

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

EVERTON LEONARDO SKEIKA

**UTILIZAÇÃO DE REDES NEURAIS COMPLETAMENTE
CONVOLUCIONAIS PARA IDENTIFICAÇÃO E MEDição DE
CRÂNIOS FETAIS**

DISSERTAÇÃO

PONTA GROSSA
2019

EVERTON LEONARDO SKEIKA

**UTILIZAÇÃO DE REDES NEURAIS COMPLETAMENTE
CONVOLUCIONAIS PARA IDENTIFICAÇÃO E MEDIÇÃO DE
CRÂNIOS FETAIS**

Dissertação apresentada como requisito parcial à obtenção do título de Mestre em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação da Universidade Tecnológica Federal do Paraná – Campus Ponta Grossa.

Área de Concentração: Sistema e Métodos Computacionais

Orientador: Prof^a. Dr^a. Mauren Louise Sguario Coelho De Andrade

Co-orientador: Prof. Dr. Hugo Siqueira

PONTA GROSSA

2019

Ficha catalográfica elaborada pelo Departamento de Biblioteca
da Universidade Tecnológica Federal do Paraná, Campus Ponta Grossa
n.79/19

S627 Skeika, Everton Leonardo

Utilização de redes neurais completamente convolucionais para identificação
e medição de crânios fetais. / Everton Leonardo Skeika, 2019.

99 f.; il.; 30 cm.

Orientadora: Profª. Drª. Mauren Louise Sguario Coelho de Andrade
Co-orientador: Prof. Dr. Hugo Valadares Siqueira

Dissertação (Mestrado em Ciência da Computação) - Programa de Pós-
Graduação em Ciência da Computação. Universidade Tecnológica Federal do
Paraná, Ponta Grossa, 2019.

1. Ultrassonografia. 2. Craniometria. 3. Aprendizagem. 4. Redes neurais
(Computação). I. Andrade, Mauren Louise Sguario Coelho de. II. Siqueira, Hugo
Valadares. III. Universidade Tecnológica Federal do Paraná. III. Título.

CDD 004

	Ministério da Educação UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ CÂMPUS PONTA GROSSA Diretoria de Pesquisa e Pós-Graduação Programa de Pós-Graduação em Ciência da	 PPGCC <small>Programa de Pós-Graduação em Ciência da Computação</small>
---	--	--

FOLHA DE APROVAÇÃO

Título de Dissertação Nº 17/2019

UTILIZAÇÃO DE REDES NEURAIS COMPLETAMENTE CONVOLUCIONAIS PARA IDENTIFICAÇÃO E MEDIDA DE CRÂNIOS FETAIS

por

Everton Leonardo Skeika

Esta dissertação foi apresentada às **14 horas** do dia **19 de novembro de 2019**, na sala da **DIRPPG**, como requisito parcial para a obtenção do título de **MESTRE EM CIÊNCIA DA COMPUTAÇÃO**, Programa de Pós-Graduação em Ciência da Computação. O candidato foi arguido pela Banca Examinadora, composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho **APROVADO**.

**Prof^a. Dr^a. Alaine Margarete Guimarães
(UEPG)**

**Prof. Dr. Erickson Freitas de Moraes
(UTFPR)**

**Prof^a. Dr^a. Simone Bello Kaminski Aires
(UTFPR)**

**Prof^a. Dr^a. Mauren Sguario Coelho de Andrade (UTFPR)
*Orientadora e presidente da banca***



Visto do Coordenador:

**Prof. Dr. André Koscienski
Coordenadora do PPGCC
UTFPR – Câmpus Ponta Grossa**

- A FOLHA DE APROVAÇÃO ASSINADA ENCONTRA-SE ARQUIVADA NA SECRETARIA ACADÉMICA -

*Dedico este trabalho à memória de minha
avó, Mirian Monteiro.*

AGRADECIMENTOS

Certamente estes parágrafos não irão atender a todas as pessoas que fizeram parte dessa importante e triste fase da minha vida. Portanto, desde já peço desculpas àquelas que não estão presentes entre essas palavras, mas elas podem estar certas que fazem parte do meu pensamento e de minha gratidão.

Primeiramente, agradeço ao meu pai Miguel Leonardo e minha mãe Ana Cristina, por terem-me suportado em casa por tanto tempo.

Agradeço aos meus amigos Fagler, Fernanda, Jhonison, Leonira, Mariana, Silvio e Walyson, pelo apoio psicológico, material, pelos conselhos, pela compreensão, pela correção, pelos *games*, memes e churrascos aos finais de semana.

Agradeço aos sociais do PPGCC: Aleffer, Bauke, Eduardo, Fábia e Lin pelos estudos em conjunto, trabalhos em grupo e pelas festividades aos finais de semestre.

Agradeço ao meu Coo-orientador Prof. Dr. Hugo Siqueira pela sua ajuda na correção desta dissertação e na estruturação do artigo.

