

**UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO**



Gustavo Bodart Guimarães Caetano

**REDES NEURAIS CONVOLUCIONAIS APLICADAS À
SEGMENTAÇÃO DE IMAGENS DE LESÕES DE PELE**

Vitória-ES

Dezembro/2018

Gustavo Bodart Guimarães Caetano

REDES NEURAIS CONVOLUCIONAIS APLICADAS À SEGMENTAÇÃO DE IMAGENS DE LESÕES DE PELE

Parte manuscrita do Projeto de Graduação do aluno Gustavo Bodart Guimarães Caetano, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Vitória-ES

Dezembro/2018

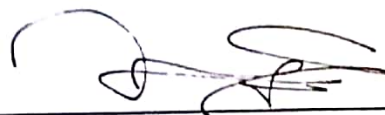
Gustavo Bodart Guimarães Caetano

REDES NEURAIS CONVOLUCIONAIS APLICADAS À SEGMENTAÇÃO DE IMAGENS DE LESÕES DE PELE

Parte manuscrita do Projeto de Graduação do aluno Gustavo Bodart Guimarães Caetano, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

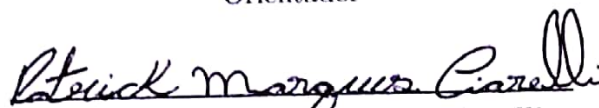
Aprovado em 5 de Dezembro de 2018.

COMISSÃO EXAMINADORA:

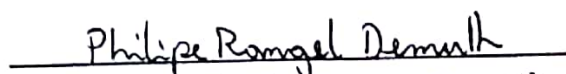


Prof. Dr. Jorge Leonid Aching
Samatelo

Universidade Federal do Espírito Santo
Orientador



Prof. Dr. Patrick Marques Ciarelli
Universidade Federal do Espírito Santo
Examinador



Prof. Msc. Philipe Rangel Demuth
Universidade Federal do Espírito Santo
Examinador



Msc. Clebeson Canuto dos Santos
Universidade Federal do Espírito Santo
Co-orientador

Vitória-ES

Dezembro/2018

AGRADECIMENTOS

Aos meus pais, pelo apoio e amor a mim concedidos durante toda a graduação, muitas vezes deixando de realizar seus sonhos para que eu realizasse o meu.

A minha irmã que me serviu de guia na carreira de engenharia e como modelo de profissional.

A Flávio Machado e Clebeson Canuto dos Santos pela ajuda indispensável em todas as etapas práticas do projeto.

Ao meu orientador por toda ajuda durante a realização do trabalho, por ser um dos melhores professores de toda minha graduação e por despertar interesse sobre esse tema ainda na sala de aula.

À banca examinadora pela aceitação do convite e pelo tempo investido para leitura e avaliação desse trabalho.

RESUMO

No Brasil, o câncer de pele é o tipo de câncer que apresenta maior número de ocorrências, sendo melanoma o subtipo com maior taxa de fatalidade. Com um cenário de morbidade tão acentuado, mostra-se necessário um diagnóstico avançado para que a doença seja controlada ainda em estágios iniciais, para que as chances de sobrevivência sejam elevadas. A dificuldade do diagnóstico de melanomas se dá em boa parte pela semelhança com pintas e sinais que se apresentam na pele. Desta maneira, é de suma importância o desenvolvimento de ferramentas de diagnóstico de alto nível, uma vez que as soluções existentes não apresentam resultados superiores aos métodos tradicionais. No presente projeto de graduação é proposta uma técnica baseada em duas etapas para efetuar a segmentação de imagens de lesões de pele, especificamente, a primeira etapa efetua uma redimensionalização e preparo das imagens dermatoscópicas e a segunda etapa efetua a tarefa de segmentação através de redes neurais convolucionais. Tal tarefa é modelada como um problema de segmentação semântica, de modo que, é usada a rede SegNet para efetuar a segmentação. A base de dados usada para realizar o treinamento e validação da rede foi obtida do desafio ISIC 2018. Na etapa de avaliação foi obtido um índice de Jaccard de 62,87%, sendo estimado então que o trabalho ficaria situado em centésimo oitavo lugar no desafio.

Palavras-chave: *Deep-learning*; Redes neurais convolucionais; *Transfer-learning*; *Semantic-Segmentation*; Melanoma; Cancer de Pele; diagnóstico automático

ABSTRACT

In Brazil, skin cancer is the type of cancer that presents the highest number of occurrences, being melanoma the subtype with the highest fatality rate. With such a morbid scenario, it is necessary to have a diagnosis in advance so that the disease is controlled even in the early stages, so that the chances of survival are higher. The difficulty of diagnosing melanomas is due mostly to the similarity with pimples and signs that appear on the skin. In this way, the development of high level diagnostic tools is of paramount importance, since existing solutions do not present superior results to traditional methods. The present graduation project makes use of a convolutional network to try to solve the problem of automatic segmentation of images of skin lesions, which is characterized as a challenge of semantic segmentation and predicts the classification of regions of the image in lesion and background. The task to be solved was obtained from the ISIC challenge 2018, which provides a database for network training and validation of solutions. In order to find a better result, techniques of image preprocessing were used. In this work an in-depth study on the SegNet architecture detailing its internal operation is made. Regarding the results, a Jaccard index of 62,87% was obtained using the training database of the ISIC 2018 challenge, being placed in hundred eighth place of the challenge.

Keywords: *Deep-learning; convolutional neural networks; Transfer-learning; Semantic-Segmentation*

LISTA DE FIGURAS

Figura 1 – Segmentação de lesão de pele	14
Figura 2 – Adversidades no banco dados	15
Figura 3 – Representação gráfica da camada convolucional	20
Figura 4 – Representação gráfica da função ReLu	21
Figura 5 – Exemplo de <i>Unpooling</i> . Elemento mais escuro equivale ao maior valor .	22
Figura 6 – Representação gráfica da Rede Neural VGGNet-16	24
Figura 7 – Exmplo de Segmentação Semântica	27
Figura 8 – Exemplos de imagens submetidas a SegNet em segmentação semântica	28
Figura 9 – Arquitetura da SegNet	29
Figura 10 – Adversidades no banco dados	33
Figura 11 – Representação do tensor \mathbf{T} em <i>one hot encoding</i> relacionado a uma máscara de segmentação \mathbf{B} de 5 classes.	35
Figura 12 – Explicação de one hot encoding	36
Figura 13 – Adversidades no banco dados	38
Figura 15 – Fluxograma dos passos de treinamento	40
Figura 17 – Adversidades no banco dados	49

LISTA DE TABELAS

Tabela 1	– Estrutura da rede VGG-16.	25
Tabela 2	– Estrutura da rede VGG-19.	26
Tabela 3	– Estrutura da rede SegNet.	30
Tabela 4	– Valores das métricas de desempenho na segmentação das imagens de lesão de pele usando a função de custo entropia cruzada $L_{Entropy}$	45
Tabela 5	– Valores das métricas de desempenho na segmentação das imagens de lesão de pele usando a função de custo índice de Jaccard aproximado L_{Dice}	45
Tabela 6	– <i>Ranking</i> para o problema de segmentação de lesão do desafio ISIC 2018.	47

LISTA DE ABREVIATURAS E SIGLAS

CNN	<i>Convolutional Neural Network</i>
DL	<i>Deep Learning</i>
GPU	<i>Graphics Processing Unit</i>
ISIC	<i>International Skin Imaging Collaboration</i>
ML	<i>Machine Learning</i>
SGD	<i>Stochastic Gradient Descent</i>
SOL	<i>Structured Output Learning</i>
UFES	Universidade Federal do Espírito Santo

SUMÁRIO

1	INTRODUÇÃO	13
1.1	Apresentação e Objeto de Pesquisa	13
1.2	Trabalhos Relacionados	16
1.3	Objetivos	17
1.4	Estrutura do Texto	18
2	EMBASAMENTO TEÓRICO	19
2.1	Introdução	19
2.2	Redes Neurais Convolucionais	19
2.2.1	Camada convolucional	19
2.2.2	Camada de <i>Pooling</i>	21
2.2.3	Camada de <i>Unpooling</i>	22
2.2.4	Camada totalmente conectada	22
2.2.5	Treinamento de uma CNN	23
2.3	VGGNet	23
2.4	<i>Transfer Learning</i>	25
2.5	Segmentação semântica	26
2.6	SegNet	28
2.7	Resumo	30
3	SOLUÇÃO PROPOSTA	32
3.1	Introdução	32
3.2	Etapa de Preparo dos Dados	32
3.2.1	Etapa de Segmentação	33
3.3	Resumo	35
4	RESULTADOS	37
4.1	Introdução	37
4.2	Alocação de Recursos	37
4.2.1	Base de dados ISIC 2018	37
4.2.2	Recursos Computacionais	39
4.3	Detalhes de implementação	40
4.3.1	Sobre o tratamento dos dados do conjunto de treinamento e teste	41
4.3.2	Sobre a implementação da rede SegNet	41
4.3.3	Sobre a etapa de treinamento da rede SegNet	42
4.3.4	Sobre a etapa de predição da rede SegNet	43

4.4	Parte Experimental	43
4.4.1	Métricas	43
4.4.2	Experimento Efetuados	45
4.4.2.1	Comparação de resultados	46
4.5	Resumo	48
5	CONCLUSÕES E PROJETOS FUTUROS	50
5.1	Conclusões	50
5.2	Temas a serem pesquisados	50
	REFERÊNCIAS	51

1 INTRODUÇÃO

1.1 Apresentação e Objeto de Pesquisa

É estimado que 171.840 novos casos de câncer de pele surjam no Brasil em 2018, configurando assim o tipo de câncer que mais afeta os brasileiros (INCA, 2017). O câncer de pele é dividido em duas classes principais, chamadas de não-melanoma e melanoma, sendo este último o de maior letalidade e foco deste trabalho. O melanoma é estimado em 6.260 novos casos no Brasil em 2018. Apesar de sua proporção ser pequena em relação aos casos de câncer não melanoma, em 2015 foram relatadas 1.794 mortes por melanoma (Ministério da saúde, 2017).

O diagnóstico é realizado inicialmente por inspeção visual, podendo ser aprimorado de maneira não invasiva por análise dermatoscópica, técnica que amplia o tamanho da lesão e profundidade de captura da imagem. Essa técnica é utilizada então para facilitar a inspeção visual de características como formato, cor e irregularidade da lesão. Apesar destes métodos serem utilizados, o diagnóstico tende a ser subjetivo e impreciso mesmo entre dermatologistas experientes. Setenta por cento dos diagnósticos não são feitos por médicos cancerologistas, o que reduz ainda mais a assertividade do diagnóstico. Assim, o diagnóstico definitivo da malignidade é extraído de análise histopatológica, ou seja, biópsia do tecido retirado do local de lesão. Em geral, o melanoma se apresenta em estágios iniciais como “pintas” ou “sinais”, o que pode confundir o especialista em primeira análise. O diagnóstico por inspeção visual sem auxílio de dermatoscopia apresenta uma acurácia de aproximadamente 60% (KITTLER et al., 2017). Com o uso de dermatoscopia, a acurácia aumenta para 75%-84% quando utilizada por profissionais experientes (KITTLER et al., 2017). Quando utilizada por profissionais não treinados ou menos experientes, os resultados não são melhores dos que os obtidos sem o uso da técnica (KITTLER et al., 2017).

Como em qualquer problema de saúde, o diagnóstico precoce aumenta as chances de cura ou sobrevida. Assim, se fazem necessárias ferramentas que amplifiquem a capacidade humana de diagnóstico. Nesse contexto, o uso de Métodos Automatizados de Diagnóstico - MAD podem aumentar a assertividade e rapidez do diagnóstico.

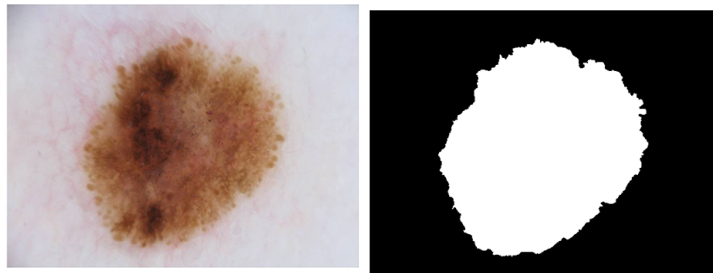
Comumente, os MAD seguem os seguintes passos de processamento: (i) segmentação da região de pele relacionada à lesão em si; (ii) extração de características da mancha da lesão; (iii) classificação da lesão em função das características extraídas. Destes três passos a segmentação da imagem dermatoscópica é fundamental, já que as posteriores etapas dependem do resultado da segmentação. Em geral, entende-se por segmentação de uma imagem a tarefa de subdividir a imagem nas suas regiões constituintes. O modo em que

as subdivisões são feitas depende do problema a ser resolvido (GONZALEZ; WOODS, 2012). Para o caso da segmentação em duas classes (binária), uma imagem binária (que classifica cada pixel da imagem em uma de duas classes) é gerada a partir da imagem original. Para o caso em estudo, será entendido como objeto binário todo conjunto de pixels na imagem binária que corresponde a uma lesão de pele na imagem dermatoscópica original. Em resumo, a segmentação de uma imagem dermatoscópica pode ser formulada como segue: dada uma imagem dermatoscópica de entrada $\mathbf{I} \in [0, 255]^{N \times M \times 3}$, a técnica de segmentação gera como saída uma imagem binária $\mathbf{B} \in [0, 1]^{N \times M}$, onde, para cada pixel $\mathbf{I}(i, j) \in [0, 255]^{3 \times 1}$ seu correspondente pixel $\mathbf{B}(i, j) = 1$ se a posição (i, j) na imagem \mathbf{I} está localizado sobre uma região de lesão de pele. Na Figura 1 pode-se ver um exemplo da técnica de segmentação, onde 1a representa \mathbf{I} e 1b representa \mathbf{B} .

