, elas precisam ser visualizadas em conjunto para permitir uma avaliação adequada do método em questão. Mas também é possível uni-las em uma única métrica através de sua média harmônica, que resulta no *F1-score*, equivalente ao coeficiente de Dice.

Este capítulo apresentou todo o referencial teórico necessário para entender o desenvolvimento deste trabalho. No próximo capítulo, será detalhada a metodologia proposta para a segmentação de pólipos em imagens de colonoscopia

3 Metodologia

Neste capítulo será descrita a metodologia proposta para segmentação de pólipos em imagens de colonoscopia desenvolvida e testada ao longo deste trabalho. A Figura 10 apresenta as etapas da metodologia, que são: aquisição de imagem, pré-processamento, segmentação e validação de resultados.

Figura 10 – Etapas da Metodologia Proposta.



Fonte: acervo do autor.

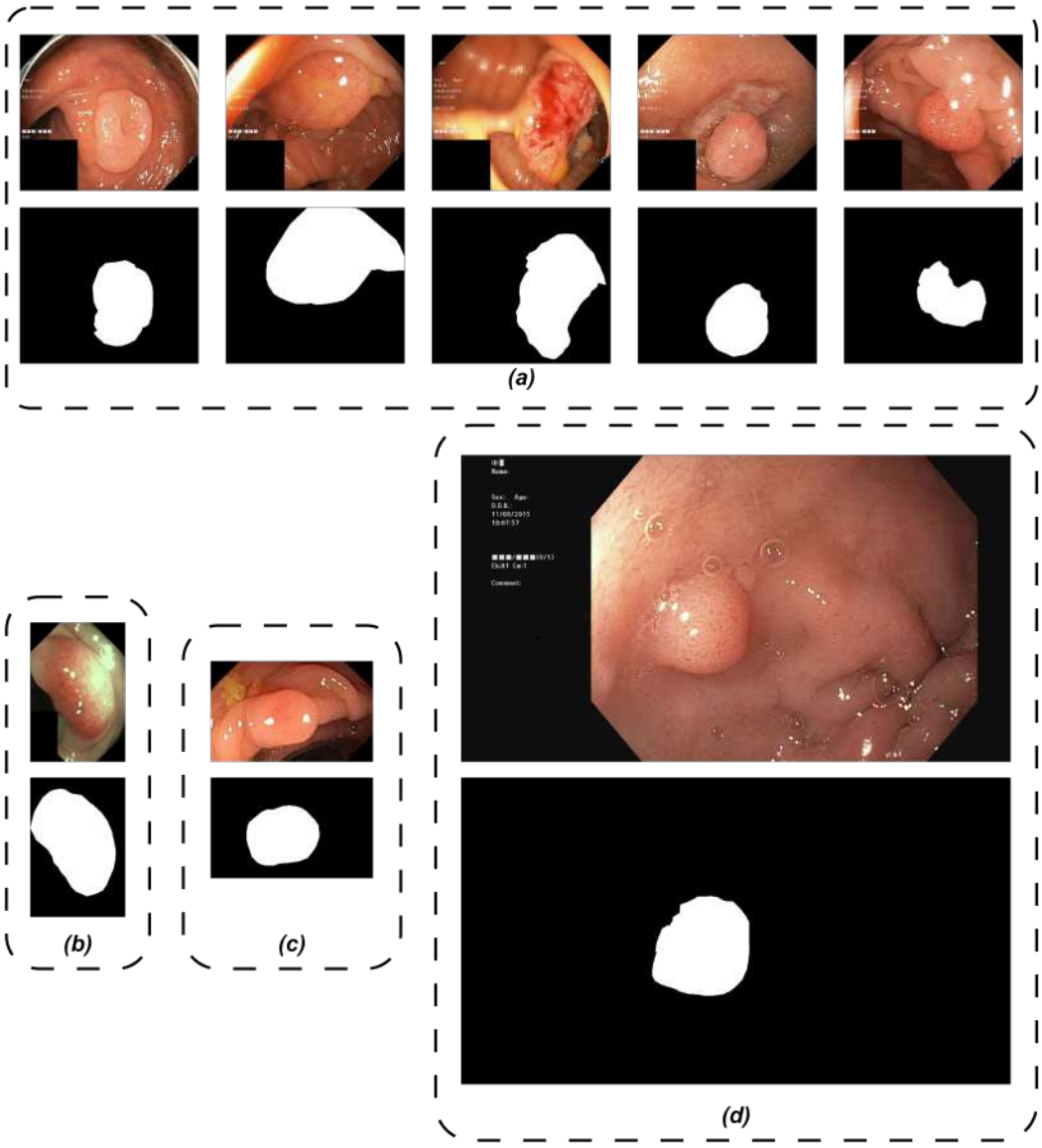
3.1 Aquisição de Imagens

A primeira etapa da metodologia foi dedicada à obtenção das imagens de colonoscopia que foram empregadas nos testes. Dessa forma, utilizou-se a base pública, disponível na internet, Kvasir-SEG (JHA et al., 2020). Esta base é formada por imagens de colonoscopia com máscaras de segmentação correspondentes indicando a presença de pólipos. As anotações foram produzidas por profissionais da saúde, sendo anotadas por um médico e verificadas por um gastroenterologista. Além disso, essa base foi escolhida pela sua quantidade significativa de imagens, sua facilidade de uso e acesso, e sua popularidade na literatura. No total, esta base contém 1000 imagens de tamanho variado, todas no formato RGB, sendo tamanho médio de 545 x 625 pixels e o tamanho moda de 530 x 622 pixels. A Figura 11 apresenta alguns exemplos de imagens dessa base, cujas as dimensões são: 546 x 625 para as imagens na Figura 11.a, 487 x 332 para a Figura 11.b (que possui a menor área e largura da base), 352 x 568 para a Figura 11.c (que possui a menor altura da base), e 1072 x 1920 para a imagem na Figura 11.d (que possui a maior altura e maior largura da base.

3.2 Pré-processamento

Para facilitar a etapa de treinamento, todas as imagens da base foram redimensionadas, usando a interpolação por área. Assim como no trabalho de Fitzgerald e

Figura 11 – Algumas imagens médias (a) e as mais atípicas (b, c, d) da base Kvasir-SEG comparadas em escala.