Por fim, agradeço a minha Orientadora, Prof^a Dr^a Mauren Louise Sguario por ter acreditado em mim desde o meu Trabalho de Conclusão de Curso (TCC), e pela enorme contribuição para o desenvolvimento desta dissertação.

*“Arriscar-se é perder o equilíbrio por uns tempos...
mas não se arriscar é perder-se a si mesmo para sempre.”*

Søren Kierkegaard

RESUMO

SKEIKA, Everton Leonardo. **Utilização de redes neurais completamente convolucionais para identificação e medição de crânios fetais.** 2019. 99 f. Dissertação (Mestrado em Ciência da Computação) - Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2019.

A ultrassonografia é uma técnica de diagnóstico por imagem não invasiva e não radioativa frequentemente utilizada no acompanhamento do desenvolvimento fetal durante a gestação. A avaliação precisa da biometria fetal é importante para a análise do crescimento do feto, diagnóstico de malformações e de possíveis doenças congênitas, garantindo assim o bem-estar da mãe e do feto durante a gestação. Porém, para uma medição precisa de estruturas anatômicas do feto é necessário conhecimento especializado do médico obstetra. Além de ser um processo tedioso e demorado, a extração do contorno é influenciada pela sua experiência. Dada esta problemática, em meados de 2018 foi lançado o desafio HC18 com objetivo de desafiar pesquisadores da área a projetar um algoritmo que possa medir automaticamente a circunferência da cabeça do feto mediante imagens de ultrassonografia bidimensionais. Neste sentido, este trabalho propôs uma solução ao desafio consistindo de uma adaptação da rede Neural Completamente Convolucional V-Net. Para atender o objetivo foi construída uma metodologia consistindo de algumas etapas como, por exemplo, a realização de pré-processamento no dataset disponibilizado pelo desafio, alterações na estrutura da rede para uma melhor segmentação das imagens, a construção de um algoritmo para pós-processamento das segmentações inefficientes e a implementação de um algoritmo para os cálculos das elipses resultantes da segmentação. Posteriormente, os resultados obtidos pela metodologia proposta foram submetidos ao desafio. Conforme a classificação obtida, demonstra-se que é possível a utilização da rede V-Net para este tipo de problema.

Palavras-chave: Ultrassonografia. Crânios fetais. Segmentação. Aprendizado profundo. Rede neural completamente convolucional.

ABSTRACT

SKEIKA, Everton Leonardo. **Use of Fully Convolutional Neural Network for identification and measurement of fetal skulls.** 2019. 99 p. Thesis (Master's in Computer Science) - Federal University of Technology – Paraná, Ponta Grossa, 2019.

Ultrasonography is a non-invasive, non-radioactive diagnostic imaging technique often used to monitor fetal development during pregnancy. An accurate assessment is important for the analysis of fetal growth, malformations diagnosis and possible congenital diseases, thus ensuring the well-being of the mother and fetus during pregnancy. However, accurate measurement of fetal anatomical structures requires specialized knowledge of the obstetrician. In addition to being a tedious and time-consuming process, contour extraction is influenced by his experience. Given this issue in mid-2018, the HC18 challenge was launched to challenge researchers in the field to develop an algorithm that can automatically measure head circumference using two-dimensional ultrasound images. In this sense, this work proposes a solution to the challenge consisting of an adaptation of the Fully Convolutional Neural Network V-Net. To serve this objective, a methodology was built consisting of some steps, such as performing the pre-processing in the dataset provided by the challenge, changes in the network structure for better segmentation of images, the construction of an algorithm for post-processing of inefficient segmentation, the implementation of an algorithm for calculating the ellipses resulting from segmentation. Subsequently, the results obtained by the proposed methodology were submitted to the challenge. According to the classification obtained, it is shown that it is possible to use the V-Net network for this type of problem.

Keywords: Ultrasonography. Fetal head. Segmentation. Deep learning. Fully convolutional neural network.