Figura 1 – Segmentação de uma imagem dermatoscópica.

(a) Imagem de Entrada \mathbf{I}

(b) Imagem de saída \mathbf{B}



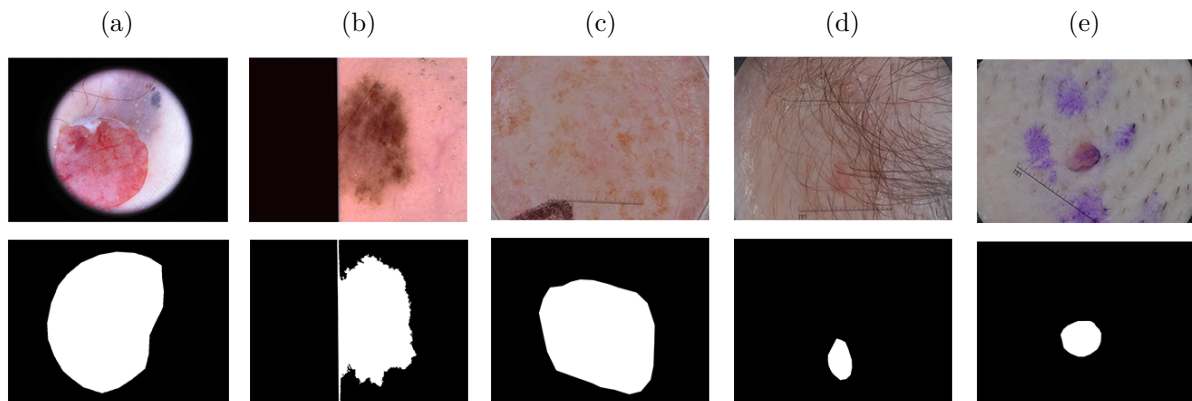
Fonte: Banco de Dados ISIC *Archive*

Entre os desafios que deve enfrentar uma técnica de segmentação de imagens dermatoscópicas estão a capacidade de poder processar as imagens de entrada considerando diferentes adversidades que podem estar presentes. Como exemplos, podem ser citadas grande quantidade de pelos (Figura 13d), presença de margem escura (Figura 13a), manchas secundárias (Figura 13c) e marcas de tinta (Figura 13e). Tais problemas restringem o uso de técnicas clássicas de segmentação, como as baseadas em limiares (LEE; CHUNG; PARK, 1990), *clustering* de dados (WU; LEAHY, 1993), entre outras técnicas.

Atualmente, uma área de pesquisa de *Machine Learning* (ML) (tradução livre, aprendizado de máquinas) denominada *Deep Learning* (DL) (tradução livre, aprendizado profundo) permite implementar técnicas de segmentação sofisticadas aplicadas a diferentes contextos, como: detecção de objetos (GIRSHICK et al., 2014), segmentação binária de documentos (OLIVEIRA; SEGUIN; KAPLAN, 2018), tradução automática de placas e letreiros (GOOD, 2015), identificação de imagens pulmonares de raios x (KALINOVSKY; KOVALEV, 2016) entre outros.

DL surge como uma bio-inspiração das estruturas neurais do córtex visual dos mamíferos

Figura 2 – Entradas e saídas com condições adversas. Em cima imagem original, abaixo a referência: (a) margem circular; (b) margem irregular; (c) manchas secundárias; (d) grande quantidade de pelos; (e) manchas de tinta.



Fonte: Banco de Dados ISIC *Archive*

(HUBEL; WIESEL, 1968). Nesse sentido, em DL são usadas estruturas em cascada de camadas de processamento não lineares. As camadas seguem um modelo hierárquico, tal que uma camada mais baixa é resultado de transformações lineares e não lineares da camada anterior (LECUN; BENGIO; HINTON, 2015). Especificamente, cada camada utiliza os dados de saída da camada anterior como dados de entrada para si. Os algoritmos de DL têm a capacidade de aprender o valor de seus parâmetros de maneira supervisionada (classificação, por exemplo) e não supervisionada (análise de padrões, por exemplo). O aprendizado é feito em níveis de representação, o que corresponde a diferentes níveis de abstração, os quais formam uma hierarquia de conceitos.

As arquiteturas de DL que mais se aproximam do modelo biológico são as *Convolutional Neural Networks* (DCNN) (tradução, redes neurais convolucionais) que foram propostas por LeCun et al. (1998a). No referido trabalho foi proposta a rede CNN denominada *LeNet-5*, aplicando-a ao problema de reconhecimento de dígitos manuscritos, sendo treinada e testada com a base de dados MNIST (*Modified National Institute of Standards and Technology database*, base de dados modificada do Instituto Nacional de Padrões e Tecnologia, tradução livre), obtendo um resultado de 99,05% de acurácia na classificação.

Neste trabalho propõe-se o uso de uma rede neural DCNN como elemento principal para a técnica de segmentação de imagens dermatoscópicas, reaproveitando-se arquiteturas CNN já conceituadas na literatura através do uso de paradigma de aprendizagem denominado *Transfer Learning* - TL (tradução livre, aprendizado por transferência). TL tem um forte impacto na área médica, sendo aplicado a diferentes problemas como na classificação automática de núcleos de células em imagens histopatológicas (BAYRAMOGLU; HEIKKILÄ, 2016) e na diferenciação de meduloblastomas (CRUZ-ROA et al., 2015).

1.2 Trabalhos Relacionados

Embora os primeiros trabalhos de segmentação de imagens de lesões de pele tenham usado técnicas de limiarização (LEE; CHUNG; PARK, 1990) e técnicas de *clusterization* (WU; LEAHY, 1993), estas não serão mencionadas por ser amplamente superadas pelas técnicas baseadas em DL. A seguir são comentados os trabalhos de interesse presentes na literatura relacionados ao problema de segmentação automatizada de imagens de câncer de pele.

Em ***A state-of-the-art survey on lesion border detection in dermoscopy images*** (CELEBI et al., 2015) os autores descrevem 50 métodos automáticos de segmentação de melanomas e avaliam os possíveis fatores que dificultem o reconhecimento da imagem ou o processo computacional (CELEBI et al., 2015). A análise é detalhada e leva em consideração vários processos como pré-processamento, eliminação de objetos e lesões secundárias, pós-processamento, diferentes técnicas de segmentação e número de parâmetros de treinamento.

Em ***Dermatologist-level classification of skin cancer with deep neural networks*** (ESTEVA et al., 2017) os autores realizam um trabalho semelhante ao proposto no desafio ISIC, que será descrito neste trabalho, porém de maneira mais abrangente para vários tipos de doença de pele. Os autores também mostram a necessidade de um grande banco de dados para resultados satisfatórios. No trabalho em questão também é utilizado TL para otimização do processo de treinamento.

Em ***Automatic skin lesion segmentation with fully convolutional-deconvolutional networks*** (YUAN; CHAO; LO, 2017) os autores descrevem suas soluções para o desafio ISIC 2017. Essa CNN ganhou o primeiro lugar do concurso na atividade de segmentação de lesões. Os autores utilizam uma arquitetura própria de 29 camadas utilizando ReLu como função de ativação de cada camada, seguida de um *ensemble* de outras 6 redes. Foi realizado um pré-processamento de normalização de cores e redimensionalização para 192×256 pixels. Também foi utilizado um processo de *data augmentation* ao aplicar rotações, espelhamentos, translações e recortes. Os autores criaram um função de custo própria baseada no índice de Jaccard. Por fim, foram utilizadas algumas estratégias de pós-processamento como calcular o maior objeto entre os segmentados e preenchimento de pequenos buracos com dilatação morfológica.

Em ***RECOD titans at ISIC challenge 2017*** (MENEGOLA et al., 2017) os autores descrevem suas soluções através de TL para o desafio ISIC 2017. As soluções ganharam o quinto e primeiro lugar, respectivamente, nas atividades de segmentação e classificação de

lesões. Os autores utilizam uma arquitetura disposta publicamente no *GitHub* baseada em uma rede U-Net. Foi realizado um pré-processamento de redimensionamento das imagens para 128×128 . Também foi utilizado um processo de *data augmentation* ao aplicar rotações, translação e recorte. Os autores utilizaram o Coeficiente de Sorensen-Dice como função de custo. Não foram utilizadas técnicas de pós-processamento.

Os trabalhos referenciados mostram a dificuldade em segmentar lesões vindas de melanoma uma vez que as imagens de treinamento podem estar poluídas com muitas interferências, como pelos e manchas secundárias, (CELEBI et al., 2015) e o banco de treinamento pode não ser grande o suficiente para que a rede neural aprenda a generalizar o suficiente para tratar imagens dermatoscópicas diversas (ESTEVA et al., 2017). Neste caso, uma vez que a segmentação da lesão principal é dificultada, posteriores processos automáticos de diagnósticos se tornam inviáveis. Este trabalho tem, então, o objetivo de tentar solucionar um problema de segmentação de lesões de pele com uma abordagem utilizando redes neurais convolucionais.

1.3 Objetivos

Objetivo Geral

- Desenvolver uma técnica de segmentação de lesões de pele baseada na rede CNN, que permita usar arquiteturas CNN já consolidadas da literatura e aplicá-las ao problema em estudo.

Objetivos Específicos

- Realizar uma revisão bibliográfica de trabalhos que estudam o problema de segmentação de imagens usando a rede CNN e TL;
- Implementar uma técnica de segmentação automática de lesões em imagens dermatoscópicas baseada em uma rede CNN;
- Validar e testar a técnica implementada.

1.4 Estrutura do Texto

O presente trabalho está estruturado da seguinte maneira:

1. **Introdução:** este capítulo tem como objetivo contextualizar o trabalho e o problema aqui estudado, apresentando o problema e ideias iniciais sobre a solução do mesmo. Além disso, são apresentados os objetivos da realização deste projeto de graduação;
2. **Referencial Teórico:** aqui são apresentados os assuntos tratados neste trabalho, tais como: CNN, TL, VGGNet, segmentação semântica e SegNet.
3. **Solução Proposta:** neste capítulo é apresentada a técnica proposta para solucionar o problema de estudo;
4. **Resultados:** neste capítulo são apresentados os resultados dos experimentos, assim como uma comparação com os trabalhos da literatura;
5. **Conclusão:** No capítulo final deste trabalho os resultados são resumidos e discutidos brevemente, além de propor melhorias para futuros estudos.

2 EMBASAMENTO TEÓRICO

2.1 Introdução

Este capítulo tem por finalidade estabelecer os conceitos teóricos necessários usados no trabalho. O capítulo se inicia descrevendo uma rede CNN, técnica básica para toda a produção do trabalho. Em seguida é feita uma apresentação da rede neural VGG e do paradigma de aprendizagem *transfer learning*. Por último é descrito o problema de segmentação semântica e a rede neural SegNet especializada neste tipo de problema.

2.2 Redes Neurais Convolucionais

A arquitetura de uma CNN possui uma entrada, uma saída e, entre elas, várias camadas ocultas de processamento. Estas camadas ocultas normalmente são camadas convolucionais, camadas de *pooling* e camadas totalmente conectadas. A seguir são explicadas cada uma destas camadas.

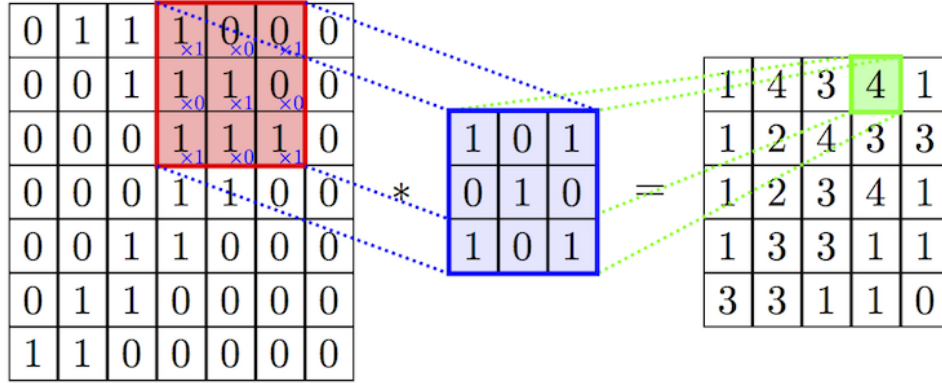
2.2.1 Camada convolucional

A camada convolucional tem como papel aprender as características de natureza espacial presentes na imagem de entrada. Essa camada extrai mapas de características através de filtros ou *kernels* convolucionais. Cada neurônio de um mapa de características é conectado a uma região de neurônios vizinhos na camada anterior. Essa vizinhança também pode ser chamada de “entrada” do neurônio em questão. Para o k -ésimo mapa, a característica relativa à posição (i, j) da l -ésima camada é calculada da seguinte maneira:

$$z_{i,j,k}^l = \mathbf{w}_k^l \mathbf{x}_{i,j}^l + b_k^l. \quad (2.1)$$

onde $\mathbf{x}_{i,j}^l$ é uma fração da entrada centrada na posição (i, j) referente à l -ésima camada; \mathbf{w}_k^l e b_k^l são o vetor de pesos e o termo de *bias* do k -ésimo filtro da l -ésima camada, respectivamente. Observe que o filtro \mathbf{w}_k^l que gera o mapa de característica $\mathbf{z}_{i,j,k}^l$ é compartilhado por todos os elementos do mapa de características e da camada em questão. Essa estratégia de compartilhar pesos tem muitas vantagens, como a redução da complexidade do modelo, tornando o treinamento da rede mais fácil. Uma explicação visual da camada de convolução pode ser observada na Figura 3

Figura 3 – Representação gráfica da camada convolucional



Fonte:

(GILLEMAN, a)

Para gerar completamente um mapa de características, normalmente, são utilizados dois estágios. Primeiramente é feita a convolução da entrada por um filtro já aprendido, e depois é aplicada a função de ativação aos resultados da convolução. Caso a função de ativação seja não linear, esta pode inserir não linearidades na CNN, que são desejáveis para o funcionamento da rede, uma vez que permitem a detecção de características não-lineares. Seja $f(\bullet)$ uma função de ativação não linear, então, o valor que retorna da função de ativação quando a entrada é a característica convolucional $z_{i,j,k}^l$ pode ser calculado como:

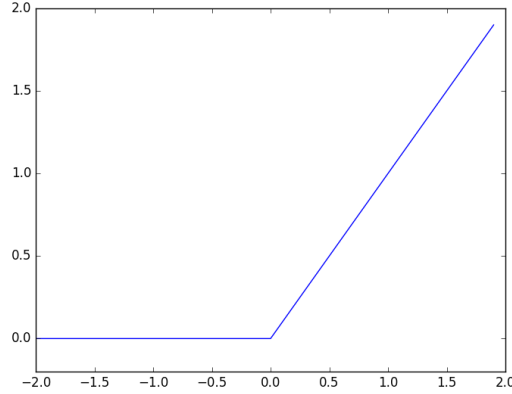
$$a_{i,j,k}^l = f(z_{i,j,k}^l). \quad (2.2)$$

Funções de ativação típicas são as funções sigmoide, tangente hiperbólico (LECUN et al., 1998b) e, mais recentemente, ReLU - *rectified linear unit* (tradução livre, unidade linear retificada) (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) (NAIR; HINTON, 2010). A ReLu pode ser definida pela Equação 2.3 e pode ser observada na Figura 4.

$$f(x) = \max(0, x) \quad (2.3)$$

A substituição das funções de ativação tradicionais para a ReLU foi de extrema importância pois diminuiu significativamente o tempo de treino e de inferência das redes neurais. Quando comparada á ativação por tangente hiperbólica, o tempo de treino foi até 6 vezes menor, dependendo da aplicação (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

Figura 4 – Representação gráfica da função ReLu



Fonte: Produção do próprio autor.