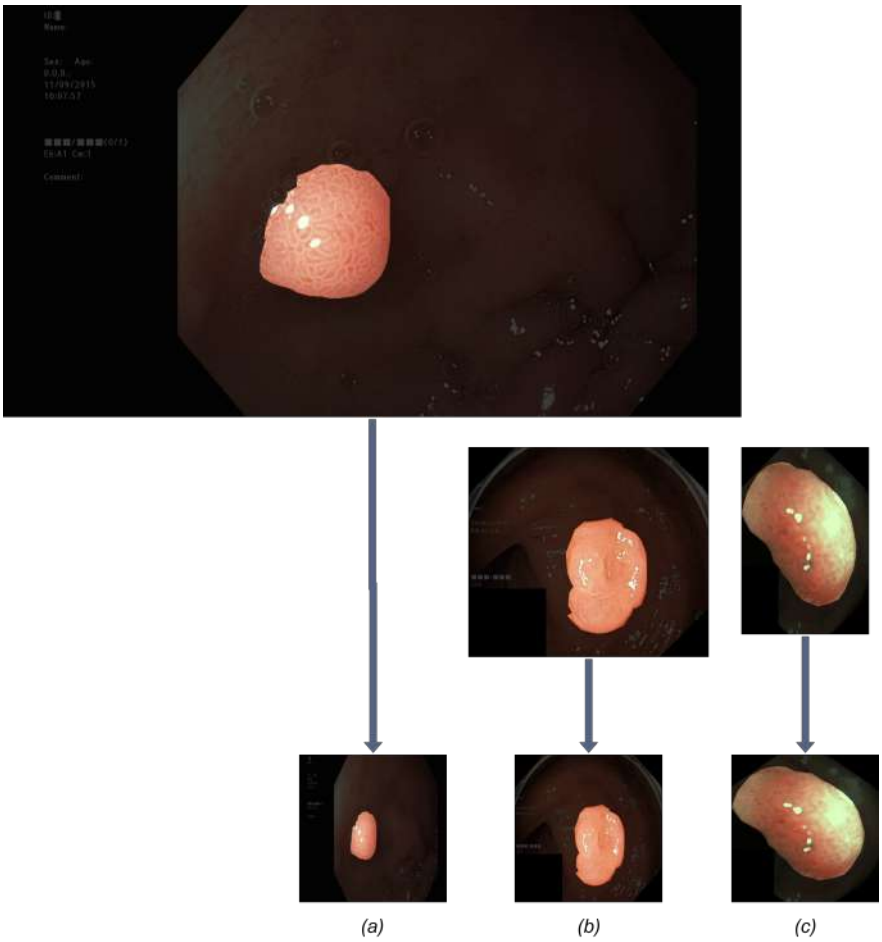


Fonte: (JHA et al., 2020).

Matuszewski (2023) foi utilizado o tamanho de 384 x 384 pixels. Esse tamanho foi escolhido devido à disponibilidade de pesos pré-treinados do Swin-Transformer V2 (LIU et al., 2022) especificamente para esse tamanho. A Figura 12 mostra o efeito deste redimensionamento em imagens diferentes.

Após o redimensionamento, foi aplicada a técnica de processamento de imagens CLAHE em cada um dos 3 canais de cada imagem. Os parâmetros da CLAHE, limite de contraste (*clip limit*) e tamanho do ladrilho (*tile grid size*) foram escolhidos como 2 e 8 x 8, respectivamente. Durante o treinamento, a CLAHE foi aplicada após a realização das transformações de augmentação. Esta técnica foi escolhida durante as etapas iniciais da pesquisa, onde foram levantadas, além de técnicas de *deep learning*, técnicas de realce de

Figura 12 – Exemplos do redimensionamento em 3 imagens diferentes: a menor imagem da base (a), uma imagem média (b), e a maior de todas (c).



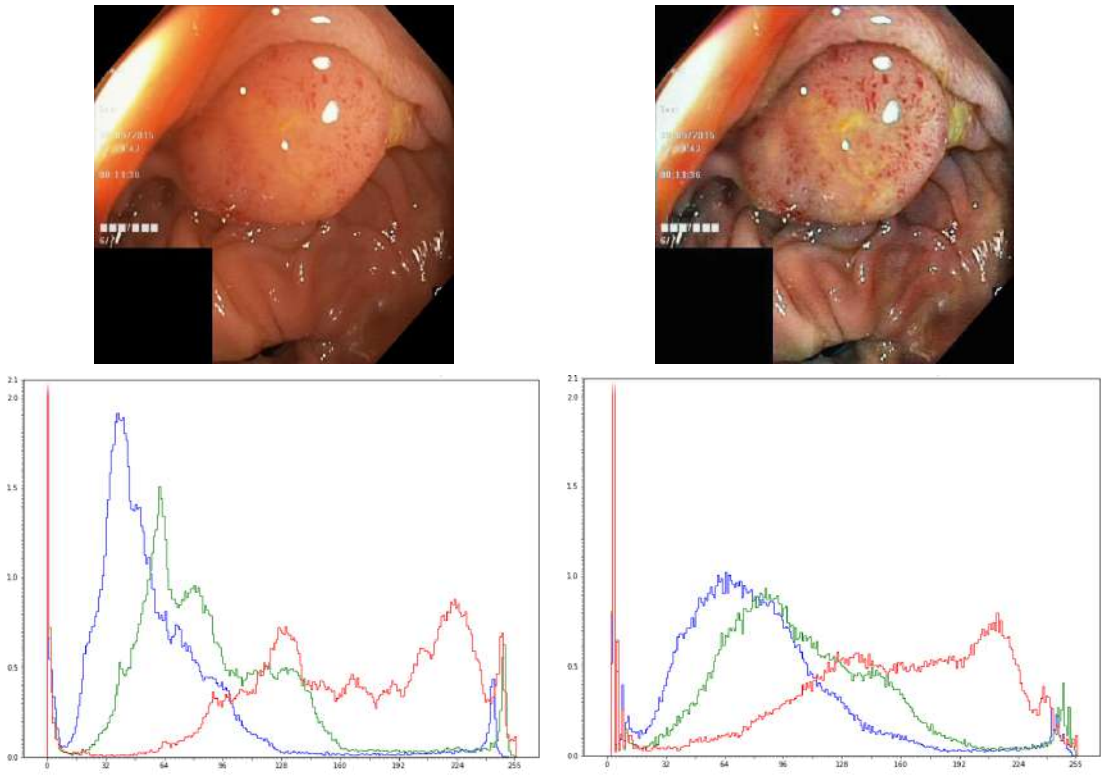
Fonte: (JHA et al., 2020).

imagens, dentre as quais a CLAHE é notável. Nos testes preliminares a CLAHE demonstrou resultados benéficos e, portanto, foi adotada como padrão ao longo do desenvolvimento do trabalho.

A Figura 13 mostra, como exemplo, uma imagem da base antes da aplicação do CLAHE (Figura 13.a) e, então, a mesma imagem após a aplicação da técnica (Figura 13.b). Além disso, a figura inclui os histogramas dos canais de cor de cada imagem logo abaixo delas. Para fins de legibilidade, em ambos os histogramas, a escala ignora os valores extremamente atípicos. Estas altas concentrações na faixa mais escura se devem às bordas pretas da imagem que ocupam cerca de 15% dos pixels. Antes da aplicação da CLAHE, essa concentração ocorre no valor de brilho 0 e após a aplicação ela ocorre no valor de brilho 3.

Após a aplicação de todos os pré-processamentos, as imagens foram normalizadas de uma distribuição entre 0 e 255 para uma distribuição entre -1 e 1. A normalização da distribuição padrão das imagens digitais para uma distribuição [-1, 1] ou [0, 1] é padrão

Figura 13 – Exemplo da imagem e seus histogramas antes (a) e depois da aplicação do CLAHE (b).



(*a*) (*b*)

Fonte: (JHA et al., 2020) e acervo do autor.