LISTA DE FIGURAS

Figura 1 - Aparelho de ultrassom	20
Figura 2 – Ultrassonografia em <i>B-mode</i> de uma gestante	21
Figura 3 – Representação de uma imagem digital em matriz de <i>pixel</i>	22
Figura 4 - Efeito do número de níveis de cinza. (a) 255 níveis; (b) 64 níveis; (c) 8 níveis e (d) 2 níveis.	23
Figura 5 – Aplicação de segmentação em uma imagem em escala de cinza	25
Figura 6 - Ilustração da arquitetura de uma LeNet que classifica imagens de entrada em célula anormal ou célula normal e suas três principais camadas: convolucional, de <i>pooling</i> e totalmente conectada.....	27
Figura 7 – Representação de uma operação de convolução utilizando um filtro de convolução Sobel GX.....	28
Figura 8 – Resultado da aplicação do Filtro de convolução Sobel Gx em uma imagem em escala de cinza	29
Figura 9 – Aplicação da função de ativação ReLu em um mapa de característica (4x4)	30
Figura 10 - Aplicação de <i>max pooling</i> utilizando <i>kernel</i> 2x2 e <i>stride</i> de 2	31
Figura 11 – Representação da função <i>Softmax</i>	32
Figura 12 - Aplicação de <i>data augmentation</i> em uma imagem colorida	33
Figura 13 – Representação da aplicação de (a) <i>Max pooling</i> e (b) <i>Max pooling dropout</i>	34
Figura 14 – Representação esquemática da arquitetura da rede FCN proposta por Long, Shelhamer e Darrell (2015)	36
Figura 15 – Imagem ilustrativa da etapa de <i>downsampling</i> com a aplicação de <i>maxpooling</i> (a) e da etapa de <i>upsampling</i> com a aplicação de <i>unpooling</i> (c) utilizando o <i>max location</i> (b).....	37
Figura 16 – Representação esquemática da arquitetura da rede U-Net proposta por Ronneberger, Fischer e Brox (2015)	39
Figura 17 - Representação esquemática da arquitetura da rede V-Net proposta por Milletari, Navab e Ahmadi (2016)	40
Figura 18 – (a) operação de convolução com <i>kernel</i> 2x2x2 e <i>stride</i> de 2, (b) operação de de-convolução com <i>kernel</i> 2x2x2 e <i>stride</i> de 2	42
Figura 19 - Diagrama de Venn baseado na comparação da segmentação automática e a <i>ground truth</i>	43
Figura 20 - Metodologia proposta.....	54
Figura 21 - Imagem representativa do desafio HC18	56
Figura 22 - Representação das imagens de ultrassonografia e as <i>ground truth</i> contidas no <i>dataset</i>	57

Figura 23 - Representação do código utilizado para o preenchimento das elipses...	61
Figura 24 - Resultado do preenchimento das <i>ground truth</i>	62
Figura 25- Resultados da segmentação obtidos pela rede V-Net 2D nos diferentes tempos de treinamento.....	65
Figura 26- Exemplo de aplicação de <i>data augmentation</i> no <i>dataset</i> utilizado (a) Imagem original. (b) Imagens aumentadas	72
Figura 27 - Algoritmo na linguagem Python para a realização do pós-processamento nas segmentações imprecisas	77
Figura 28- Figura ilustrativa dos resultados obtidos após o algoritmo de pós-processamento.....	78
Figura 29 - Ilustração das medidas necessárias	80
Figura 30 – Algoritmo na linguagem Python para o cálculo dos valores da elipse....	81
Figura 31 - Gráfico de evolução do valor médio de acurácia de acordo com as alterações impostas na arquitetura da rede	85
Figura 32 - Imagem comparativa entre alguns dos resultados obtidos pela rede U-Net, V-Net utilizada como base e a V-Net adaptada	87

LISTA DE TABELAS

Tabela 1 - Exemplo de tabela para a submissão dos resultados	57
Tabela 2 - Configurações específicas da placa de vídeo GTX TITAN	60
Tabela 3 – Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação ao número de épocas e de passos utilizados para o treinamento da rede V-Net.....	63
Tabela 4 - Tabela comparativa dos resultados das métricas de desempenho em relação ao número de épocas e de passos utilizados para o treinamento da rede V-Net.....	64
Tabela 5 - Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação à aplicação da técnica <i>batch normalization</i>	66
Tabela 6 - Tabela comparativa dos resultados das métricas de desempenho em relação ao uso da técnica <i>batch normalization</i>	66
Tabela 7 - Tabela comparativa dos resultados das métricas de desempenho em relação à reestruturação do <i>batch normalization</i>	67
Tabela 8 - Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação a alteração da função de ativação da última camada da rede.....	67
Tabela 9 - Tabela comparativa dos resultados das métricas de desempenho em relação ao tipo de função de ativação aplicada na última camada da rede	68
Tabela 10 - Tabela comparativa dos resultados obtidos para cada função de ativação	69
Tabela 11 - Tabela comparativa dos resultados das métricas de desempenho em relação as Funções de Ativação aplicadas	69
Tabela 12 - Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação aos valores de <i>dropout</i>	70
Tabela 13- Tabela comparativa dos resultados das métricas de desempenho em relação aos valores de <i>dropout</i>	70
Tabela 14 - Tabela demonstrativa das operações e valores utilizados para a geração de imagens sintéticas pela técnica de <i>data augmentation</i>	72
Tabela 15 - Tabela comparativa dos resultados obtidos com as edições no <i>data augmentation</i>	73
Tabela 16- Tabela comparativa dos resultados das métricas de desempenho em relação às edições no tamanho do <i>batch</i>	73
Tabela 17- Tabela comparativa dos resultados das métricas de desempenho em relação ao uso da função de Perda.....	74
Tabela 18 - Tabela comparativa dos resultados obtidos com as edições no número de estágios da rede.....	75
Tabela 19 - Tabela comparativa dos resultados das métricas de desempenho em relação aos números de estágios da rede	75