2.2.2 Camada de *Pooling*

Entre duas camadas de convolução pode ser inserida uma camada de *pooling*. Esse tipo de camada tem como objetivo alcançar uma invariância nos deslocamentos espaciais (JADERBERG et al., 2015), reduzindo a resolução espacial do mapa de características. Essa redução é atingida através de passos, ou *strides*, não unitários. Assim, as ativações apenas são calculadas em intervalos de posições de entrada de número igual ao tamanho do passo, caso as dimensões sejam de número par. De acordo com a rede de referência, que será apresentada, pode-se utilizar, por exemplo, um passo de tamanho 2, o que reduz as dimensões espaciais pela metade em cada eixo do plano. Cada mapa de características é conectada com o respectivo mapa da camada convolucional anterior. Para poder definir a saída gerada pela camada de *pooling*, pode-se tomar $pool(\bullet)$ como função matemática de *pooling*. Assim, a saída relativa ao mapa de características $\mathbf{a}_{:, :, k}^l$ é representada como:

$$y_{i,j,k}^l = pool(a_{m,n,k}^l), \forall (m,n) \in \mathcal{R}_{i,j}, \quad (2.4)$$

onde $\mathcal{R}_{i,j}$ é uma vizinhança local centrada na posição (i,j) . Duas operações comuns de *pooling* são *max pooling* e *average pooling*. A primeira retorna o maior valor dentro de seu campo receptivo, e a segunda retorna a média dos valores de seu campo receptivo. Cada uma tem seu benefício em relação ao tipo de rede e informação aplicada na entrada. Uma explicação visual do *max pooling* pode ser observada na figura 5.

2.2.3 Camada de *Unpooling*

Uma camada de *unpooling* permite gerar um mapa de características de maior resolução a partir de um mapa de características de menor resolução. Comumente esta operação é não inversível (cada elemento do mapa de características origem corresponde a diferentes elementos do mapa de características destino). Tal problema é contornado através de um mecanismo descrito em (NOH; HONG; HAN, 2015) e é demonstrado na Figura 5. Especificamente, este mecanismo consta: em uma dada operação de *max pooling*, de 64×64 para 32×32 , por exemplo, em uma dada vizinhança o maior elemento é o único elemento a ser passado para a função de ativação. Simultaneamente, a rede guarda a localização deste maior elemento dentro desta vizinhança, para todas as vizinhanças avaliadas pelo *max pooling*. Esse procedimento é repetido para todas as camadas de *max pooling*. Então, em uma camada de *unpooling*, a rede recupera a informação salva na camada de *max pooling* inversamente equivalente. Para o exemplo dado, a camada de *unpooling* de 32×32 para 64×64 , recupera as posições salvas da camada de *max pooling*, de 64×64 para 32×32 . Então o valor resultante da camada de *unpooling* em uma vizinhança é inserido exatamente na posição salva da camada de *max pooling*. Os outros elementos dessa vizinhança são completados com o valor 0.

Figura 5 – Exemplo de *Unpooling*. Elemento mais escuro equivale ao maior valor



Fonte: (NOH; HONG; HAN, 2015)

2.2.4 Camada totalmente conectada

A camada totalmente conectada leva esse nome por conectar todos os neurônios da camada anterior a cada um dos neurônios da camada atual. A rede neural em si segue uma lógica de que quanto mais profunda, mais características específicas podem ser mapeadas, o que pode, inclusive, levar a *overfitting*, que é o aprendizado tão profundo que a rede não consegue generalizar e apenas consegue reconhecer as entradas de treinamento. Assim, é observado

que filtros das primeiras camadas têm como função detectar características mais básicas, enquanto filtros de camadas mais profundas extraem características mais abstratas. As camadas totalmente conectadas são utilizadas então para interpretar essas características abstratas, simulando ações de raciocínio complexo, por exemplo, reconhecendo pessoas de uma foto, de acordo com as características faciais extraídas. Após todas essas camadas, existe uma camada final chamada de saída, sendo comum a prática de utilizar um neurônio para cada classe possível de saída em um algoritmo de classificação. No caso de segmentação, é utilizado um processo “reverso”, pois a saída desejada é uma representação binária de mesma dimensão da entrada.

2.2.5 Treinamento de uma CNN

Todo o conceito de inteligência artificial em CNN se baseia em parâmetros livres que entreguem um resultado esperado de acordo com uma entrada submetida. Para a obtenção desses parâmetros é comum que CNN utilizem de algoritmos de aprendizado. O algoritmo mais utilizado para essa aplicação é o de *backpropagation* (LECUN et al., 1989). Esse algoritmo calcula o gradiente de uma função objetivo para determinar como alterar os parâmetros livres da rede com a finalidade de minimizar os erros que afetam o desempenho na tarefa objetivo da rede.

2.3 VGGNet

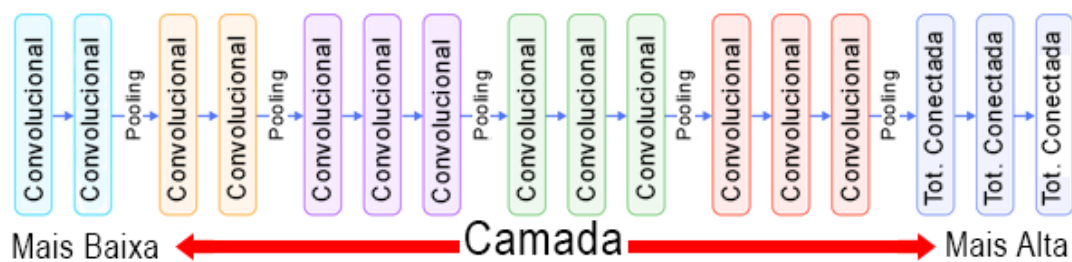
A rede neural VGG foi proposta em (SIMONYAN; ZISSERMAN, 2015). Possui duas arquiteturas diferentes, uma com 16 e outra com 19 camadas. Utiliza filtros 3×3 em todas as camadas convolucionais. Ambas arquiteturas são apresentadas nas Tabelas 1 e 2. Todas as tabelas foram montadas levando em consideração uma imagem de entrada de dimensões $224 \times 224 \times 3$. Essa rede foi campeã na categoria localização do *ImageNet* ILSVRC 2014. O ILSVRC (RUSSAKOVSKY et al., 2015) consiste em um desafio de processamento automático de imagens e tem como foco a localização de objetos, detecção de objetos e classificação de imagens. Este desafio se tornou referência em visão computacional e fornece um banco de dados de mais de quatorze milhões de imagens rotuladas, além de conjuntos adicionais menores de validação e teste. Nos últimos anos, as soluções baseadas em DL se mostraram superiores, obtendo melhores resultados em todas as classificações.

A VGG pode ser comparada com as arquiteturas vencedoras dos anos anteriores a ela, que foram:

- Em 2012, a *AlexNet* venceu o desafio de classificação utilizando 5 camadas convolucionais e 2 totalmente conectadas. Este foi o primeiro ano que uma CNN profunda foi utilizada.
- Em 2013 a arquitetura vencedora foi semelhante à *AlexNet*, porém, utilizando filtros de 7×7 em vez de 11×11 na primeira camada, e o passo foi alterado de 4 para 2.

Assim, é possível observar que a VGGNet tem uma arquitetura com mais do que o dobro de profundidade em relação as anteriores. Também é possível perceber que a redução da dimensão dos filtros foi realizada como no ano anterior, porém, a VGGNet possui filtros 3×3 . A VGGNet também tem valor didático quando se observa sua estrutura na Figura 6, pois tem uma composição simples e linear de camadas de convolução e de *pooling*.

Figura 6 – Representação gráfica da Rede Neural VGGNet-16



Fonte: (SLIDESHARE,) (traduzido)

Existe uma grande quantidade de trabalhos em que se usa VGGNet e a técnica de TL como nos citados a seguir:

- Em (AYTAR; VONDRICK; TORRALBA, 2016) os autores utilizam DL para associar sons ambiente à vídeos não rotulados. Para a parte do trabalho relacionada a aquisição de informações do vídeo, foi utilizada VGGNet e TL.
- Em (NG et al., 2015) os autores utilizam VGGNet para reconhecer emoções no rosto de uma pessoa presente em uma imagem.
- Em (CRUZ-ROA et al., 2015) os autores utilizam VGGNet para fazer a diferenciação de meduloblastoma em imagens histopatológicas.
- Em (GHAZI; YANIKOGLU; APTOULA, 2017) é utilizada VGGNet para identificar plantas em fotos.

Tabela 1 – Estrutura da rede VGG-16.

Camada	Filtros	Dimensão da saída
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
<i>Max Pooling</i>	$2 \times 2 \times 64$	$112 \times 112 \times 64$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
<i>Max Pooling</i>	$2 \times 2 \times 128$	$56 \times 56 \times 128$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
<i>Max Pooling</i>	$2 \times 2 \times 256$	$28 \times 28 \times 256$
Convolutacional	$3 \times 3 \times 256$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 256$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 256$	$28 \times 28 \times 512$
<i>Max Pooling</i>	$2 \times 2 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
<i>Max Pooling</i>	$2 \times 2 \times 512$	$7 \times 7 \times 512$
Totalmente conectada	$1 \times 1 \times 4096$	$1 \times 1 \times 4096$
Totalmente conectada	$1 \times 1 \times 4096$	$1 \times 1 \times 4096$
Totalmente conectada	$1 \times 1 \times 4096$	$1 \times 1 \times 1000$

2.4 *Transfer Learning*

TL é uma técnica utilizada em ML que consiste em utilizar redes neurais já criadas para um específico problema, e utilizá-la em um problema diferente. A intensão na utilização desta técnica é economizar esforço na etapa de treinamento. A utilização de TL encontra benefícios principalmente nas seguintes situações:

- **Treinamento de redes profundas.** O treinamento de uma rede neural com muitas camadas pode requerer semanas para convergir, mesmo com recursos computacionais em grande quantidade. Assim, ao se utilizar TL, não apenas se reduzem os recursos necessários, mas também o tempo até a geração de resultados.
- **Banco de dados reduzido.** Em algumas situações o banco de dados disponível para o treinamento pode ser muito reduzido, seja pela raridade dos dados rotulados corretamente, pelo custo relacionado à coleta e classificação dos dados ou por outro motivo específico da aplicação. A utilização de bancos de dados reduzidos pode levar a um aprendizado muito específico em que a rede apenas sabe tratar os dados do

Tabela 2 – Estrutura da rede VGG-19.

Camada	Filtros	Dimensão da saída
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
<i>Max Pooling</i>	$2 \times 2 \times 64$	$112 \times 112 \times 64$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
<i>Max Pooling</i>	$2 \times 2 \times 128$	$56 \times 56 \times 128$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
<i>Max Pooling</i>	$2 \times 2 \times 256$	$28 \times 28 \times 256$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
<i>Max Pooling</i>	$2 \times 2 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
<i>Max Pooling</i>	$2 \times 2 \times 512$	$7 \times 7 \times 512$
Totalmente conectada	$1 \times 1 \times 4096$	$1 \times 1 \times 4096$
Totalmente conectada	$1 \times 1 \times 4096$	$1 \times 1 \times 4096$
Totalmente conectada	$1 \times 1 \times 1000$	$1 \times 1 \times 1000$

conjunto de treinamento. Desta maneira, a rede não generaliza bem e não consegue trabalhar com dados de teste.

Na utilização de TL, as camadas mais altas, onde as características mais específicas do problema original se encontram, são retiradas. Caso seja identificada a necessidade, novas camadas podem ser inseridas de acordo com a tarefa. As camadas mais baixas que tratam de características mais abrangentes podem manter os seus pesos ou ajustá-los em pequena proporção, de maneira que o menor processamento possível seja utilizado.