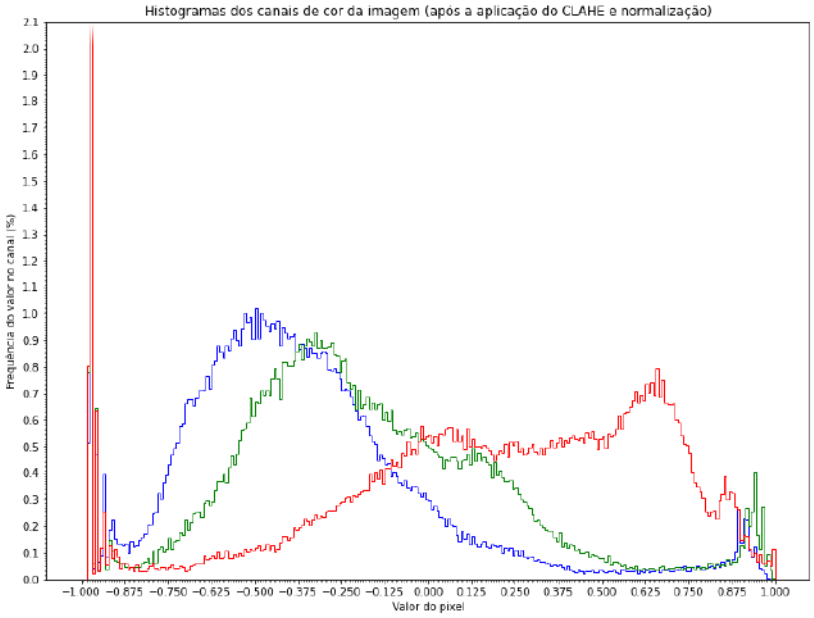
no processamento de imagens com redes neurais, pois esta primeira gera obstáculos na adequação da rede neural aos dados. Isso ocorre pois os valores nessa distribuição são muito altos se comparados aos valores com os quais as redes trabalham internamente, principalmente no caso de uma rede cuja saída é restrita à distribuição [0, 1], como é o caso de modelos de classificação e, portanto, segmentação.

A Figura 14 mostra o novo histograma da imagem (Figura 13.b após a normalização e logo antes de ser processada pela rede neural. O histograma possui 256 classes uniformes entre -1 e 1. Assim como na figura (Figura 13.b, a escala ignora a quarta classe (-0.9765625 a -0.96875, equivalente ao nível de brilho 3) devido a seu caráter atípico.

3.3 Segmentação

O objetivo desta etapa é propor uma arquitetura de rede neural que seja capaz de separar automaticamente o pólipo do resto da imagem. Este tipo de processamento automático pode ser aplicado para auxiliar durante a realização e revisão de exames de colonoscopia e, possivelmente, durante a cirurgia de remoção de pólipos. Os pesos do modelo são ajustados ao longo do treinamento de acordo com a diferença computada pela

Figura 14 – Histogramas dos canais de cor da Figura 13.b após a normalização.



Fonte: acervo do autor

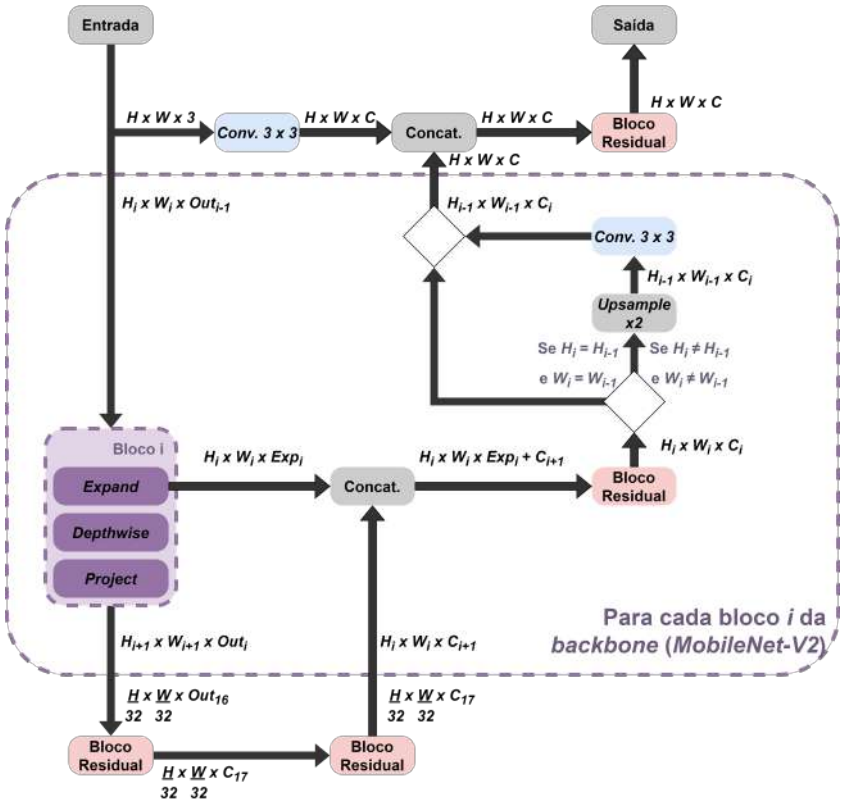
função de *loss* entre a segmentação produzida e a verdade fundamental fornecida pela base.

A arquitetura do modelo utilizado nesta metodologia é uma variação do modelo utilizado por Fitzgerald e Matuszewski (2023), FCB - Swin V2 Transformer, que, por sua vez, é um aprimoramento da arquitetura proposta por Sanderson e Matuszewski (2022), FCB Former.

Neste trabalho, tenta-se avaliar se o efeito de aplicar *transfer learning* com uma *backbone* pré-treinada, nos moldes de abordagens tradicionais para segmentação com arquitetura U-Net, ao ramo convolucional. Para tal, decidiu-se substituir o segmento codificador do ramo de FCN proposto por Sanderson e Matuszewski (2022) por uma Mobile Net V2 (DONG et al., 2020) com pesos pré-treinados na ImageNet, adaptando a arquitetura conforme o necessário. A arquitetura do ramo de FCN proposto pode ser vista na Figura 15. As arquiteturas do ramo de ViT e do cabeçote de predição foram mantidas iguais às utilizadas por Fitzgerald e Matuszewski (2023), previamente detalhadas na Seção 2.3.

A Mobile Net V2 (DONG et al., 2020) foi escolhida devido a sua boa relação entre desempenho e custo computacional, sendo a menor e mais rápida *backbone* dentre as disponibilizadas por padrão no keras enquanto ainda possui desempenho comparável ao das outras, apesar de menor (KERAS, 2015). A Mobile Net V2 não possui pesos pré-treinados na ImageNet com entradas de tamanho 384 x 384, portanto foram utilizados os pesos para entradas 224 x 224.

Figura 15 – Ramo de FCN proposto neste trabalho, com a Mobile Net V2 (DONG et al., 2020) como *backbone*.