Tabela 20- Tabela comparativa dos resultados entre as segmentações (spp - sem pós-processamento) e (cpp - com pós-processamento)	79
Tabela 21- Tabela comparativa dos resultados gerais entre as segmentações (spp - sem pós-processamento) e (cpp - com pós-processamento)	79
Tabela 22- Tabela comparativa entre o 1º colocado no desafio com os resultados obtidos pela V-Net adaptada e a V-Net base	84
Tabela 23- Tabela comparativa da Acurácia, Perda e o Tempo de execução entre a rede U-Net base, V-Net base e a V-Net adaptada	86
Tabela 24- Tabela comparativa dos resultados das métricas de desempenho entre a rede U-Net base, V-Net base e a V-Net adaptada	86

LISTA DE SIGLAS

BN	<i>Batch Normalization</i>
CNN	<i>Convolutional Neural Network</i>
CSV	<i>Comma-Separated Values</i>
DCNN	<i>Deep Convolutional Neural Network</i>
DSC	<i>Dice Similarity Coefficient</i>
ELU	<i>Exponential Linear Units</i>
FCL	<i>Fully Connected Layer</i>
FCN	<i>Fully Convolutional Neural Network</i>
FN	Falso Negativo
FP	Falso Positivo
GPU	<i>Graphics Processing Unit</i>
JSC	<i>Jaccard Similarity Coefficient</i>
HC	<i>Head Circumference</i>
HD	<i>Hausdorff Distance</i>
PDI	Processamento Digital de Imagens
PNG	<i>Portable Network Graphic</i>
PReLU	<i>Parametric ReLu</i>
RNA	Redes Neurais Artificiais
ReLU	<i>Rectified Linear Units</i>
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo

SUMÁRIO

1 INTRODUÇÃO	15
1.1 OBJETIVOS.....	17
1.1.1 Objetivos Específicos	17
1.2 JUSTIFICATIVA	17
1.3 ESTRUTURA DO TRABALHO	18
2 REFERENCIAL TEÓRICO.....	19
2.1 O ULTRASSOM	19
2.2 PROCESSAMENTO DE IMAGENS	21
2.3 SEGMENTAÇÃO DE IMAGENS	23
2.4 <i>DEEP LEARNING</i>	25
2.4.1 Rede Neural Convolucional.....	26
2.4.1.1 A camada convolucional.....	27
2.4.1.2 A camada de subamostragem.....	30
2.4.1.3 A camada totalmente conectada	31
2.4.2 Técnicas para Otimização de CNN	32
2.4.2.1 <i>Data augmentation</i>	33
2.4.2.2 <i>Dropout</i>	34
2.4.2.3 <i>Batch Normalization</i>	35
2.4.3 Rede Completamente Convolucional	35
2.4.3.1 Rede completamente convolucional U-Net	38
2.4.3.2 Rede completamente convolucional V-Net.....	40
2.5 MÉTRICAS DE DESEMPENHO	42
2.5.1 Taxas de Sucesso	44
2.5.2 Métricas de Similaridade	45
2.5.3 Métrica de Distância.....	46
2.5.3.1 Distância de Hausdorff	46
2.6 CONSIDERAÇÕES FINAIS	47
3 TRABALHOS RELACIONADOS	48
3.1 CONSIDERAÇÕES FINAIS	53
4 METODOLOGIA	54
4.1 METODOLOGIA PROPOSTA.....	54
4.1.1 Estudos sobre o Desafio HC18	55
4.1.1.1 O desafio HC18	55

4.1.1.2 O <i>dataset</i>	56
4.1.1.3 Processo para a submissão e avaliação	57
4.1.2 Configuração do Ambiente de Trabalho	58
4.1.2.1Configurações de máquina.....	60
4.1.3 Pré-Processamento do <i>Dataset</i>	61
4.1.4 Adaptação da Rede V-Net.....	62
4.1.4.1 Uso de <i>batch normalization</i>	65
4.1.4.2 Alteração na camada de ativação	67
4.1.4.3 Alteração da função de ativação	68
4.1.4.4 Aplicação de <i>dropout</i>	69
4.1.4.5 Alteração no <i>data augmentation</i>	71
4.1.4.6 Alteração na função de perda	74
4.1.4.7 Alteração na profundidade da rede	74
4.1.5 Aplicação de Pós-Processamento.....	76
4.1.6 Implementação do Algoritmo para Cálculo do HC.....	80
4.1.7 Submissão ao desafio HC18	82
4.2 CONSIDERAÇÕES FINAIS	82
5 RESULTADOS E DISCUSSÕES.....	84
5.1 CONSIDERAÇÕES FINAIS	88
6 CONCLUSÃO	89
6.1 TRABALHOS FUTUROS	90
REFERÊNCIAS.....	91

1 INTRODUÇÃO

A ultrassonografia é um método de diagnóstico de patologias que utiliza ondas ultrassônicas para geração de imagens em tempo real. Devido a sua natureza não invasiva e não radioativa, é a modalidade de escolha em muitas aplicações clínicas devido ao seu custo reduzido em comparação com outras modalidades de captura de imagem, como a Tomografia Computadorizada ou a Ressonância Magnética (RUEDA et al., 2014; WU et al., 2017).