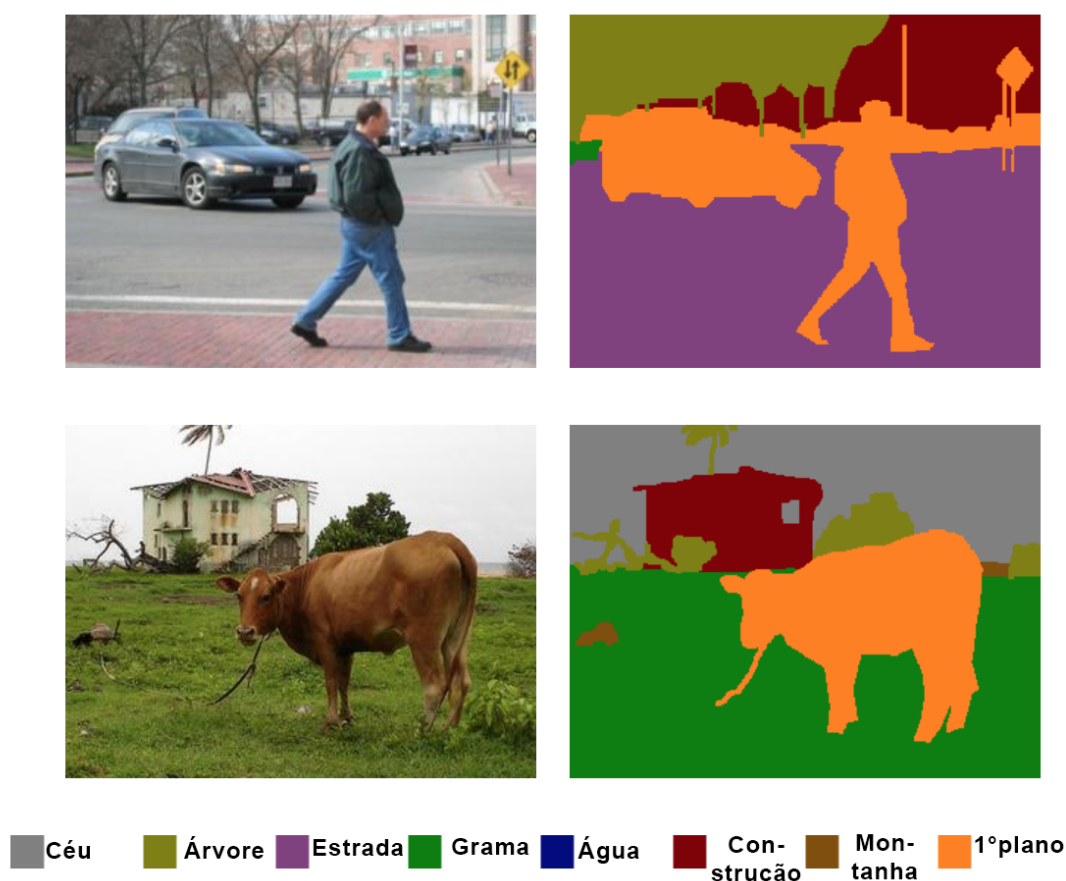
2.5 Segmentação semântica

De maneira geral, Segmentação Semântica é a classificação de cada pixel de uma imagem dentro de um conjunto de classes pré especificadas. O resultado desse processo é uma imagem que contém a classe correspondente a cada pixel da imagem de entrada, como

pode ser visto na Figura 7. Por se tratar, basicamente, de uma classificação de cada pixel, a tarefa se torna muito mais complexa do que uma simples classificação.

Este trabalho considera duas classes: lesão e fundo. Porém, existem diversos desafios que usam bancos de dados que avaliam a segmentação semântica com mais de duas classes, por exemplo: (i) COCO - *Common objects in context* - 80 classes; (ii) PASCAL VOC - *Visual Objects Classes* - 20 classes (encerrou em 2012, porém seu banco de dados ainda é muito utilizado). Existe ainda o ISBI - *International Symposium on Biomedical Imaging* (Simpósio Internacional de Imagem Biomédica, tradução livre), o qual possui diversos desafios da área médica, vários deles de segmentação semântica. O concurso ISIC foi parte do ISBI no ano de 2017.

Figura 7 – Exmplo de Segmentação Semântica

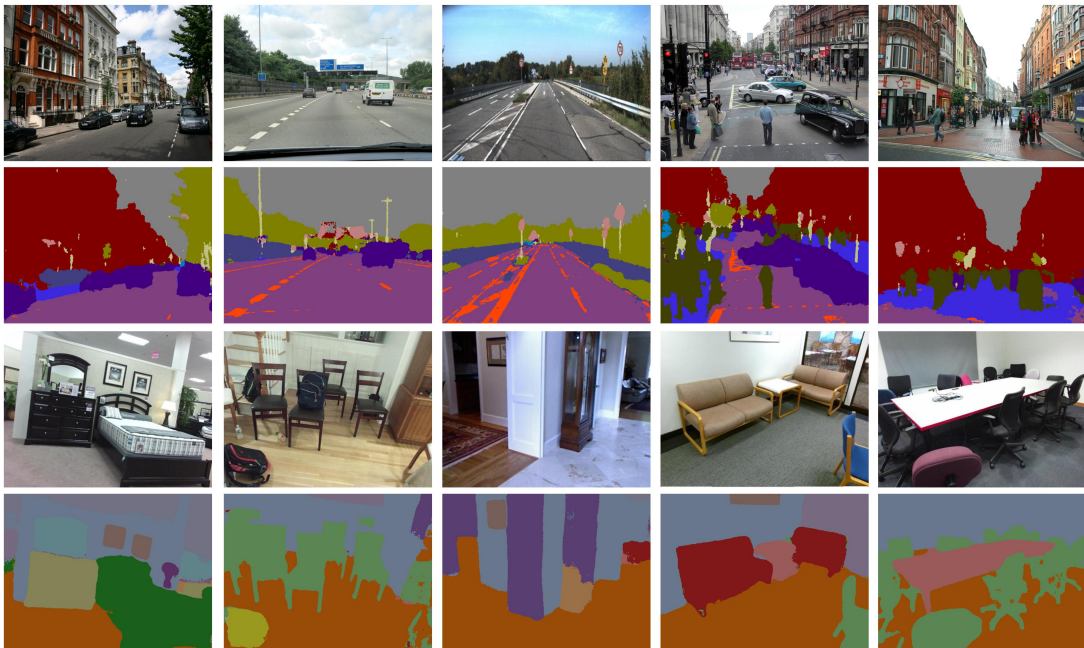


Fonte: (CHEN,) (traduzido)

2.6 SegNet

A rede neural SegNet proposta por (BADRINARAYANAN; KENDALL; CIPOLLA, 2017), é definida pelos autores como uma rede de arquitetura profunda *encoder-decoder* para segmentação multi-classe pixel a pixel. Exemplos de aplicação da SegNet são mostrados na Figura 8.

Figura 8 – Exemplos de imagens submetidas a SegNet em segmentação semântica



Fonte: (GILLEMAN, b)

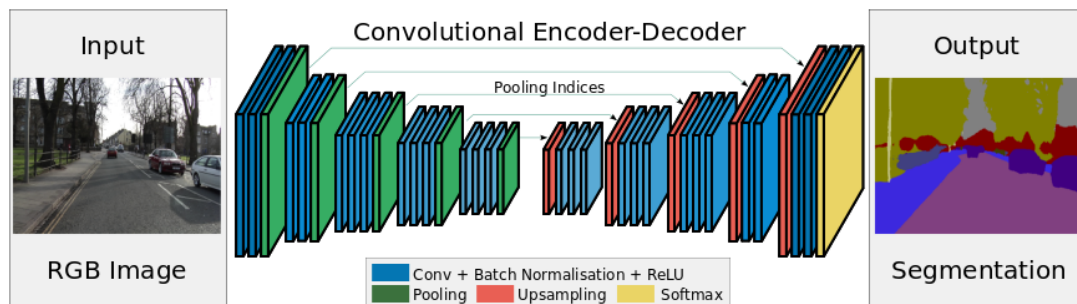
A arquitetura da rede SegNet pode ser observada na Figura 9, ela consiste de camadas de processamento não linear, também chamadas de *encoders* e um conjunto equivalente de *decoders* em série com um classificador pixel a pixel. Especificamente:

- **Encoder.** Formado por camadas convolucionais com *batch normalization*¹, função de ativação ReLu, e camadas de *Max Pooling*. O objetivo do *encoder* é armazenar em um mapa de resolução reduzida todas as características locais e espaciais da imagem original.
- **Decoder.** Formado por camadas convolucionais com *batch normalization* e camadas de *UnPooling* sem utilizar funções de ativação ReLu. A função do *decoder* é criar mapas de características na resolução do objeto de entrada utilizando o mapa de características de baixa resolução resultante do *encoder*

¹ *batch normalization* é uma técnica que regula todos as características para escalas de 0 a 1. Essa operação acelera o processo de treinamento.

- **Classificador pixel a pixel.** Neste caso se trata de uma função *softmax* que tem como entrada os mapas de característica gerados pelo *decoder* para gerar, pixel a pixel, uma distribuição probabilística do pixel pertencer a cada classe. Cada camada do *one hot encoder* possui a probabilidade daquele pixel pertencer a essa classe. Somando-se todas as camadas, a probabilidade resulta em 1. A camada que possui o maior valor, então, identifica a classe em que aquele pixel pertence.

Figura 9 – Arquitetura da SegNet



Fonte: (BADRINARAYANAN; KENDALL; CIPOLLA,)

É importante notar que o *encoder* da SegNet se trata de uma rede VGGNet-16 sem a camada de classificação, implicando que a VGGNet-16 seja usada como extrator de características. Nesse sentido, os pesos originais da VGGNet-16 são usados. O *decoder* funciona como uma VGGNet-16 “reversa”. Essa estrutura pode ser observada na descrição da rede apresentada na tabela 3.

Na literatura existem diferentes trabalhos que usam a rede SegNet, como os citados a seguir:

- Em (BADRINARAYANAN; KENDALL; CIPOLLA, 2017) os autores utilizam DL para fazer segmentação semântica de imagens de trânsito veicular para utilização em veículos autônomos. No mesmo artigo os autores aplicam a rede a imagens internas residenciais para a segmentação de objetos, móveis e outras estruturas como paredes e teto.
- Em (SONG et al., 2015) os autores utilizam SegNet para segmentar o corpo humano em cenas variadas.
- Em (BREBISSON; MONTANA, 2015) os autores utilizam SegNet para fazer a segmentação de partes anatômicas do cérebro humano.
- Em (KALINOVSKY; KOVALEV, 2016) é utilizada SegNet para segmentar o pulmão humano em imagens de raios x.

Tabela 3 – Estrutura da rede SegNet.

Camada	Filtros	Dimensão da saída
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
<i>Max Pooling</i>	$2 \times 2 \times 64$	$112 \times 112 \times 64$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
<i>Max Pooling</i>	$2 \times 2 \times 128$	$56 \times 56 \times 128$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
<i>Max Pooling</i>	$2 \times 2 \times 256$	$28 \times 28 \times 256$
Convolutacional	$3 \times 3 \times 256$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 256$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 256$	$28 \times 28 \times 512$
<i>Max Pooling</i>	$2 \times 2 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
<i>Max Pooling</i>	$2 \times 2 \times 512$	$7 \times 7 \times 512$
<i>UnPooling</i>	$2 \times 2 \times 512$	$7 \times 7 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$14 \times 14 \times 512$
<i>UnPooling</i>	2×2	$14 \times 14 \times 512$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
Convolutacional	$3 \times 3 \times 512$	$28 \times 28 \times 512$
<i>UnPooling</i>	2×2	$28 \times 28 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
Convolutacional	$3 \times 3 \times 256$	$56 \times 56 \times 256$
<i>UnPooling</i>	2×2	$56 \times 56 \times 128$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
Convolutacional	$3 \times 3 \times 128$	$112 \times 112 \times 128$
<i>UnPooling</i>	2×2	$112 \times 112 \times 64$
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
Convolutacional	$3 \times 3 \times 64$	$224 \times 224 \times 64$
Convolutacional	$3 \times 3 \times \text{NumeroDeClasses}$	$224 \times 224 \times \text{NumeroDeClasses}$

2.7 Resumo

Deste capítulo pode-se observar que: as CNN estão cada vez mais sendo utilizadas em aplicações de visão computacional. O campo de pesquisa de DL tem avanços contínuos, como pode ser observado na utilização da VGG como parte da estrutura da SegNet. Suas

aplicações são diversas, indo de veículos autônomos até análise biomédica. A seguir é descrito os passos para real execução do trabalho, tal que se tomou como arquitetura a SegNet sendo adicionadas etapas de pré-processamento. Portanto, o seguinte capítulo está dedicado especificamente às etapas de processamento e classificação.

3 PROPOSTA

3.1 Introdução

Nesse capítulo é apresentada a solução implementada para a tarefa de segmentação de lesões de pele. Tal proposta está baseada em duas etapas, especificamente:

- **Etapla de Preparo dos Dados.** Aqui, cada imagem dermatoscópica é redimensionada.
- **Etapla de Segmentação.** Cada pixel da imagem dermatoscópica de entrada redimensionada é classificada como pertencente a uma de duas possíveis classes: fundo ou objeto de interesse (lesão de pele). Tal tarefa é modelada como um problema de segmentação semântica, de modo que, é usada a rede SegNet para efetuar tal trabalho, nesse sentido, a saída da rede é uma imagem binária, onde os pixels correspondentes à região de lesão de pele recebem o rótulo 1, caso contrário 0.