Fonte: acervo do autor

A Mobile Net V2 é composta por 17 blocos distribuídos ao longo de 5 estágios: o primeiro e segundo estágios contém 2 blocos cada, o terceiro contém 3, o quarto contém 7, e o quinto contém 3 novamente. Os estágios possuem dimensões espaciais decrescentes (reduzidas pela metade a cada estágio, já começando com a metade da imagem original no primeiro estágio) e dimensão de canais crescentes (sem progressão regular). Todos os blocos são divididos internamente em 3 etapas: *expand*, *depthwise* e *project*, cada etapa sendo composta por uma convolução (convolução *depthwise* na etapa assim nomeada), normalização por lotes e ativação ReLU. O último bloco de cada estágio, exceto o do quinto, realiza o *downsampling* entre a etapa *expand* e a etapa *depthwise*. Para adaptá-la como um codificador semelhante ao proposto por Sanderson e Matuszewski (2022), utilizou-se a saída da etapa *expand* de cada bloco como ponto de conexão entre o codificador e o decodificador e manteve-se a arquitetura do decodificador quase inalterada.

O número de filtros dos blocos residuais do decodificador são definidos pela mesma progressão a cada estágio utilizada por Sanderson e Matuszewski (2022), partindo do número de filtros inicial C. Porém, o número de grupos da normalização em grupo destes blocos precisou ser reduzido para 16, pois nem todas as conexões com a *backbone* possuíam

número de filtros divisível por 32. Ao receber a última conexão de cada estágio, é feito o *upsampling* da mesma maneira da arquitetura de ramo convolucional prévia.

Buscando assemelhar a arquitetura ao utilizado por Sanderson e Matuszewski (2022) e Fitzgerald e Matuszewski (2023), foi adicionado um novo estágio no "topo" da arquitetura que trabalha com as mesmas dimensões espaciais da imagem, pois a Mobile Net V2 começa seu processamento com dimensões já reduzidas. O novo estágio contém apenas um bloco residual com número de filtros igual a C e é paralelo à *backbone*, sendo a última conexão recebida pelo decodificador. Assim têm-se um total de 18 conexões entre o codificador e o decodificador. Neste trabalho foi utilizado C igual a 64, conforme feito por Fitzgerald e Matuszewski (2023).

A função de *loss* utilizada nesta metodologia substitui a função Dice *loss* pela Jaccard *logarithmic loss* (JLL), que é definida na Equação 3.1, onde J é a função do coeficiente de Jaccard, previamente definido na Seção 2.4.

$$JLL(P, V) = -\ln[J(V, P)] \tag{3.1}$$

Essa função foi escolhida por apresentar melhor desempenho durante o desenvolvimento dos experimentos. Acredita-se que isso se deva à maior sofisticação do mecanismo de logaritmo em comparação com uma simples subtração, que aumenta significativamente a *loss* em casos de desempenho muito baixo e a reduz em casos de desempenho próximo à perfeição, e ao uso da métrica do coeficiente de Jaccard, que compartilha a sensibilidade à forma da segmentação presente no coeficiente de Dice, porém não atribui tanto peso aos verdadeiros positivos, o que a torna mais rígida no contexto de segmentação com *deep learning*. Assim, caberá a trabalhos futuros uma investigação mais aprofundada para confirmação ou não dessas hipóteses.

A saída da rede neural é uma máscara com valores contínuos entre 0 e 1 (ou seja, é a probabilidade de cada pixel ser parte do pólipo) e dimensões iguais às da entrada da rede. Então, esta saída é redimensionada para o tamanho original da imagem utilizando o mesmo método de interpolação que foi usado para padronizar o tamanho das imagens no início da metodologia. Após este redimensionamento, a máscara é discretizada, de tal forma que cada pixel possua valor igual a 0 ou igual a 1. Isso é feito através de um processo de *thresholding* onde todo pixel cujo valor é maior que 0,5 é considerado como 1, e o restante é considerado como 0. O resultado deste processo é considerado a máscara final para fins de comparação com a máscara verdadeira.

3.4 Validação dos Resultados

A etapa final de Validação dos Resultados visa não só medir o desempenho da metodologia proposta, mas também discutir possíveis melhorias. Assim, a base de imagens foi particionada nos subconjuntos de treino, de teste e de validação. Foi usada a seguinte configuração: treinamento, com 800 imagens (80% da base); validação, com 100 (10% da base), e teste, também com 100 imagens (10% da base). A divisão foi feita de acordo com a divisão realizada por Fitzgerald e Matuszewski (2023), ordenando as imagens da base em ordem alfabética dos nomes de arquivo e separando as primeiras 800 imagens no conjunto de treino, as 100 seguintes no de validação, e o restante no de teste. Esse método foi escolhido para propósitos de comparação com outras metodologias existentes na literatura. Além disso, para validar os resultados dos experimentos utilizou-se as métricas: coeficiente de Dice, coeficiente de Jaccard, precisão (*precision*), e revocação (*recall*), conforme descritas na Seção 2.4.

Este capítulo apresentou e descreveu detalhadamente a metodologia proposta para segmentação semântica de pólipos em imagens de colonoscopia. No capítulo seguinte, serão apresentados e discutidos os resultados obtidos a partir da aplicação desta metodologia.

4 Resultados

Neste capítulo, serão descritos e discutidos os resultados alcançados pela metodologia proposta, obtidos conforme descrito na Seção 3.4. A seguir serão detalhados o experimento em que foi aplicada a metodologia e os resultados obtidos, além de uma comparação com os trabalhos relacionados.

4.1 Descrição do Experimento

Durante o treinamento foi utilizada a augmentação, no intuito de reduzir o *overfitting*. Isso foi feito aplicando um conjunto de transformações aleatórias sobre cada imagem antes de permitir que o modelo utilize-as para treinar. Um novo conjunto de transformações é gerado aleatoriamente toda vez que o modelo acessa um lote de imagens (um conjunto para cada imagem), e as transformações são aplicadas após o redimensionamento mas antes das outras etapas do pré-processamento. Os conjuntos de transformações são divididos em transformações geométricas, que são aplicadas simultaneamente à imagem e à máscara, e transformações de cor, que são aplicadas somente à imagem.

O conjunto das transformações geométricas é gerado com os seguintes parâmetros: 50% de probabilidade de um *flip* horizontal e a mesma probabilidade de um *flip* vertical; uma operação de rotação cujo o ângulo é escolhido aleatoriamente dentro de uma faixa de -180º a 180º com amostragem uniforme; uma operação de translação cujas intensidades vertical e horizontal foram escolhidas aleatoriamente dentro de uma faixa de -12,5% a 12,5% da respectiva dimensão com amostragem uniforme (de -48 a 48 pixels no caso de dimensões 384 x 384); e uma operação de zoom cuja intensidade foi escolhida aleatoriamente dentro de uma faixa de -25% a 25% (aplicado de forma equivalente em ambas as dimensões).

E o conjunto das transformações de cor é gerado com os seguintes parâmetros: ajuste de brilho com delta selecionado aleatoriamente no intervalo entre -0,4 e 0,4 com amostragem uniforme; ajuste de contraste com fator selecionado aleatoriamente no intervalo entre 0,5 e 1,5 com amostragem uniforme; ajuste de saturação com fator selecionado aleatoriamente no intervalo entre 0,75 e 1,25 com amostragem uniforme; ajuste de matiz com delta aleatoriamente selecionado na faixa entre -0,01 e 0,01 com amostragem uniforme; e desfoque gaussiano com um kernel de tamanho 25 x 25 e sigma selecionado aleatoriamente no intervalo de 0,001 a 2 com amostragem uniforme. Esses parâmetros de augmentação foram escolhidos visando aproximar os processos descritos por Fitzgerald e Matuszewski (2023) e Sanderson e Matuszewski (2022).