A ultrassonografia é uma técnica utilizada como auxiliar no diagnóstico médico em diversas áreas da medicina, tais como: obstetrícia, ginecologia, oftalmologia, neurologia e cardiologia, além de sua utilização como ferramenta comum em procedimentos terapêuticos (AL-KARMI et al., 1994).

Na obstetrícia, a ultrassonografia é amplamente utilizada para a avaliação do desenvolvimento fetal durante a gravidez, em que, ao utilizar a imagem gerada pelo equipamento de ultrassom, o especialista obtém as medidas do comprimento da cabeça do feto, as medidas relativas ao corpo, bem como a análise de seus movimentos a fim de identificar e prevenir o surgimento de doenças congênitas (ZAYED et al., 2001).

Algumas das principais medidas para avaliação do desenvolvimento fetal são: circunferência da cabeça (*head circumference - HC*), diâmetro biparietal (*biparietal diameter*), circunferência do abdômen (*abdominal circumference*), largura do fêmur (*femur length*), largura do humerus (*humerus length*), entre outras (CARNEIRO et al., 2008). Os cálculos destes parâmetros são gerados por meio de uma função matemática específica, auxiliando na estimativa da idade gestacional e peso do feto analisado (SANDERS e JAMES, 1985).

Na prática, a delimitação da área a ser mensurada é feita manualmente por um médico. Este procedimento requer conhecimento especializado, sendo um processo maçante e que consome tempo (JARDIM e FIGUEIREDO, 2005). Além disso, a extração do contorno dos ossos e órgãos em formação é influenciada pela experiência do avaliador. Para facilitar este processo e auxiliar nas análises dos resultados, técnicas de segmentação de imagens e medição automáticas são necessárias (LU; TAN; FLOYD, 2005).

No entanto, a segmentação automática de fetos em imagens de ultrassom continua a ser uma tarefa desafiadora. Dentre os desafios pode-se citar o fato da imagem gerada pelo ultrassom apresentar várias distribuições de intensidade devido a diferentes condições de aquisição. Além disso, uma série de ruídos, como sombras acústicas, ruído *speckle* e baixo contraste podem dificultar o reconhecimento das bordas presentes na imagem (LU; TAN; FLOYD, 2005; WU et al., 2017).

Dada a importância do problema, recentemente foi construído um desafio público chamado HC18¹ cujo objetivo é desafiar pesquisadores da área a projetar um algoritmo que possa medir automaticamente a circunferência da cabeça do feto mediante imagens de ultrassonografia bidimensionais (2D). O desafio possibilita a submissão de resultados, tornando possível a comparação entre os resultados obtidos por outros pesquisadores da área.

Dentre as diversas técnicas de segmentação de imagens existentes na literatura, uma nova abordagem baseada em Redes Neurais Artificiais vem obtendo grande eficácia, sendo denominada Rede Neural Convolucional (*Convolutional Neural Network - CNN*) (GIRSHICK et al., 2014). A CNN é um tipo de rede baseada na estrutura do córtex visual de mamíferos e tornou-se popular e bem-sucedida em diversas tarefas, tais como: reconhecimento visual e detecção de objetos por meio da rede *Faster-RCNN* (REN et al., 2015), classificação de imagens pela rede GoogLeNet (SZEGEDY et al., 2015), segmentação de objetos em imagens por meio da Rede Completamente Convolucional (*Fully Convolutional Network - FCN*) (LONG; SHELHAMER; DARRELL, 2015), entre outras.

Nesse sentido, este trabalho propõe a segmentação automática de crânios fetais em imagens de ultrassonografia bidimensionais (2D), utilizando a rede Completamente Convolucional V-Net² com o objetivo de reconhecer e medir a circunferência do crânio fetal. Pretende-se gerar uma padronização das medidas e possivelmente melhorar a confiabilidade da idade gestacional do feto. Além disso, este trabalho propõe uma nova solução para o desafio HC18, a fim de comparar os resultados com os demais trabalhos já submetidos.

¹ <https://hc18.grand-challenge.org/>

² <https://arxiv.org/abs/1606.04797>

1.1 OBJETIVOS

O objetivo geral deste trabalho é a segmentação de crânios fetais em imagens de ultrassonografia bidimensionais por meio da Rede Completamente Convolucional V-Net.

1.1.1 Objetivos Específicos

Para atender o objetivo deste trabalho, foram propostos os seguintes objetivos específicos:

- Investigar as principais técnicas de *Deep Learning* para segmentação de imagens médicas;
- Pesquisar técnicas de pré-processamento que auxiliem na precisão da segmentação por Redes Neurais Completamente Convolucionais;
- Modelar a rede V-Net para que seja capaz de segmentar com eficiência o crânio fetal;
- Projetar um algoritmo para o cálculo preciso da circunferência do crânio fetal;
- Submeter os resultados obtidos pela rede V-Net ao desafio HC18;
- Produzir uma análise comparativa dos resultados obtidos pela rede V-Net com a rede Neural Completamente Convolucional U-Net³, bem como uma comparação com os resultados já submetidos ao desafio HC18.