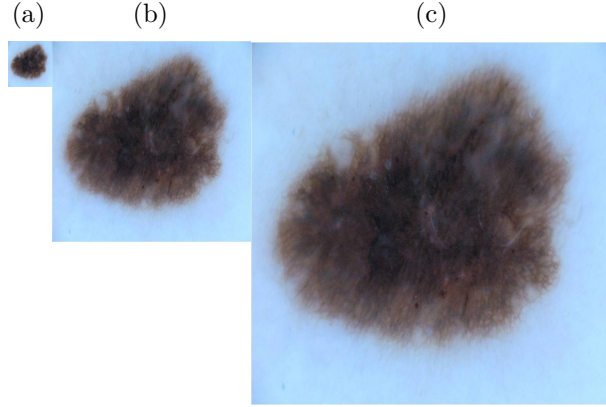
Cada etapa será explicada em detalhes nas próximas seções.

3.2 Etapla de Preparo dos Dados

O objetivo desta etapa é redimensionar o tamanho das imagens dermatoscópicas do banco de dados em estudo a um tamanho específico, sendo ele 128×128 pixels ou 576×576 pixels. Uma comparação de nitidez das lesões para diferentes resoluções é mostrada na Figura 10.

- O valor de 128×128 foi escolhido por utilizar poucos recursos computacionais e ser, então, conveniente para testes.
- O valor de 576×576 foi escolhido por ser a menor dimensão original de imagem existente no banco de dados em estudo, como será comentado na seção 4.2.1, sendo assim o maior valor possível de redimensionamento em que não fosse necessária expansão de imagens e possível inserção de elementos por esse processo.

Figura 10 – Imagens de entrada em diferentes resoluções: (a) imagem 128×128 ; (b) imagem 576×576 ; (c) imagem em tamanho original.



Fonte: Banco de Dados ISIC *Archive*

3.2.1 Etapa de Segmentação

O problema de aprendizado supervisionado pode ser formulado de forma geral como: seja, $\mathcal{T} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, o conjunto de treinamento onde $\mathbf{x}_i \in \mathcal{X}$ é o i -ésimo ponto amostrado e $y_i \in \mathbb{R}$ é o rótulo de classe correspondente ao ponto \mathbf{x}_i . Suponha-se que existe uma função objetivo f parametrizada por θ que relacionam cada ponto amostrado \mathbf{x} com seu correspondente rótulo associado y . Então, tomando em conta o conjunto de treinamento \mathcal{T} , deseja-se determinar os parâmetros θ que minimizem uma função de custo L , formalmente

$$\theta_{min} = \underset{\theta}{\operatorname{argmin}} L(\mathcal{T}, \theta). \quad (3.1)$$

Tarefas clássicas de ML que podem ser modeladas como um problema de aprendizado supervisionado são as de classificação e regressão de dados.

Em *Structured Output Learning* - SOL (tradução livre, aprendizado de saídas estruturadas), o problema de aprendizado supervisionado é generalizado em relação à saída esperada, ou seja, a saída esperada já não é um rótulo em si, de forma geral, a saída fica definida como um tipo de objeto estruturado¹ (imagem, texto, áudio, etc). Nesse contexto, o problema de segmentação semântica pode ser formulado como um problema de SOL. Para o caso em estudo, o problema de segmentação de uma imagem dermatoscópica pode ser formulado como segue: seja, $\mathcal{T} = \{(\mathbf{I}_1, \mathbf{B}_1), \dots, (\mathbf{I}_N, \mathbf{B}_N)\}$, o conjunto de treinamento, onde $\mathbf{I}_i \in [0, 255]^{N \times M \times 3}$ é a i -ésima imagem dermatoscópica de entrada, $\mathbf{B}_i \in [0, 1]^{N \times M}$

¹ Uma definição *ad hoc* de dados estruturados é a seguinte: são dados que consistem de várias partes, e não apenas as partes em si contêm informação, mas também a maneira pela qual as partes estão juntas.

é a imagem binária correspondente \mathbf{I}_i . Suponha-se que existe uma função objetivo $\mathbf{f} : [0, 255]^{N \times M \times 3} \rightarrow [0, 1]^{N \times M}$ parametrizada por $\boldsymbol{\theta}$ que relacionam cada imagem \mathbf{I} com sua correspondente representação binária \mathbf{B} . Então, tomando em conta o conjunto de treinamento \mathcal{T} , deseja-se determinar os parâmetros $\boldsymbol{\theta}$ que minimizem uma função de custo L , tal como é indicado na Equação (3.1).

Nesta proposta, foi considerado que:

Em relação à função objetivo \mathbf{f} . Ela vem definida pela estrutura da rede SegNet. Neste caso, a saída gerada por \mathbf{f} corresponde à saída do *decoder* da SegNet. Sendo assim, foi necessário usar na etapa de treinamento a representação *one hot encoding*², denotada por \mathbf{T} , relacionada a cada imagem binária \mathbf{B} do conjunto de treinamento. \mathbf{T} é um tensor de ordem 3, com profundidade equivalente ao número de classes da segmentação semântica, neste caso 2, especificamente $\mathbf{T} \in [0, 1]^{N \times M \times 2}$. Um exemplo do tensor \mathbf{T} é mostrado na Figura 11. Para a etapa de inferência, dado que a saída do *decoder* da SegNet está no formato *one hot encoding*, foi necessário efetuar a conversão a uma máscara de segmentação através da aplicação da função *argmax* sobre a saída gerada pelo *decoder*. A Figura 12 mostra a sobreposição da máscara de segmentação gerada após a aplicação do *argmax* sobre a imagem de entrada.

Em relação à função de custo a minimizar L . Foram testadas duas funções de custo, a Entropia cruzada e o Índice de Jaccard Aproximado. Na explicação a seguir de cada uma das funções de custo citadas, será usada a seguinte notação: \mathbf{T} é o *one hot encoding* relacionado à imagem binária esperada, $\hat{\mathbf{T}}$ é o *one hot encoding* predito (saída do *decoder* da rede SegNet), \odot representa a multiplicação elemento a elemento (produto Hadamard) e $\text{sum}(\bullet)$ é uma função que soma todos os elementos de um tensor.

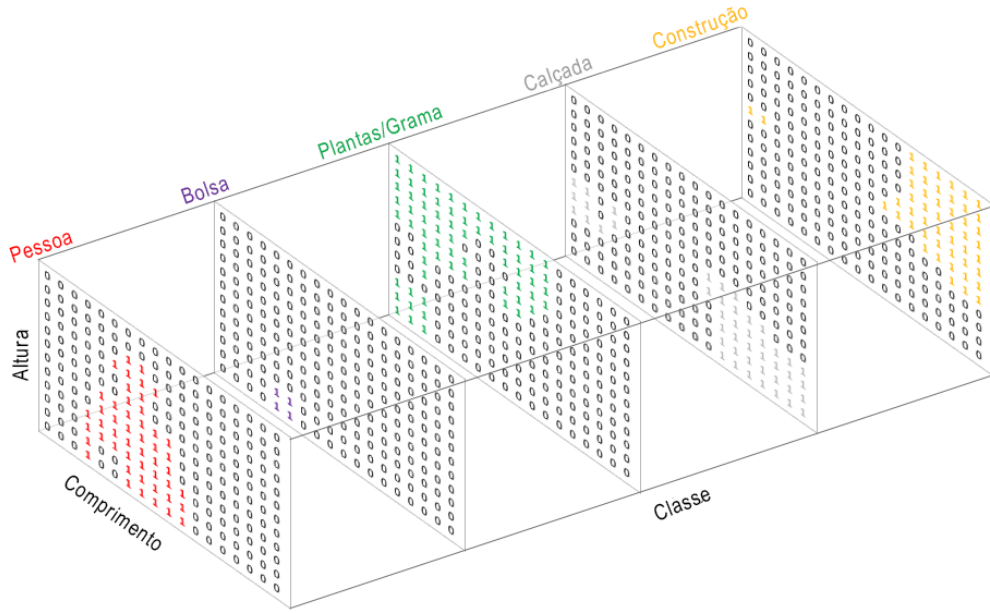
- **Entropia cruzada.** Esta função de custo indica a distância entre a distribuição probabilística que a rede encontra como correta, e a real distribuição da saída esperada. Esta função é amplamente utilizada em trabalhos de segmentação semântica. De maneira matemática, a função pode ser definida como na Equação (3.2).

$$L_{Entropy}(\mathbf{T}, \hat{\mathbf{T}}) = -\frac{1}{NM} \text{sum}(\mathbf{T} \odot \ln(\hat{\mathbf{T}})) \quad (3.2)$$

Onde, $\ln(\bullet)$ representa a função logaritmo neperiano aplicada a cada elemento de um tensor.

² A codificação *one hot encoding* consiste em vários canais, tantos quanto o número de classes para uma segmentação, sendo que cada pixel somente tem valor 1 no canal que representa a classe em que aquele pixel da imagem original estaria representado.

Figura 11 – Representação do tensor \mathbf{T} em *one hot encoding* relacionado a uma máscara de segmentação \mathbf{B} de 5 classes.



Fonte: (JORDAN,) (traduzido)

- **Índice de Jaccard Aproximado.** A utilização do índice de Jaccard de maneira direta como uma função de custo não é aplicável³, usando-se no caso uma aproximação, definida como o índice DICE que se apresenta como:

$$L_{Dice}(\mathbf{T}, \hat{\mathbf{T}}) = 1 - \frac{2 * \text{sum}(\mathbf{T} \odot \hat{\mathbf{T}})}{\text{sum}(\mathbf{T} \odot \mathbf{T}) + \text{sum}(\hat{\mathbf{T}} \odot \hat{\mathbf{T}})} \quad (3.3)$$

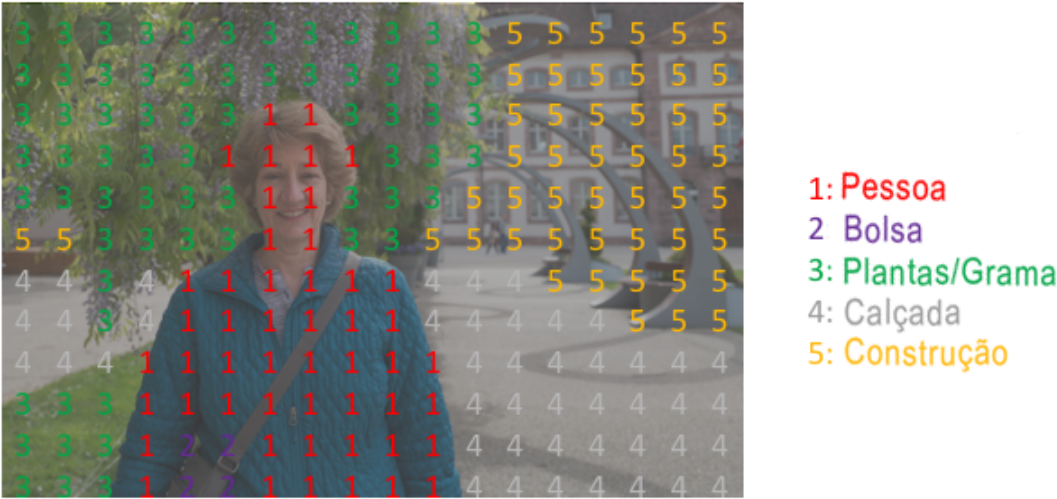
A correlação do índice Dice com o de Jaccard é de tal maneira que quando um índice é 1 o outro também o é. Tal como quando um índice se torna 0 o outro também será.

3.3 Resumo

O que segue no capítulo 4 são especificidades da implementação da solução como limitação de poder computacional e etapas de preparo dos dados. Também serão indicados detalhes dos recursos utilizados, métricas e resultados gerados.

³ O cálculo do índice de Jaccard requer determinar a imagem de segmentação inferida ($\hat{\mathbf{B}}$) a partir da saída predita em formato *one hot encoding* ($\hat{\mathbf{T}}$), fazendo-se necessário o uso da função *argmax*, porém a função *argmax* não é diferenciável, implicando que não cumpra o quesito principal exigido pelos algoritmos de otimização de redes neurais.

Figura 12 – Explicação de one hot encoding



Fonte: (JORDAN,) (traduzido)

4 RESULTADOS

4.1 Introdução

Neste capítulo serão apresentados os resultados obtidos pela técnica proposta. Dessa maneira, o capítulo inicia descrevendo o banco de dados utilizado para treinamento e as etapas de pré-processamento e treino. Também são apresentadas as métricas para avaliar o desempenho da rede tal como o resultado dos experimentos. Por fim são apresentados os campeões do desafio e comparados os resultados dos mesmos com o método proposto neste trabalho.

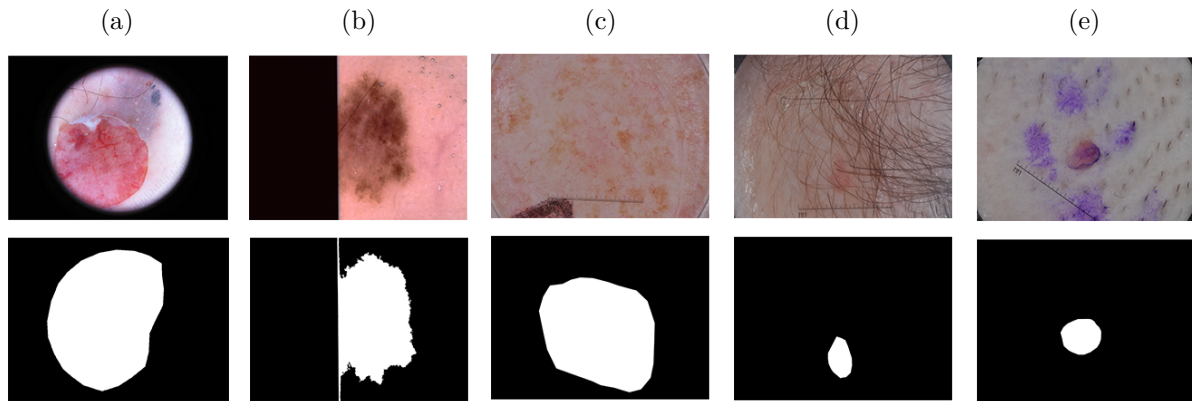
4.2 Alocação de Recursos

4.2.1 Base de dados ISIC 2018

A instituição ISIC possui o maior grupo de imagens dermatoscópicas de lesão de pele rotuladas, escolhidas, organizadas e dispostas publicamente. Atualmente o *ISIC Archive* possui mais de treze mil imagens dermatoscópicas que foram coletadas dos principais centros clínicos do mundo. Tomando como base esse conjunto de imagens, a ISIC criou o *ISIC Challenge Skin Lesion Analysis Towards Melanoma Detection*, onde os participantes são incentivados a desenvolver ferramentas de suporte ao diagnóstico de melanomas utilizando sistemas automáticos de análise de imagem. Este desafio consiste em três atividades: (i) segmentação de lesões; (ii) detecção de características de lesões; (iii) diagnóstico de lesões.