O otimizador utilizado no treinamento foi o AdamW, com *learning rate* inicial de $1e^{-5}$, que é reduzida por um fator de 0,6 sempre que o modelo passa mais de 10 épocas sem melhorar a performance no conjunto de validação até um mínimo de $1e^{-7}$. O modelo foi treinado ao longo de 200 épocas com *batch size* 1, com os pesos do modelo sendo salvos ao final de épocas em que houve melhora da performance no conjunto de validação. Estes parâmetros também foram escolhidos visando reproduzir o descrito por Fitzgerald e Matuszewski (2023).

Todo o código foi implementado no *framework keras tensorflow* (CHOLLET et al., 2015), com auxílio do código disponibilizado por Sanderson e Matuszewski (2022), que precisou ser adaptado pois usa o *framework pytorch*, e do repositório disponibilizado por Leondgarse (2022), que oferece implementações e conversões de modelos populares de atenção para o *keras*. A execução dos treinamentos foi feita na plataforma do Google Colab, utilizando os ambientes de execução em GPU T4 e V100.

4.2 Resultados e Discussão

Ao final do treinamento, foram carregados os pesos que atingiram o melhor coeficiente de Dice no conjunto de validação ao longo do processo. Então, este modelo foi utilizado para produzir máscaras de segmentação sobre as 100 imagens do conjunto de teste. Cada uma destas máscaras foi redimensionada do tamanho padronizado de 384 x 384 para o tamanho original da imagem-objeto e, então, comparadas com as máscaras de segmentação dos especialistas ao longo das seguintes métricas: coeficiente de Dice, coeficiente de Jaccard, precisão (*precision*), revocação (*recall*), e acurácia. A média e desvio padrão de cada métrica ao longo das 100 imagens estão descritas na Tabela 2.

Tabela 2 – Resultados obtidos pela metodologia proposta

Métrica	Média	Desvio Padrão
Coeficiente de Dice	93,37%	10,5069%
Coeficiente de Jaccard (IoU)	88,91%	13,8623%
Precisão (Precision)	94,41%	12,0260%
Revocação (Recall)	93,96%	9,3370%
Acurácia	97,96%	3,4412%

Estes resultados foram considerados dentro do esperado em respeito às médias, porém os desvios padrão chamaram atenção devido a seus valores relativamente altos: cerca de 11% na maioria das métricas, muito acima das expectativas.

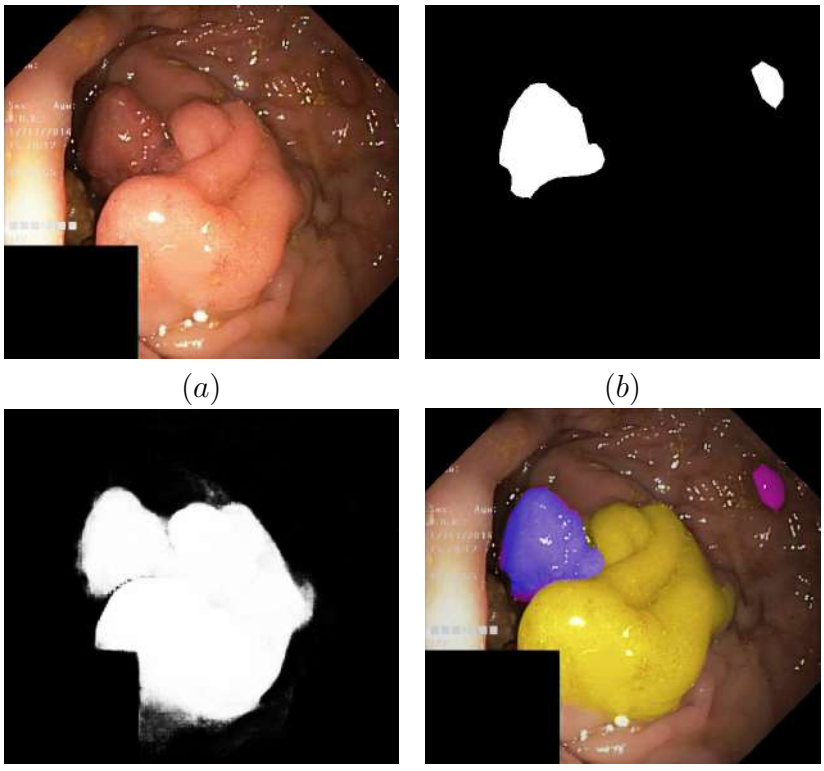
Após revisão dos casos individuais, foram identificados 3 casos que deveriam ser analisados mais profundamente. Neles, o modelo obteve os seus 3 piores desempenhos para as métricas de Dice e Jaccard, tendo mais de 20 pontos percentuais abaixo de todos os

outros casos em ambas as métricas. Cada um destes casos anômalos será apresentado e discutido individualmente a seguir, visando entender as causas dessa má performance.

Para ajudar na análise dos casos, foram produzidas representações do erro do modelo: imagens que possuem a interseção, em azul, e a diferença entre a máscara original, em magenta, e a previsão, em amarelo, sobrepostas na imagem original. Este tipo de imagem permite visualizar as regiões da imagem que acarretam maiores dificuldades para o modelo.

O pior caso, de acordo com os coeficientes de Dice e Jaccard, é referente à Figura 16.a. Nele, o modelo pontuou 31,93% no coeficiente de Dice e 19% no de Jaccard. Além disso, neste caso também ocorreram a pior precisão e a pior acurácia, de 19,62% e 77,72%, respectivamente. A revocação do modelo nesta imagem também foi baixa: 85,76%, a sétima pior da base de teste. A máscara verdadeira e a segmentação produzida pelo modelo podem ser vistas nas Figuras 16.b, e 16.c, respectivamente. A Figura 16.d é a representação visual do erro do modelo.

Figura 16 – Do pior caso, a imagem e máscara originais, e a segmentação prevista pelo modelo (a, b, e c, respectivamente), e uma representação de erro (d).



(a) (b)

(c) (d)