1.2 JUSTIFICATIVA

A ultrassonografia é uma ferramenta importante para análise gestacional e o diagnóstico precoce de doenças congênitas. Mensurar o crânio fetal e estimar a idade gestacional permite ao especialista monitorar o desenvolvimento saudável do feto. No entanto, a mensuração é feita manualmente pelo médico exigindo tempo e conhecimento específico. Trabalhos utilizando FCNs têm obtido resultados expressivos na resolução de diferentes tipos de problemas na área de processamento de imagens médicas. Neste contexto, a aplicação de FCN poderá segmentar

³ <https://pdfs.semanticscholar.org/0704/5f87709d0b7b998794e9fa912c0aba912281.pdf>

automaticamente o crânio fetal, objetivando uma padronização na medição automática do mesmo.

1.3 ESTRUTURA DO TRABALHO

Este trabalho é constituído de seis capítulos. O Capítulo 2 aborda o referencial teórico necessário para o desenvolvimento do trabalho. Os temas abordados são: ultrassom, processamento de imagens, segmentação de imagens, *deep learning*, rede neural convolucional, rede neural completamente convolucional, rede neural completamente convolucional U-Net, rede neural completamente convolucional V-Net e métricas de desempenho. O Capítulo 3 apresenta alguns trabalhos recentes relacionados ao tema proposto. O Capítulo 4 apresenta a metodologia utilizada no desenvolvimento do presente trabalho. Os temas abordados são: estudos sobre o desafio HC18, configuração do ambiente de trabalho, pré-processamento do *dataset*, adaptação da rede V-Net, aplicação de pós-processamento, implementação do algoritmo para o cálculo do HC e submissão ao desafio HC18. O Capítulo 5 apresenta os resultados e discussões sobre a metodologia proposta. Finalmente, o Capítulo 6 descreve as conclusões e trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo descreve os principais conceitos teóricos que fundamentam o desenvolvimento deste trabalho. Na Seção 2.1 são apresentados os conceitos sobre ultrassom. Seção 2.2 apresenta os conceitos fundamentais sobre processamento de imagens. A Seção 2.3 descreve os conceitos fundamentais sobre segmentação de imagens. Já na Seção 2.4 são descritos os conceitos fundamentais sobre *deep learning*, redes neurais convolucionais e redes neurais completamente convolucionais. Finalmente a Seção 2.5 apresenta algumas das métricas de desempenho, sendo: taxas de sucesso, métricas de similaridade e métricas de distância.

2.1 O ULTRASSOM

Também chamado de ultrassonografia ou ecografia, o ultrassom é um método de diagnóstico de diversas patologias que utiliza ondas ultrassônicas para geração de imagens em tempo real. Ao contrário de outros métodos de diagnóstico por imagem, como a Tomografia Computadorizada e a Ressonância Magnética, a ultrassonografia não utiliza radiação ionizante para a formação da imagem, não é um método invasivo ao paciente e seu custo é inferior (RUEDA et al., 2014; WU et al., 2017).

O diagnóstico por ultrassom é largamente utilizado em diferentes áreas médicas devido à sua simplicidade e seu baixo custo em relação a outros métodos de diagnósticos, além de não ser utilizada nenhuma forma de radiação ou de matéria contrastante para a formação de imagens (GUARIGLIA, 2016). Devido a estas características este exame é usado principalmente em crianças e gestantes, seja como parte do pré-natal ou no rastreamento de males em recém-nascidos.

O aparelho de ultrassom é uma máquina que utiliza ondas sonoras de alta frequência para produzir imagens de estruturas internas do corpo humano. Tais ondas são mecânicas e necessitam de um meio para se propagar como sólido, líquido ou gasoso.

O limite da audição humana varia entre 20Hz (*hertz*) e 20KHz (*kilohertz*). Frequências sonoras acima dos 20KHz são conhecidas como ultrassons e abaixo dos 20Hz são conhecidas como infrassons (KREMKAU, 1996). As frequências utilizadas

para aplicações médicas geralmente mantêm-se na gama dos 500KHz e 100MHz (*megahertz*) (ABUHAMAD et al., 2014).

A Figura 1 ilustra um aparelho de ultrassom em uso por um médico.

Figura 1 - Aparelho de ultrassom



Fonte: Leval (2006)

De forma resumida, o funcionamento de um aparelho de ultrassom (Figura 1) consiste na emissão de ondas ultrassônicas por meio de um equipamento chamado transdutor. O transdutor converte a energia elétrica em energia mecânica (ultrassom) e vice-versa, definindo a direção, frequência e geometria do feixe de som. Este aparelho normalmente é colocado na superfície do corpo do paciente para a emissão das ondas ultrassônicas. Estas, em contato com o órgão do paciente, serão refletidas e absorvidas pelo próprio equipamento (RAMOS, 2010). A onda retornada, também chamada de “eco”, é então quantificada em tons de cinza, em que os ecos com maiores intensidades de sinal serão representados por pontos brancos e os com menores intensidades por pontos pretos. Posteriormente esta informação será exibida no monitor em forma de imagem (MACIEL e PERREIRA, 1997).