Para o caso do desafio relacionado à segmentação de lesões de pele, a instituição coordenadora do desafio ISIC 2018 oferece uma base de dados de treinamento formada por 2.594 imagens de entrada e uma mesma quantidade de imagens de referência, ou seja, as imagens segmentadas pelos própria ISIC. Da mesma maneira, são fornecidas 100 imagens para validação e 1.000 imagens para testes finais. As imagens de entrada são do formato JPEG, possuem diferentes tamanhos variando entre 576×768 e 6748×4499 pixels, e podem ter elementos específicos, como molduras pretas ou régua métricas, tal como mostrado na Figura 13. As imagens de referência são do formato PNG com tamanhos exatamente iguais às suas respectivas entradas e são representações das localização das lesões de pele dentro da imagem de entrada. Esta representação se dá em máscaras de 8 bits de contraste (256

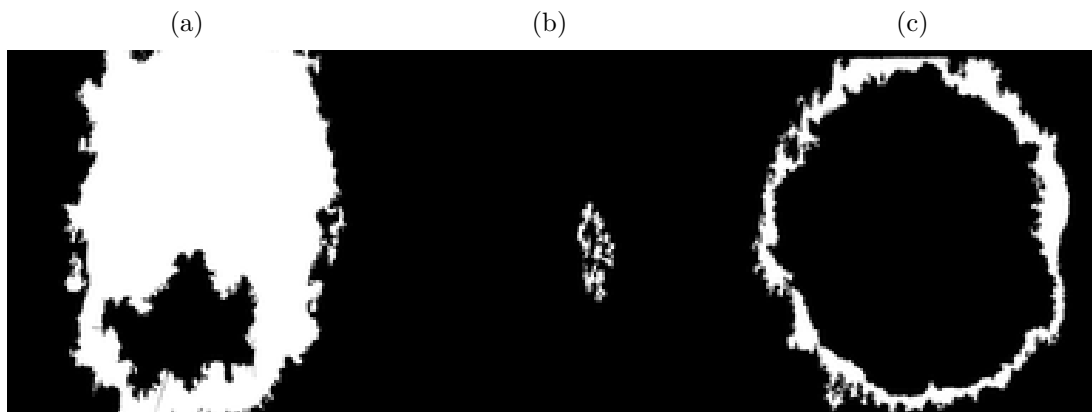
Figura 13 – Entradas e saídas com condições adversas. Em cima imagem original, abaixo a referência: (a) margem circular; (b) margem irregular; (c) manchas secundárias; (d) grande quantidade de pelos; (e) manchas de tinta.



Fonte: Banco de Dados ISIC *Archive*

níveis de cinza), sendo que cada pixel é diferenciado em: 0, que representa área de fundo, ou áreas que não são referentes à lesão principal; 255, que representa a lesão principal. É relevante dizer que no site de descrição do desafio ¹ os criadores do concurso descrevem que a imagem gerada como máscara de segmentação deve ter apenas um objeto branco sem espaços pretos em seu interior, mas esse requisito não é encontrado em suas imagens de referência. Essas ocorrências podem ser exemplificadas pela Figura 14.

Figura 14 – Exemplos de imagens do banco de dados ISIC 2018



Fonte: Banco de Dados ISIC *Archive*

As imagens usadas para treinar a CNN foram divididas em dois conjuntos: treinamento e validação. O conjunto de validação consiste em 100 imagens aleatoriamente retiradas do conjunto de treinamento original. Esse procedimento é realizado tendo em vista que o conjunto de validação original fornecido pela ISIC não possui as imagens de referência,

¹ <<https://challenge2018.isic-archive.com/task1/>>

impossibilitando a comparação das imagens preditas com as originais. As métricas oficiais eram geradas ao se submeter as imagens preditas ao sistema do desafio ISIC, que está fechado após ter acabado o período do concurso.

4.2.2 Recursos Computacionais

Recursos de *Software*. O projeto foi inteiramente desenvolvido usando a linguagem de programação Python, por ser uma linguagem com módulos de programação que facilitam o trabalho com redes neurais. Python é uma linguagem interpretada orientada a objeto de alto-nível. Ela foi construída para permitir ao usuário formular soluções complexas com poucas linhas de código e para ser mais legível, quando comparadas com outras linguagens existentes. Para a implementação da rede neural CNN foi utilizado *Tensorflow*, uma biblioteca de código aberto desenvolvido pela *Google brain Team*². A biblioteca oferece recursos que facilitam a criação e treinamento de redes neurais para diversas aplicações. A ferramenta suporta a linguagem de programação *Python*.

Recursos de *Hardware*. Duas plataformas de hardware foram usadas:

- **Plataforma *Google Colaboratory*.** Esta é uma plataforma *online* gratuita criada para disseminar o conhecimento e aplicações de ML. Essa plataforma permite a edição e execução de algoritmos em *Python* com processamento em nuvem. Apesar de suas funcionalidades, existe uma limitação de 12 horas por sessão e um limite de armazenamento total de 12 GB. Caso qualquer uma das duas limitações seja atingida a sessão é encerrada e todo processo, variáveis e arquivos criados são perdidos. É válido comentar que mesmo não atingindo os limites, o serviço é suscetível a encerramento aleatório. O *Colaboratory* possui suporte para geração de gráficos, integração com *TensorFlow* e com máquinas locais e permite instalação de bibliotecas externas de *Python*. Além disso, a máquina utilizada para pré-processamento de imagem, desenvolvimento no *colaboratory* e para cálculo de métricas possui a seguinte configuração: (i) sistema operacional Linux, distribuição Ubuntu Server 16.04; (ii) processador Intel Core i5-3330, 3.00GHz com 4 núcleos físicos; (iii) memória RAM de 8 GB; (iv) unidade de armazenamento de 1TB (disco rígido); (v) placa de vídeo AMD Radeon HD 6670, com 1 GB de memória dedicada.
- **Plataforma do laboratório VIROS.** Para o treinamento da rede utilizando imagens de resolução 576×576 foi utilizado um computador do laboratório VIROS com a seguinte configuração: (i) sistema operacional Linux, distribuição Ubuntu

² <<https://research.google.com/teams/brain/>>

Server 16.04; (ii) processador Intel Core i7-7770, 3.60GHz com 4 núcleos físicos; (iii) memória RAM de 32 GB; (iv) unidade de armazenamento de 1TB (disco rígido); (v) placa de vídeo Nvidia Geforce GTX 1080Ti, com 11 GB de memória dedicada.

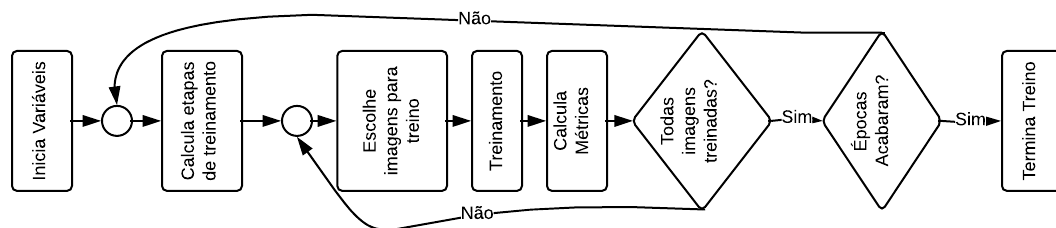
4.3 Detalhes de implementação

Para as próximas descrições serão efetuados alguns comentários sobre os hiper-parâmetros do treinamento de uma rede neural e sobre os passos que formam uma sessão de *Tensorflow* quando uma rede neural é treinada.

A seguir são dadas as definições dos hiper-parâmetros de importância relacionados ao treinamento de uma rede neural, especificamente: (i) **tamanho do *batch***, quantidade de imagens utilizadas para cada iteração de treino; (ii) **taxa de aprendizado**, valor escalar pelo qual é multiplicada a taxa de alteração dos pesos a cada iteração de treinamento; (iii) **número de épocas**, quantidade de vezes em que a rede recebeu o conjunto inteiro de treino durante a aprendizagem.

Em teoria, durante a sessão do *Tensorflow* os seguintes passos deveriam ocorrer: (i) iniciar as variáveis; (ii) começar um laço *for* relativo as épocas de treino; (iii) identificar o número necessário de etapas de treinamento por época, de acordo com o tamanho do *batch* indicado, para que todos os dados sejam submetidos a etapa de treinamento; (iv) criar um *batch* do tamanho indicado com imagens de entrada, retiradas aleatoriamente do banco total de imagens, e suas respectivas imagens de saída; (v) alimentar a seção de otimização (treinamento) com esse *batch* e efetivamente treinar a rede; (vi) calcular métricas de perda e acurácia; (vii) repetir os procedimentos dentro do laço *for* relativo as épocas até que o número de épocas seja alcançado. Esses passos podem ser observados na Figura 15

Figura 15 – Fluxograma dos passos de treinamento



Fonte: Produção do próprio autor.

4.3.1 Sobre o tratamento dos dados do conjunto de treinamento e teste

Para reduzir o tempo de processamento na etapa de treinamento, foi realizada uma etapa de conversão prévia das imagens para *arrays* em pares de entrada e saída. Para isso foi utilizada uma estratégia de converter todas as imagens em *arrays*, criar uma lista de imagens de entrada e uma de imagens de saída esperada em ordem tal que o k -ésimo item da primeira lista seria a imagem de entrada referente ao k -ésimo item da lista de imagens de saída. Então as duas listas são inseridas em um dicionário com chaves para os dois tipos de imagem. Para salvar esse dicionário em um arquivo que pudesse ser guardado e utilizado quando necessário, o dicionário foi serializado com o módulo *Pickle* e guardado em um arquivo de extensão *.pkl*.

Durante os testes foi identificado que a GPU da máquina do VIROS disponível não possui memória suficiente para *batches* de mais de 2 imagens por vez. Também foi identificado que a quantidade de RAM disponível também não era suficiente para fazer um pré-carregamento de todas as imagens ao mesmo tempo. Uma possível solução seria criar vários arquivos *.pkl* e iterar os *batches* dentro dessas pequenas listas. Para evitar um direcionamento do treinamento para um mínimo local relativo a alguma parte específica do banco de dados, foi realizada troca de ordem aleatória de todos os elementos do banco de dados. Então foram criados 30 arquivos *.pkl* que torna então possível o carregamento do banco de imagens por partes e o treinamento se torna menos tendencioso à um mínimo local.

Para aumentar a generalização do segmentador é possível realizar *data augmentation*, ou seja, aplicar distorções iguais nas entradas e saídas para aumentar a quantidade de dados diferentes usados pela rede para treinar. Como já foi comentado, a RAM é um limitador no processo de treinamento, então não é possível gerar novas imagens distorcidas e guardar simultaneamente as imagens originais. Dessa maneira, o processo de *data augmentation* é feito *on-the-fly*, ou seja, durante a seção do *Tensorflow*. Apesar de resolver o problema, aumenta o tempo de processamento da seção, aumentando, então, o tempo total de treino da rede.

4.3.2 Sobre a implementação da rede SegNet

Para a implementação da rede SegNet em *Tensorflow* foi utilizado um modelo disponível em *GitHub*³. O código não era estruturado exatamente como uma SegNet e estava com falhas de sintaxe, então foram realizadas correções para que o mesmo funcionasse e aceitasse

³ <https://github.com/sjchoi86/Tensorflow-101/blob/master/notebooks/semseg_basic.ipynb>

os dados de entrada e saída do banco de dados. Nesse sentido, como a etapa de conversão de imagens em *arrays* foi realizada em etapas precedentes, e as seções de treinamento não eram sempre realizadas em máquinas locais, o código foi alterado para que recebesse os dados de entrada em formato *.pkl*, sendo este resgatado automaticamente do *google drive*, para seções no *colaboratory*, ou de pastas específicas, para seções locais na máquina do laboratório VIROS.

4.3.3 Sobre a etapa de treinamento da rede SegNet

Como os dados precisam ser separados em pequenos *batches* para não estourar a memória da máquina, uma etapa extra foi inserida antes da atribuição das imagens para o mini *batch* de treinamento. Nesta etapa, cada parte do banco de imagens, armazenado em um arquivo *.pkl*, é carregado para treino a cada N épocas, sendo:

$$N = \frac{N_{pkl}}{N_{epoch}}. \quad (4.1)$$

onde, N_{pkl} é o número de arquivos *.pkl* e N_{epoch} é o número de épocas escolhido para o treino. Esse carregamento é realizado através de um procedimento que realiza os seguintes passos: (i) carrega o arquivo *.pkl* específico para a época; (ii) converte de *.pkl* para *arrays*; (iii) converte a matriz de rótulos 576×576 para um *tensor* do tipo *one-hot-encoded*, já descrito anteriormente; (iv) retorna a lista de imagens a serem repassadas para o processo de treino.