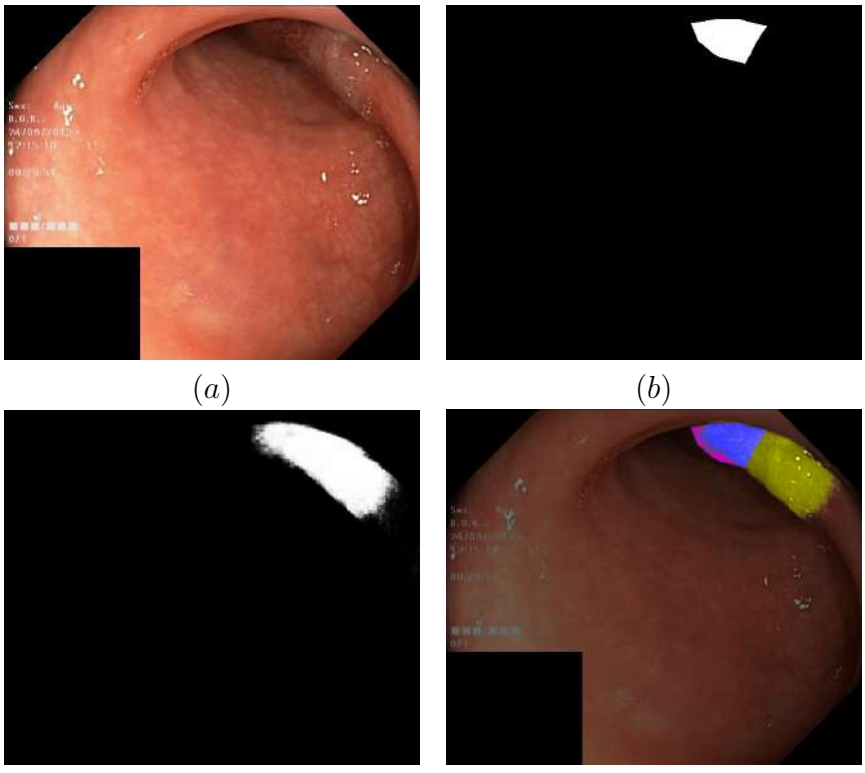
Fonte: (JHA et al., 2020) e acervo do autor.

Neste caso, podem-se elencar algumas características distintas possivelmente responsáveis pela dificuldade da rede em segmentar seu objeto. Um dos prováveis obstáculos é a presença de dois pólipos distintos e distantes um do outro, este é agravado por 3 outros

problemas: ambos os pólipos se encontram fora do foco espacial e de luminosidade da imagem; um dos pólipos é bem pequeno (que foi despercebido pelo modelo, prejudicando muito a revocação); e, mais notavelmente, a presença, central na imagem, de um conjunto de dobras das paredes intestinais que se assemelha visualmente a um pólipo, o que resultou na precisão baixíssima.

O segundo pior caso, pelo critério do Dice e Jaccard, é o da imagem exibida na Figura 17.a. Operando sobre esta imagem, o modelo obteve um coeficiente de Dice de 43,5% e Jaccard de 27,8%, a segunda pior precisão de 29,41%, a sexta pior revocação de 83,57%, e acurácia de 96,78%. Dentre os possíveis obstáculos na imagem podem-se destacar: o pólipo pequeno, fora do foco, e posicionado em uma dobra, bem como a presença do que parece ser muco, que foi erroneamente classificado como um pólipo ocasionando a baixa precisão.

Figura 17 – Do segundo pior caso, a imagem e máscara originais, e a segmentação prevista pelo modelo (a, b, e c, respectivamente), e uma representação de erro (d).



(*a*) (*b*)

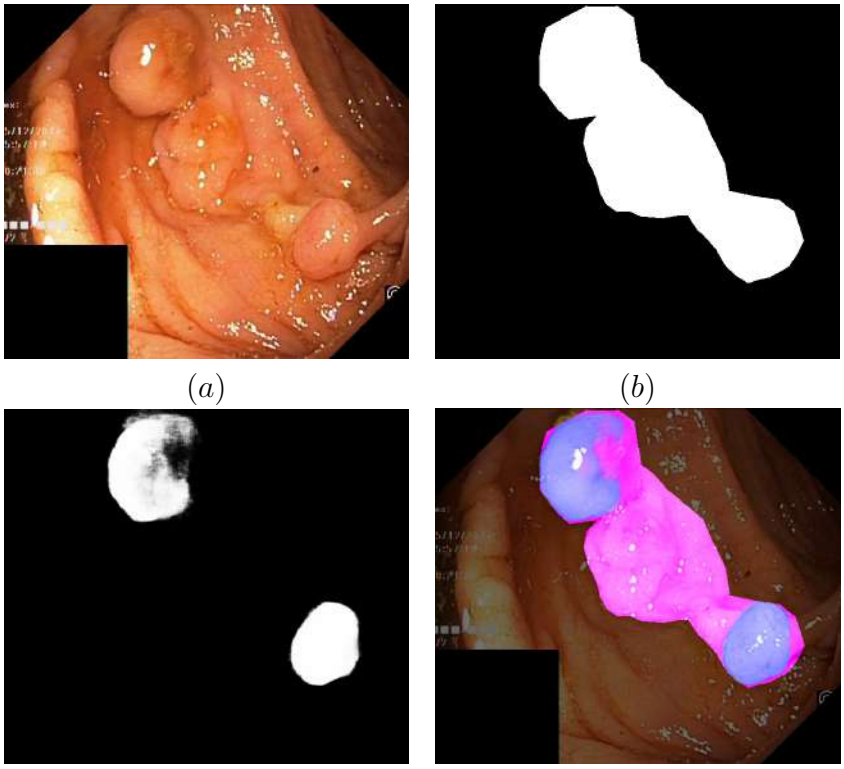
(*c*) (*d*)

Fonte: (JHA et al., 2020) e acervo do autor.

O terceiro pior caso, conforme os critérios previamente citados, é o da imagem na Figura 18.a, com Dice de 52,24%, Jaccard de 35,35% e revocação de 35,43% (a pior da base), e acurácia de 86,21%, a segunda mais baixa. Apesar disso, o caso possui uma precisão de 99,34%, 15º melhor nos testes. A mais notável qualidade possivelmente detrimental à segmentação é a forma incomum do alvo, composto por 3 pólipos aglomerados em sequência. Isto pode ser notado na discrepância entre a segmentação das diferentes partes do objeto:

o pólipo inferior, com aparência mais convencional, foi marcado com refino próximo da média; o superior, fora de foco e com coloração irregular, foi demarcado de forma mais grosseira; e o pólipo do meio foi despercebido por completo, possivelmente devido a seu relevo mais próximo da parede intestinal ou seu formato e coloração irregulares. A alta precisão deve-se ao tamanho da máscara-verdade e a falta de possíveis distrações ao seu redor, quase toda a segmentação do modelo está contida na segmentação dos especialistas.

Figura 18 – Do terceiro pior caso, a imagem e máscara originais, e a segmentação prevista pelo modelo (a, b, e c, respectivamente), e uma representação de erro (d).



(*a*) (*b*)

(*c*) (*d*)

Fonte: (JHA et al., 2020) e acervo do autor.

A Tabela 3, mostra o desempenho do modelo proposto sem levar em consideração os três casos analisados anteriormente. Nela, observa-se ganhos relevantes, cerca de 1%, em todas as métricas, porém, mais significativamente, diminui o desvio padrão drasticamente. Isto indica que estas 3 amostras são fortemente responsáveis pelos altos desvios encontrados, indicando a necessidade de uma investigação mais aprofundada de quais características ocasionam estas anomalias e, consequentemente, desenvolver modelos que consigam superar estes obstáculos.