A Figura 2 ilustra uma imagem gerada pelo aparelho de ultrassom.

Figura 2 – Ultrassonografia em *B-mode* de uma gestante



Fonte: Adaptado de Obgyn (2006)

Na Figura 2 pode-se observar na imagem bidimensional diferentes tons de cinza. Esta técnica de geração de imagem é chamada *B-mode* (*brightness modulation*) na qual a amplitude do eco é representada como diferentes pontos de brilho (HAUFF; REINHARDT; FOSTER, 2008).

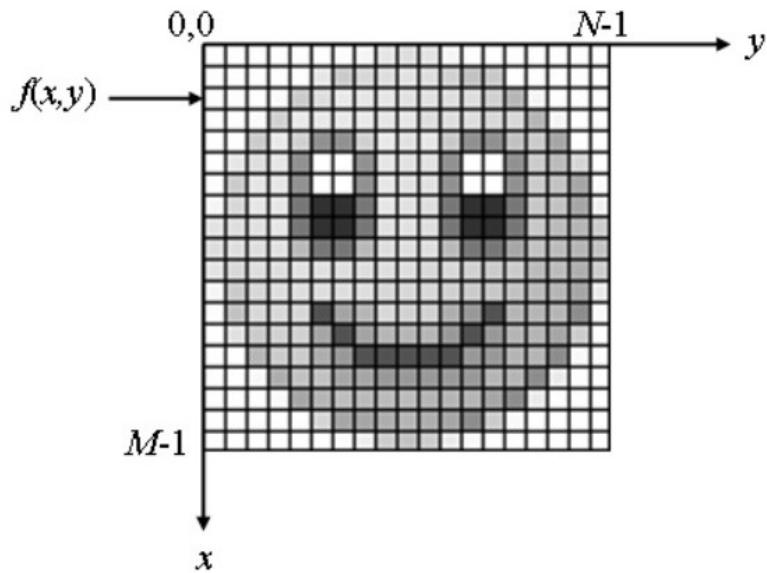
2.2 PROCESSAMENTO DE IMAGENS

Segundo Marques Filho e Vieira Neto (1999) o Processamento Digital de Imagens (PDI) viabiliza duas tarefas: a primeira é a melhora nas informações presentes em uma dada imagem digital para auxiliar a visão humana; a segunda é, dada uma cena, o computador analisá-la automaticamente.

Sanches (p. 48, 2009) explica que “uma imagem digital é composta de um número finito de elementos denominados *pixels* (*picture elements* ou elementos de imagem) representados na forma de uma matriz bidimensional $M \times N$, onde M representa o número de linhas e N o número de colunas.”

Matematicamente uma imagem monocromática pode ser definida como uma função bidimensional, sendo $f(x, y)$, em que x e y são coordenadas espaciais e a amplitude f em qualquer par de coordenadas (x, y) é a intensidade ou nível de cinza da imagem naquele ponto (GONZALEZ e WOODS, 2008). A Figura 3 ilustra uma imagem digital em escala de cinza.

Figura 3 – Representação de uma imagem digital em matriz de *pixel*



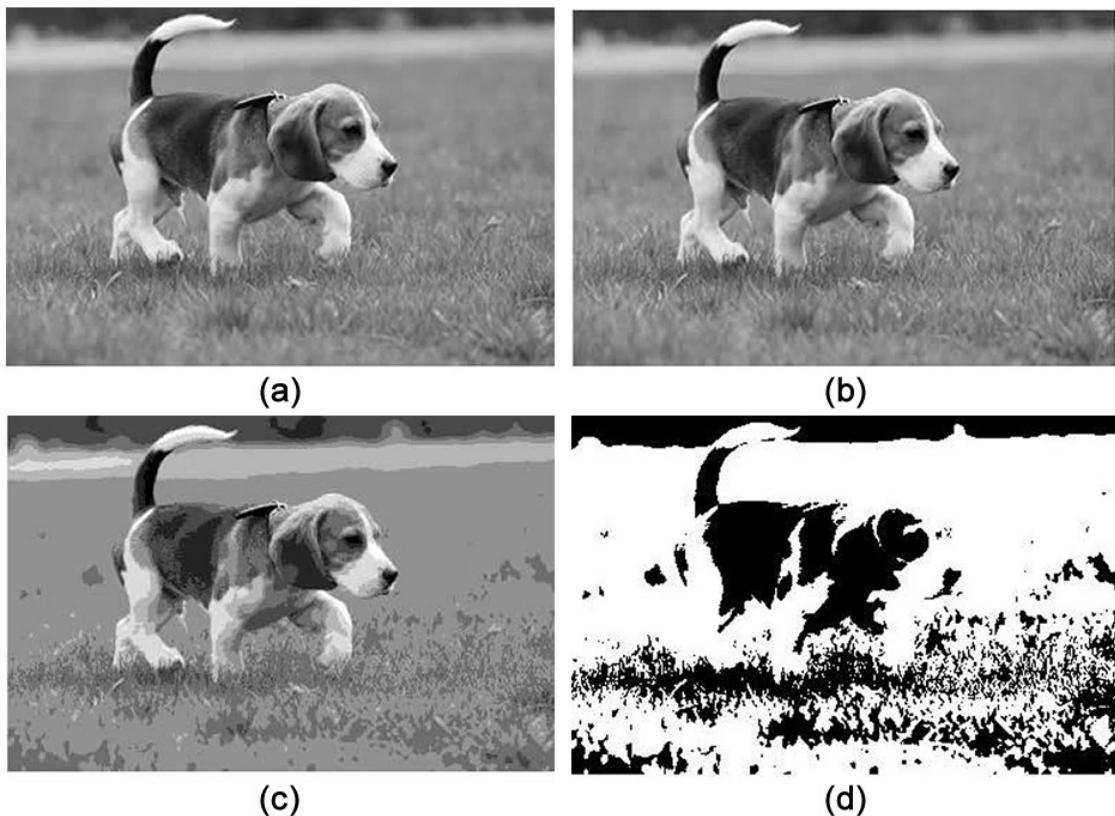
Fonte: Sanches (2009)