Para a etapa de treino foi usado o otimizador Adam (KINGMA; BA, 2014). Como foram realizados testes com dois tamanhos de imagem, um menor na plataforma *colaboratory*, e um maior na máquina do laboratório VIROS com limitações diferentes, os procedimentos serão descritos separadamente.

- **Para o conjunto de imagens de tamanho 128×128 .** Foi escolhido 128 como tamanho do *batch* por ser o maior possível sem que ultrapassasse o limite de memória da GPU disponível. Foram realizados vários testes e pode ser notado que para testes com mais de 3.000 épocas começavam a surgir efeitos de *overfitting*.
- **Para o conjunto de imagens de tamanho 576×576 .** Foi identificado uma limitação de memória da GPU que ocasionou a definição do tamanho do *batch* para 2 imagens. Foi definido o número de épocas (N_{epoch}) como 3.000 para compensar a utilização de *batches* de 2 imagens.

Para ambos os conjuntos de imagens foram realizados testes utilizando *data augmentation*. A mesma estratégia foi utilizada nos dois casos. Especificamente:

- Cada *batch* selecionado para o treino foi, primeiramente, submetido a operações de rotação e espelhamento. A aleatoriedade da distorção foi realizada aplicando uma rotação de $k \times 90^\circ$, sendo k um número aleatoriamente calculado entre 0 e 3, além de um possível espelhamento em uma probabilidade de 1 em 2. Desta maneira, uma mesma imagem de entrada poderia ser submetida ao treino de 8 maneiras diferentes. Esta aleatoriedade foi utilizada para não tendenciar o treinamento da rede para um tipo de operação específico.

4.3.4 Sobre a etapa de predição da rede SegNet

Após o treino, todas as 2.594 imagens são inseridas na rede para que sejam geradas predições. Vale notar que somente as 100 imagens não utilizadas para o treino são consideradas para o cálculo das métricas. As outras imagens preditas somente foram geradas para ajudar a avaliar dificuldades da rede e predições peculiares.

4.4 Parte Experimental

4.4.1 Métricas

Para determinar a pontuação de cada participante no desafio ISIC 2018 relacionado à segmentação de lesões de pele os avaliadores utilizaram a média limiarizada do índice de Jaccard (JI). O índice de Jaccard se trata de uma relação de similaridade entre a imagem esperada e a predita, tal que seu valor é calculado pela interseção das áreas predita e esperada sobre a união das mesmas. Tal índice é calculado via a Equação (4.2) e representado para uma melhor compreensão na Figura 16.

$$JI = \frac{|\mathbf{B} \cap \hat{\mathbf{B}}|}{|\mathbf{B} \cup \hat{\mathbf{B}}|} \quad (4.2)$$

Aqui, \mathbf{B} representa a imagem binária esperada, $\hat{\mathbf{B}}$ representa a imagem binária predita e $|\bullet|$ é o operador de cardinalidade.


$$JI = \frac{|\mathbf{B} \cap \hat{\mathbf{B}}|}{|\mathbf{B} \cup \hat{\mathbf{B}}|} = \frac{\text{área da interseção}}{\text{área da união}}$$


Figura 16 – Representação do Índice de Jaccard.

A pontuação de um participante é calculado da seguinte maneira: (i) cada imagem binária predita é comparada com a respectiva imagem binária esperada utilizando o índice de Jaccard definido na Equação (4.2); (ii) a pontuação de cada imagem é calculada como uma limiarização do índice de Jaccard, tal como é indicado na Equação (4.3); (iii) é realizada uma média do JI resultante de todo o conjunto de validação/teste.

$$JI = \begin{cases} 0 & JI < 0,65 \\ JI & JI \geq 0,65 \end{cases} \quad (4.3)$$

Como é descrito em (CODELLA et al., 2017), somente o índice de Jaccard como métrica não é uma medida correta para avaliação de segmentação por não identificar exatamente em qual imagem a segmentação falha. Desta maneira, apesar de não serem utilizadas para classificação no concurso, outras métricas a nível de pixels foram utilizadas a fim de oferecer uma melhor compreensão do desempenho da rede. São essas métricas: (i) acurácia (Ac), que avalia a capacidade da técnica de classificar corretamente os pixels como objeto ou fundo; (ii) sensibilidade (Se), que avalia a capacidade da técnica em classificar corretamente os pixels como objeto; (iii) especificidade (Es), que avalia a capacidade da técnica em classificar corretamente os pixels como fundo; e finalmente a medida CD que é coeficiente de Dice, ou a relação de similaridade entre imagem predita e referência. As relações que definem essas métricas são apresentadas nas Equações (4.4) – (4.7), onde: VP são os verdadeiros positivos (número de pixels corretamente classificados pela técnica como objeto), VN são os verdadeiros negativos (número de pixels corretamente classificados pela técnica como fundo), FP são os falsos positivos (número de pixels classificados erroneamente pela técnica como objeto), e FN são falsos negativos (número de pixels classificados erroneamente pela técnica como fundo).

$$Ac = \frac{VP + VN}{VP + VN + FP + FN} * 100 \quad (4.4)$$

$$Se = \frac{VP}{VP + FN} * 100 \quad (4.5)$$

$$Es = \frac{VN}{VN + FP} * 100 \quad (4.6)$$

$$CD = \frac{2VP}{2VP + FP + FN} * 100. \quad (4.7)$$

4.4.2 Experimento Efetuados

Foram efetuados dois experimentos. O primeiro experimento, treinando a rede SegNet usando a função de custo entropia cruzada ($L_{Entropy}$), e o segundo experimento, treinando a rede SegNet usando a função de custo índice de Jaccard aproximado (L_{Dice}). Em cada experimento foi usado o seguinte procedimento: (i) para cada conjunto de imagens (de 128×128 ou 576×576) a rede SegNet é treinada considerando o uso ou não de *data augmentation*; (ii) uma vez treinada a rede, é efetuado o processo de inferência usando o conjunto de teste e as métricas de validação são calculadas. Os resultados desses experimentos estão expostos nas Tabelas 4 e 5.

Tabela 4 – Valores das métricas de desempenho na segmentação das imagens de lesão de pele usando a função de custo entropia cruzada $L_{Entropy}$.

Metodologia	JI	$Se(\%)$	$Es(\%)$	$Ac(\%)$	CD
128×128	29,73	82,04	93,56	86,04	70,71
$128 \times 128 + Data\ augmentation$	29,81	82,54	92,54	85,43	61,19
576×576	58,51	87,20	97,72	98,84	72,75
$576 \times 576 + Data\ augmentation$	59,99	82,89	98,79	99,68	90,64

Tabela 5 – Valores das métricas de desempenho na segmentação das imagens de lesão de pele usando a função de custo índice de Jaccard aproximado L_{Dice} .

Metodologia	JI	$Se(\%)$	$Es(\%)$	$Ac(\%)$	CD
128×128	27,95	83,74	90,95	84,43	59,48
$128 \times 128 + Data\ augmentation$	31,99	82,05	92,90	85,42	60,55
576×576	61,2	85,82	98,21	99,35	82,67
$576 \times 576 + Data\ augmentation$	62,87	85,30	98,46	99,52	86,66

Alguns comentários sobre as Tabelas 4 e 5:

- O experimento que obteve o melhor resultado foi usando o conjunto de imagens de 576×576 , treinando a rede SegNet com *data augmentation* e função de custo o índice de Jaccard aproximado (L_{Dice}). Tal experimento atingiu na etapa de inferência um índice de Jaccard médio limiarizado de 62,87%. Em geral, é possível observar um elevação das métricas como um todo, quando se utiliza um tamanho maior de imagem e *data augmentation*.
- Considerando o conjunto de imagens com resolução 576×576 , a proposta obteve como resultado médio do índice de Jaccard um valor próximo do limiar de desconsideração de resultados por imagem. Sendo assim, muitas imagens se encontravam abaixo do limiar, o que acarretava em um menor índice de Jaccard médio limiarizado (efetiva métrica do desafio).

- Através das outras métricas e das imagens preditas foi possível perceber também que a técnica proposta gera segmentações menos “conservadoras” do que as geradas por médicos e utilizadas como referência, ou seja, segmentava a lesão em si com menos bordas de segurança. Essas bordas de segurança podem ser utilizadas por médicos ao se avaliar a área a ser retirada para biópsia.
- Os treinos realizados com a dimensão de 128×128 duraram cerca de 6 horas ,enquanto os treinos com a dimensão de 576×576 duraram cerca de 30 horas.

Ao serem geradas imagens segmentadas do conjunto de teste, é possível identificar algumas dificuldades nas imagens segmentadas via a rede SegNet. De maneira geral, a rede tende a gerar máscaras de menor tamanho do que a máscara original, a não ser quando existem outros objetos na imagem dermatoscópica. Foi mostrado que existem imagens em que a rede consegue ignorar os adesivos na pele como na Figura 17f e casos em que não foi possível, como na Figura 17d. É interessante notar o exemplo da Figura 17e onde a imagem original é maior e menos uniforme do que a predita. A mesma imagem mostra que a figura de referência não ignora a régua da imagem colorida, mas a predita sim. Por fim, vale notar o desempenho da rede nas Figuras 17b e 17c onde as lesões são densas e de diversos tons da cor principal, muitos espaços marcados erroneamente como fundo aparecem. Já na Figura 17a mesmo com vários pelos sobrepostos a lesão principal, o resultado fica muito próximo da imagem de referência.

4.4.2.1 Comparação de resultados

Ao final da competição, mais de 200 equipes submeteram seus trabalhos. A Tabela 6 indica o nome das 10 equipes melhor colocadas e seus índices de desempenho. É importante notar que era possível submeter mais de um método de segmentação, então algumas equipes possuíram mais de um resultado no pódio.

A classificação geral é medida pela média limiarizada do índice de Jaccard de todas as imagens.

Não é possível realizar uma comparação exata com o pódio pois as imagens estadas não são iguais por motivos já explicados. Ainda assim, o resultado obtido foi inferior aos obtidos pelos participantes no pódio. Se comparado a lista de participantes completa, a rede proposta se situaria na posição 108º no desafio ISIC 2018.

Tabela 6 – *Ranking* para o problema de **segmentação de lesão** do desafio ISIC 2018.

Posição	Equipe	desempenho
1	chengyao qian	0.802
2	Youngseok0001	0.799
3	JiYuanFeng JiYuanFeng	0.799
4	Yuan Xue	0.798
5	Navid Alemi Koohbanani	0.796
6	chengyao qian	0.794
7	Youngseok0001	0.794
8	Youngseok0001	0.794
9	Miguel Molina	0.788
10	Navid Alemi Koohbanani	0.784

Fonte – Produção do próprio autor

Uma breve descrição dos métodos usados pelos três primeiros colocados no *ranking* geral permite uma comparação com o método proposto neste trabalho.

1. **ISIC 2018 – Skin Lesion Analysis** Fizeram uso de:

- pré-processamento: normalização da iluminância e da cor, redimensionamento para 512×512 ;
- arquitetura do classificador: *Ensemble* de Mask-RCNN com uma rede de segmentação própria baseada em *DeepLab* e *PSPNet* contendo uma *ResNet 101*
- treino: Treino completo dos pesos utilizando otimização de Adam e parando o treino quando a rede dava sinais de *overfitting*;
- *data augmentation*: rotação, distorção de cor e espelhamento;

2. **Team HolidayBurned at ISIC CHALLENGE 2018** (GONZÁLEZ-DÍAZ, 2017) . Fez uso de:

- pré-processamento: normalização da iluminância e da cor, redimensionamento para 192×256 ;
- arquitetura do classificador: *Ensemble* de DeepLab com outras 5 redes incluindo *DenseNet* , *U-net*, VGG16 e *Inception v3*
- treino: Treino utilizando *transfer learning* de todos os modelos do *Ensemble* e ajuste fino;
- *data augmentation*: rotação, distorção de cor, translação, espelhamento e recorte;

3. **Automatic Skin Lesion Segmentation by Feature Aggregation Convolutional Neural Network** (GONZÁLEZ-DÍAZ, 2017) . Fez uso de:

- pré-processamento: redimensionamento para 512×512 ;
- arquitetura do classificador: rede própria utilizando como base *ResNet34*
- treino: Treino completo dos pesos utilizando otimização gradiente descendente com momentum;
- *data augmentation*: rotação, distorção de cor e iluminação, translação, espelhamento e recorte;

É importante notar que os melhores classificados utilizaram *ensembles*, arquiteturas de redes neurais próprias, *transfer learning* e *data augmentation* para melhorarem seus resultados.