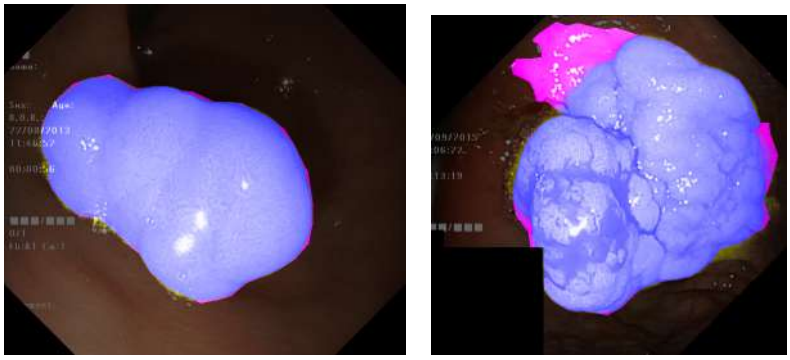
No melhor dos casos, o modelo atingiu um coeficiente de Dice de 98,9% e um Jaccard de 97,82%, além de uma precisão de 99,04%, uma revocação de 98,75%, e uma acurácia de 99,34%. A imagem é a que pode ser vista na Figura 19.a juntamente com a visualização das segmentações e sua intersecção. A imagem apresenta um pólipo grande

Tabela 3 – Resultados obtidos pela metodologia proposta sem os casos anômalos

Métrica	Média	Desvio Padrão
Coeficiente de Dice	94,94%	5,4197%
Coeficiente de Jaccard (IoU)	90,81%	8,7202%
Precisão (Precision)	95,81%	6,7588%
Revocação (Recall)	94,76%	7,2193%
Acurácia	98,30%	2,5360%

e com formato regular, em claro destaque na imagem, e poucas outras características distratoras no espaço restante.

Figura 19 – Representações de erro no melhor caso (a) e em um caso mediano (b).



(*a*) (*b*)

Fonte: (JHA et al., 2020) e acervo do autor.

Na Figura 19.b, pode-se ver um exemplo de um caso onde o modelo obteve desempenho próximo da média: o caso da imagem, com Dice de 93,29% e Jaccard de 87,42%, além de precisão de 98,02%, revocação de 88,99%, e acurácia de 94,09%. Na imagem, aparecem tanto características prejudiciais para o modelo corrente, como o formato e coloração irregulares do pólipo, quanto benéficas, como o tamanho e destaque do mesmo. Este equilíbrio seria um possível motivo do resultado mediano. Casos como estes reforçam ainda mais a necessidade de desenvolver um modelo que se adeque a essas características-problema.

4.3 Comparação com os Trabalhos Relacionados

A comparação do método desenvolvido com os resultados publicados em outros trabalhos na literatura pode ser vista na Tabela 4. O desempenho obtido pelo modelo proposto foi 0,83 pontos percentuais inferior ao obtido por Fitzgerald e Matuszewski (2023) no quesito Dice e 0,82 no quesito Jaccard. Ainda assim, os resultados obtidos foram promissores, pois estão próximos aos resultados produzidos pelos outros trabalhos.

Outro fato evidenciado pelos resultados produzidos por este trabalho é que os benefícios do *transfer learning* no ramo convolucional em adição ao ramo de ViT são

limitados. Porém, este déficit pode ser justificado por alguma diferença na implementação, visto que Fitzgerald e Matuszewski (2023) ainda não havia divulgado seu código-fonte durante o desenvolvimento desta pesquisa, levando à necessidade de implementar o modelo tomando como base o código-fonte fornecido por Sanderson e Matuszewski (2022), além da necessidade de migração de *framework*. Ademais, a *backbone* utilizada está entre as que possuem os desempenhos mais modestos no cenário atual, sendo focada em velocidade de processamento (KERAS, 2015). É possível que *transfer learning* realizado com modelos mais arrojados proporcione resultados maiores.

Tabela 4 – Comparação dos resultados da metodologia proposta com outros trabalhos relacionados usando a base Kvasir-SEG

Métrica	**Coeficiente de Dice**	**Índice de Jaccard**
Meta-Polyp	95,90%	92,10%
DUCK-Net	95,02%	90,51%
ESFPNet	93,10%	88,70%
UGCANet	92,80%	88,10%
PVT-GCASCADE	92,74%	87,90%
FCB Former	93,85%	89,03%
FCB Swin V2 Transformer	94,20%	89,73%
Nossa metodologia	93,37%	88,91%

Este capítulo descreveu e discutiu os resultados obtidos com a metodologia proposta. No capítulo seguinte, será feita a conclusão e algumas sugestões para trabalhos futuros.

5 Conclusão

Este trabalho teve como objetivo propor e testar uma metodologia usando rede neural para a segmentação semântica de pólipos em imagens de colonoscopia. Para alcançar esse objetivo, estruturou-se uma arquitetura de rede neural que usa um ramo de Vision Transformer e um ramo de FCN combinados para produzir a segmentação, onde ambos usam *backbones* pré-treinadas: um Swin Transformer V2 para o ramo ViT e uma Mobile Net V2 para o ramo FCN.

O modelo proposto alcançou os seguintes resultados: 93,37% de coeficiente de Dice, 88,91% de índice de Jaccard, 94,41% de precisão e 93,96% de revocação. Além disso, também foram comparados os resultados obtidos pela metodologia proposta com os trabalhos relacionados, onde mostraram-se promissores.

Por outro lado, observou-se que, apesar de alcançar bons resultados, em algumas imagens contendo pólipos muito pequenos, pólipos fora do foco, múltiplos pólipos, ou dobras intestinais em foco, a metodologia teve dificuldade de fazer a correta separação do pólipo do restante da imagem.

Outra limitação evidenciada é o fato de que, devido às restrições de tempo e de acesso a processamento de alta performance, não foi possível conduzir experimentos de ablação, que seriam úteis principalmente para averiguar o impacto da CLAHE e da nova *loss*, e nem propor e testar arquiteturas com outros *backbones* ou outras combinações de parâmetros para a CLAHE. Também não foi feita uma avaliação da metodologia proposta em outras bases de imagens ou a avaliação de generalização entre bases.

Como sugestões de trabalhos futuro pretende-se:

- explorar o uso de outros *backbones* no ramo FCN, como, por exemplo, a ResNet152 V2, que possui maior desempenho na base ImageNet, contudo são mais pesadas;
- verificar o impacto da transferência de conhecimento em outras bases e na generalização do modelo;
- testar a metodologia proposta para a segmentação de outros achados em imagens biomédicas;
- estudar o impacto do uso do logaritmo do índice de Jaccard em vez do negativo do coeficiente de Dice em funções de *loss*;
- dedicar mais atenção à melhora do ramo de FCN, visto que existem trabalhos na literatura atual focados no aprimoramento de arquiteturas de tipo U-Net, como a

DUCK-Net, que propõe um bloco convolucional mais sofisticado, ou a Focus U-Net, que desenvolve uma conexão codificador-decodificador mais arrojada; e,

- propor melhorias à estrutura geral das arquiteturas de fusão FCN-Transformer, por exemplo, poderia-se verificar a viabilidade de maior integração entre os ramos.

Referências

CHANG, Q.; AHMAD, D.; TOTH, J.; BASCOM, R.; HIGGINS, W. E. Esfpnet: efficient deep learning architecture for real-time lesion segmentation in autofluorescence bronchoscopic video. In: SPIE. *Medical Imaging 2023: Biomedical Applications in Molecular, Structural, and Functional Imaging*. [S.l.], 2023. v. 12468, p. 1246803. Citado na página 14.

CHOLLET, F. et al. *Keras*. 2015. <https://keras.io>. Citado na página 48.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A Large-Scale Hierarchical Image Database. In: *CVPR09*. [S.l.: s.n.], 2009. Citado na página 25.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. Citado na página 29.