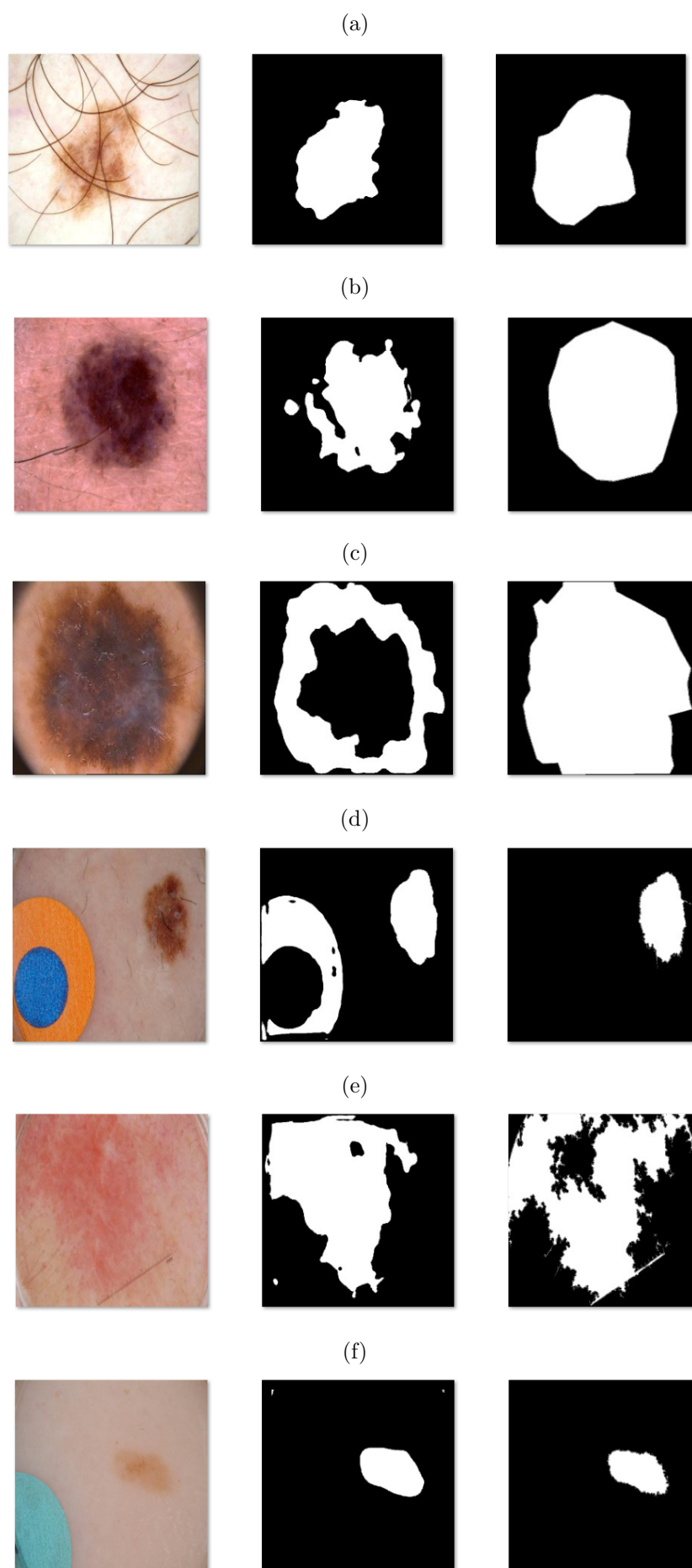
Apesar de *transfer learning* teoricamente acelerar o processo de treino e talvez entregar um resultado melhor do que treinar os pesos totalmente, não foi possível utilizar esta técnica por dificuldades de implementação ao tentar integrar essa funcionalidade ao código utilizado.

Data augmentation foi utilizada mas não de maneira tão avançada como nos trabalhos citados, em sua maioria por falta de recursos computacionais. Dos três artigos comentados, o que utilizou menos procedimentos de *data augmentation* somente realizou distorção de cor como procedimento a mais do que o proposto trabalho. Tendo isso considerado, a mesma equipe utilizou duas GPUs iguais a que estava disponível para o desenvolvimento deste trabalho. Outras equipes possuíam ainda mais poder computacional, chegando a 8 GPUs para servidores (Tesla v100) e 16 vezes mais memória RAM.

4.5 Resumo

Neste capítulo foram descritos os recursos utilizados para a realização deste trabalho, tanto *software* como *hardware*. Da mesma maneira, foram descritos detalhes da implementação da solução proposta, da arquitetura utilizada, do processo de preparo dos dados, do treino e de dificuldades encontradas ao longo do desenvolvimento. Também foi possível avaliar o desempenho da rede neural criada na atividade de segmentação automática de imagens. Foi realizada a comparação com os outros competidores do desafio ISIC 2018 e comentados as diferenças em relação à solução proposta.

Figura 17 – Entradas e saídas com condições adversas. A esquerda imagens originais. No centro imagens preditas, A direita imagens segmentadas originais



5 CONCLUSÕES E PROJETOS FUTUROS

5.1 Conclusões

O objetivo principal deste trabalho foi implementar uma técnica para a segmentação automática de lesões de pele utilizando redes neurais convolucionais profundas. Para isso, foi utilizada a rede SegNet com uma etapa de pré-processamento de imagens. Tal técnica foi avaliada usando o banco de dados fornecido pelo desafio ISIC 2018. Consequentemente, utilizando o índice de Jaccard médio limiarizado como métrica principal, o melhor resultado obtido foi de 62,87%, se colocando então na posição 108º no desafio ISIC 2018. O trabalho permitiu: (i) comprovar os benefícios de usar a arquitetura SegNet baseada em CNN para a tarefa de segmentação de lesões de pele; (ii) perceber a dificuldade de se lidar com imagens médicas, se tratando de imagens com uma diversidade de interferências, métodos de aquisição e diferentes metodologias de anotação de imagens de referência.

5.2 Temas a serem pesquisados

Como trabalhos futuros:

- Adicionar técnicas de pós-processamento como filtragem para manter apenas um objeto e técnicas de preenchimento de furos na imagem binária inferida.
- Devem ser pesquisadas técnicas de *ensemble*, tendo em vista que atualmente as redes neurais consolidadas estão sendo unidas para gerar resultados melhores do que o estado da arte atual.
- Devem ser estudadas mais técnicas de *data augmentation* para que seja possível aumentar o número de épocas sem causar *overfitting*. Desta maneira a rede estaria também mais preparada para realizar uma segmentação mais genérica e não específica ao banco de treinamento.
- Utilizar outras abordagens de pré-processamento de imagens.

REFERÊNCIAS

- AYTAR, Y.; VONDRICK, C.; TORRALBA, A. Soundnet: Learning sound representations from unlabeled video. In: Advances in Neural Information Processing Systems. [S.l.: s.n.], 2016. p. 892–900. Citado na página 24.
- BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA, R. <<http://mi.eng.cam.ac.uk/projects/segnet>>. Acessado em: 2018. Citado na página 29.
- BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE, 2017. Citado 2 vezes nas páginas 28 e 29.
- BAYRAMOGLU, N.; HEIKKILÄ, J. Transfer learning for cell nuclei classification in histopathology images. In: SPRINGER. European Conference on Computer Vision. [S.l.], 2016. p. 532–539. Citado na página 15.
- BREBISSON, A. de; MONTANA, G. Deep neural networks for anatomical brain segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. [S.l.: s.n.], 2015. p. 20–28. Citado na página 29.
- CELEBI, M. E.; WEN, Q.; IYATOMI, H.; SHIMIZU, K.; ZHOU, H.; SCHAEFER, G. A state-of-the-art survey on lesion border detection in dermoscopy images. Dermoscopy Image Analysis, Boca Raton, CRC Press, p. 97–129, 2015. Citado 2 vezes nas páginas 16 e 17.
- CHEN, C. A. A. <<https://thegradient.pub/semantic-segmentation/>>. Acessado em: 2018. Citado na página 27.
- CODELLA, N. C. F.; GUTMAN, D.; CELEBI, M. E.; HELBA, B.; MARCHETTI, M. A.; DUSZA, S. W.; KALLOO, A.; LIOPYRIS, K.; MISHRA, N.; KITTLER, H.; HALPERN, A. Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC). CoRR, p. 2–6, 2017. Citado na página 44.
- CRUZ-ROA, A.; ARÉVALO, J.; JUDKINS, A.; MADABHUSHI, A.; GONZÁLEZ, F. A method for medulloblastoma tumor differentiation based on convolutional neural networks and transfer learning. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. 11th International Symposium on Medical Information Processing and Analysis. [S.l.], 2015. v. 9681, p. 968103. Citado 2 vezes nas páginas 15 e 24.
- ESTEVA, A.; KUPREL, B.; NOVOA, R. A.; KO, J.; SWETTER, S. M.; BLAU, H. M.; THRUN, S. Dermatologist-level classification of skin cancer with deep neural networks. Nature, Macmillan Publishers Limited, part of Springer Nature. All rights reserved., v. 542, p. 115, jan 2017. Citado 2 vezes nas páginas 16 e 17.
- GHAZI, M. M.; YANIKOGLU, B.; APTOULA, E. Plant identification using deep neural networks via optimization of transfer learning parameters. Neurocomputing, Elsevier, v. 235, p. 228–235, 2017. Citado na página 24.

- GILLEMANN, D. <<http://www.deeplearningessentials.science/convolutionalNetwork/>>. Acessado em: 2018. Citado na página 20.
- GILLEMANN, D. <<http://mi.eng.cam.ac.uk/projects/segnet>>. Acessado em: 2018. Citado na página 28.
- GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2014. p. 580–587. Citado na página 14.
- GONZÁLEZ-DÍAZ, I. Incorporating the knowledge of dermatologists to convolutional neural networks for the diagnosis of skin lesions. CoRR, abs/1703.01976, 2017. Citado na página 47.
- GONZALEZ, R. C.; WOODS, R. E. Digital image processing. [S.l.]: Upper Saddle River, NJ: Prentice Hall, 2012. Citado na página 14.
- GOOD, O. How google translate squeezes deep learning onto a phone. 2015. <<https://ai.googleblog.com/2015/07/how-google-translate-squeezes-deep.html>>. Acessado em 21/05/2018. Citado na página 14.
- HUBEL, D. H.; WIESEL, T. N. Receptive fields and functional architecture of monkey striate cortex. The Journal of Physiology, v. 195, n. 1, p. 215–243, mar 1968. ISSN 0022-3751. Citado na página 15.
- INCA, . Brasil. Instituto Nacional de Câncer José Alencar Gomes da Silva. Coordenação de Prevenção e V. Estimativa 2018: incidência de câncer no Brasil. Rio de Janeiro: [s.n.], 2017. Citado na página 13.
- JADERBERG, M.; SIMONYAN, K.; ZISSERMAN, A. et al. Spatial transformer networks. In: Advances in neural information processing systems. [S.l.: s.n.], 2015. p. 2017–2025. Citado na página 21.
- JORDAN, J. <<https://www.jeremyjordan.me/semantic-segmentation/>>. Acessado em: 2018. Citado 2 vezes nas páginas 35 e 36.
- KALINOVSKY, A.; KOVALEV, V. Lung image segmentation using deep learning methods and convolutional neural networks. Minsk: Publishing Center of BSU, 2016. Citado 2 vezes nas páginas 14 e 29.
- KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014. Citado na página 42.
- KITTLER, H.; PEHAMBERGER, H.; WOLFF, K.; BINDER, M. Diagnostic accuracy of dermoscopy. The Lancet Oncology, Elsevier, v. 3, n. 3, p. 159–165, dec 2017. ISSN 1470-2045. Citado na página 13.
- KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. Advances In Neural Information Processing Systems, p. 1–9, 2012. Citado na página 20.
- LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. Nature, v. 521, n. 7553, p. 436–444, 2015. ISSN 14764687. Citado na página 15.

- LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. Neural computation, MIT Press, v. 1, n. 4, p. 541–551, 1989. Citado na página 23.
- LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE, v. 86, n. 11, p. 2278–2323, 1998. ISSN 00189219. Citado na página 15.
- LECUN, Y.; BOTTOU, L.; ORR, G. B.; MÜLLER, K. R. Efficient backprop. In: _____. Neural Networks: Tricks of the Trade. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. p. 9–50. Citado na página 20.
- LEE, S. U.; CHUNG, S. Y.; PARK, R. H. A comparative performance study of several global thresholding techniques for segmentation. Computer Vision, Graphics, and Image Processing, Elsevier, v. 52, n. 2, p. 171–190, 1990. Citado 2 vezes nas páginas 14 e 16.
- MENEGOLA, A.; TAVARES, J.; FORNACIALI, M.; LI, L. T.; AVILA, S.; VALLE, E. Recod titans at isic challenge 2017. arXiv preprint arXiv:1703.04819, 2017. Citado na página 16.
- Ministério da saúde. Sistema de informações sobre mortalidade. 2017. <<http://www.datasus.gov.br>>. Acessado em: 13/09/2017. Citado na página 13.
- NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. USA: Omnipress, 2010. (ICML'10), p. 807–814. Citado na página 20.
- NG, H.-W.; NGUYEN, V. D.; VONIKAKIS, V.; WINKLER, S. Deep learning for emotion recognition on small datasets using transfer learning. In: ACM. Proceedings of the 2015 ACM on international conference on multimodal interaction. [S.l.], 2015. p. 443–449. Citado na página 24.
- NOH, H.; HONG, S.; HAN, B. Learning deconvolution network for semantic segmentation. In: Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2015. p. 1520–1528. Citado na página 22.
- OLIVEIRA, S. A.; SEGUIN, B.; KAPLAN, F. dhsegment: A generic deep-learning approach for document segmentation. arXiv preprint arXiv:1804.10371, 2018. Citado na página 14.
- RUSSAKOVSKY, O.; DENG, J.; SU, H.; KRAUSE, J.; SATHEESH, S.; MA, S.; HUANG, Z.; KARPATY, A.; KHOSLA, A.; BERNSTEIN, M.; BERG, A. C.; FEI-FEI, L. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), v. 115, n. 3, p. 211–252, 2015. Citado na página 23.
- SIMONYAN, K.; ZISSERMAN, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations (ICRL), abs/1409.1, p. 1–14, 2015. Citado na página 23.
- SLIDESHARE. <<https://image.slidesharecdn.com/cnnmodels-170714134232/95/convolutional-neural-network-models-deep-learning-50-638.jpg?cb=1500039867>>. Acessado em: 2018. Citado na página 24.

SONG, C.; HUANG, Y.; WANG, Z.; WANG, L. 1000fps human segmentation with deep convolutional neural networks. In: IEEE. Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on. [S.l.], 2015. p. 474–478. Citado na página 29.

WU, Z.; LEAHY, R. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. IEEE transactions on pattern analysis and machine intelligence, IEEE, v. 15, n. 11, p. 1101–1113, 1993. Citado 2 vezes nas páginas 14 e 16.

YUAN, Y.; CHAO, M.; LO, Y.-C. Automatic skin lesion segmentation with fully convolutional-deconvolutional networks. arXiv preprint arXiv:1703.05165, 2017. Citado na página 16.