DICE, L. R. Measures of the amount of ecologic association between species. *Ecology*, JSTOR, v. 26, n. 3, p. 297–302, 1945. Citado na página 37.

DONG, K.; ZHOU, C.; RUAN, Y.; LI, Y. Mobilenetv2 model for image classification. In: IEEE. *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*. [S.l.], 2020. p. 476–480. Citado 3 vezes nas páginas 7, 43 e 44.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S. et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. Citado na página 29.

DUMITRU, R.-G.; PETELEAZA, D.; CRACIUN, C. Using duck-net for polyp image segmentation. *Scientific Reports*, Nature Publishing Group UK London, v. 13, n. 1, p. 9803, 2023. Citado na página 13.

FITZGERALD, K.; MATUSZEWSKI, B. Fcb-swinv2 transformer for polyp segmentation. *arXiv preprint arXiv:2302.01027*, 2023. Citado 17 vezes nas páginas 7, 15, 30, 31, 32, 33, 34, 35, 36, 40, 43, 45, 46, 47, 48, 52 e 53.

GONZALEZ, R. C.; WOODS, R. E. *Processamento de Imagens Digitais*. [S.l.]: Editora Blucher, 2000. Citado 3 vezes nas páginas 17, 18 e 19.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning*. [S.l.]: MIT press, 2016. <http://www.deeplearningbook.org>. Citado 8 vezes nas páginas 13, 19, 20, 21, 22, 23, 24 e 25.

HUANG, S.; LU, Z.; CHENG, R.; HE, C. Fapn: Feature-aligned pyramid network for dense image prediction. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2021. p. 864–873. Citado na página 14.

HUNG, P. V.; MANH, N. D.; OANH, N. T.; THUY, N. T.; SANG, D. V. Ugcanet: A unified global context-aware transformer-based network with feature alignment for endoscopic image analysis. *arXiv preprint arXiv:2307.06260*, 2023. Citado na página 14.

INCA. *Câncer de intestino*. 2022. Disponível em: <https://www.gov.br/inca/pt-br/assuntos/cancer/tipos/intestino>. Acesso em: 12-12-2023. Citado 2 vezes nas páginas 12 e 16.

INCA. *Estatísticas de câncer*. 2022. Disponível em: <https://www.gov.br/inca/pt-br/assuntos/cancer/numeros>. Acesso em: 12-12-2023. Citado na página 12.

INCA. *Maranhão - estimativa dos casos novos*. 2022. Disponível em: <https://www.gov.br/inca/pt-br/assuntos/cancer/numeros/estimativa/estado-capital/maranhao>. Acesso em: 12-12-2023. Citado na página 12.

INCA. *O que é câncer?* 2022. Disponível em: <https://www.gov.br/inca/pt-br/assuntos/cancer/o-que-e-cancer>. Acesso em: 12-12-2023. Citado na página 16.

JACCARD, P. The distribution of the flora in the alpine zone. 1. *New phytologist*, Wiley Online Library, v. 11, n. 2, p. 37–50, 1912. Citado na página 37.

JHA, D.; SMEDSRUD, P. H.; RIEGLER, M. A.; HALVORSEN, P.; LANGE, T. de; JOHANSEN, D.; JOHANSEN, H. D. Kvasir-seg: A segmented polyp dataset. In: SPRINGER. *MultiMedia Modeling: 26th International Conference, MMM 2020, Daejeon, South Korea, January 5–8, 2020, Proceedings, Part II 26*. [S.l.], 2020. p. 451–462. Citado 8 vezes nas páginas 39, 40, 41, 42, 49, 50, 51 e 52.

KERAS. *Keras Applications*. 2015. Disponível em: <https://keras.io/api/applications>. Acesso em: 12-12-2023. Citado 2 vezes nas páginas 43 e 53.

LECUN, Y.; BOSER, B.; DENKER, J.; HENDERSON, D.; HOWARD, R.; HUBBARD, W.; JACKEL, L. Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, v. 2, 1989. Citado na página 20.

LEONDGARSE. *Keras CV Attention Models*. [S.l.]: GitHub, 2022. <https://github.com/leondgarse/keras_cv_attention_models>. Citado na página 48.

LIU, Z.; HU, H.; LIN, Y.; YAO, Z.; XIE, Z.; WEI, Y.; NING, J.; CAO, Y.; ZHANG, Z.; DONG, L. et al. Swin transformer v2: Scaling up capacity and resolution. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2022. p. 12009–12019. Citado 5 vezes nas páginas 15, 29, 30, 31 e 40.

LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 3431–3440. Citado na página 25.

LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. Citado na página 23.

OPENCV. *Histograms - 2: Histogram Equalization*. 2017. Disponível em: <https://docs.opencv.org/4.x/d5/daf/tutorial_py_histogram_equalization.html>. Acesso em: 12-12-2023. Citado na página 18.

PIZER, S. M.; AMBURN, E. P.; AUSTIN, J. D.; CROMARTIE, R.; GESELOWITZ, A.; GREER, T.; ROMENY, B. ter H.; ZIMMERMAN, J. B.; ZUIDERVELD, K. Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, Elsevier, v. 39, n. 3, p. 355–368, 1987. Citado na página 18.

RAHMAN, M. M.; MARCULESCU, R. G-cascade: Efficient cascaded graph convolutional decoding for 2d medical image segmentation. *arXiv preprint arXiv:2310.16175*, 2023. Citado na página 14.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. [S.l.], 2015. p. 234–241. Citado 6 vezes nas páginas 7, 12, 13, 25, 26 e 34.

ROY, A. G.; NAVAB, N.; WACHINGER, C. Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks. In: SPRINGER. *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part I*. [S.l.], 2018. p. 421–429. Citado 2 vezes nas páginas 14 e 33.

SANDERSON, E.; MATUSZEWSKI, B. J. Fcn-transformer feature fusion for polyp segmentation. In: SPRINGER. *Annual Conference on Medical Image Understanding and Analysis*. [S.l.], 2022. p. 892–907. Citado 15 vezes nas páginas 7, 14, 29, 30, 31, 32, 34, 35, 36, 43, 44, 45, 47, 48 e 53.

SORENSEN, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons. *Biologiske skrifter*, v. 5, p. 1–34, 1948. Citado na página 37.

TRINH, Q.-H. Meta-polyp: a baseline for efficient polyp segmentation. *arXiv preprint arXiv:2305.07848*, 2023. Citado na página 13.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017. Citado 6 vezes nas páginas 7, 20, 26, 27, 28 e 29.

YU, W.; LUO, M.; ZHOU, P.; SI, C.; ZHOU, Y.; WANG, X.; FENG, J.; YAN, S. Metaformer is actually what you need for vision. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2022. p. 10819–10829. Citado na página 13.

YU, W.; SI, C.; ZHOU, P.; LUO, M.; ZHOU, Y.; FENG, J.; YAN, S.; WANG, X. Metaformer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, 2023. Citado na página 13.

YUE, K.; SUN, M.; YUAN, Y.; ZHOU, F.; DING, E.; XU, F. Compact generalized non-local network. *Advances in neural information processing systems*, v. 31, 2018. Citado na página 14.