Uma imagem digital pode ser obtida por um processo denominado digitalização, que consiste em dois principais passos, a amostragem e a quantização. A amostragem é a resolução espacial da imagem, que está diretamente ligada à quantidade de linhas e colunas nas suas respectivas direções x e y , gerando uma matriz de $M \times N$ amostras. A quantização consiste em definir o número inteiro L de níveis de cinza permitidos para cada ponto da imagem (PEDRINI e SCHWARTZ, 2008).

O valor que cada *pixel* pode assumir é determinado pela quantização considerada para codificar os níveis de intensidade (GONZALEZ e WOODS, 2001).

A quantização em uma imagem digital em escala de cinza está expressa em *8 bits*, em que o valor de cada *pixel* será um valor inteiro na faixa de 0 à $2^n - 1$, em que o valor 0 caracteriza a cor preta, variando sua intensidade de cor até o valor máximo $2^8 - 1 = 255$, que representa a cor branca. Ou seja, quanto maior o valor de n , maior o número de níveis de cinza presentes na imagem (MARQUES FILHO e VIEIRA NETO, 1999). A Figura 4 ilustra uma mesma imagem representada em diferentes níveis de quantização.

Figura 4 - Efeito do número de níveis de cinza. (a) 255 níveis; (b) 64 níveis; (c) 8 níveis e (d) 2 níveis.



Fonte: Autoria própria

Nota-se na Figura 4 que as diferenças são maiores na Figura 4 (c) com 8 níveis de cinza e na Figura 4 (d) com apenas 2 (imagem binária), do que na Figura 4 (a) com 255 e a Figura 4 (b) com 64. Isso ocorre devido ao fato de o olho humano não possuir sensibilidade às mudanças de intensidade acima de 30 níveis de cinza (CRÓSTA, 1993).

2.3 SEGMENTAÇÃO DE IMAGENS

A segmentação de imagens subdivide uma imagem em regiões ou objetos que a compõem (GONZALEZ; WOODS, 2010). Sendo fundamentada em duas características dos tons de cinza de uma imagem: a descontinuidade e a similaridade. A descontinuidade consiste em particionar a imagem baseando-se em mudanças bruscas nos níveis de cinza. A similaridade fundamenta-se pela agregação de *pixels* em função da sua semelhança com seus *pixels* vizinhos.

Segundo (GONZALEZ; WOODS, 2001), a segmentação é um processo que partitiona uma região espacial (imagem) \mathbb{R} em n sub-regiões, $\mathbb{R}_1, \mathbb{R}_2, \dots, \mathbb{R}_n$ de tal forma que:

a) $\bigcup_{i=1}^n \mathbb{R}_i = \mathbb{R}$

- Indica que cada *pixel* deve estar em uma região;

b) \mathbb{R}_i é um conjunto conectado, $i = 1, 2, \dots, n$

- Requer que os *pixels* de uma região estejam conectados;

c) $\mathbb{R}_i \cap \mathbb{R}_j = \emptyset$, para todo $i \neq j$, onde $i \neq j$

- Indica que as regiões devem ser disjuntas, ou seja, cada *pixel* pertence apenas a uma região;

d) $P(\mathbb{R}_i) = VERDADEIRA$, para $i = 1, 2, \dots, n$

- Propriedade em que os *pixels* de uma região devem todos ter a mesma intensidade;

e) $P(\mathbb{R}_i \cup \mathbb{R}_j) = FALSA$, para qualquer região adjacente \mathbb{R}_i e \mathbb{R}_j

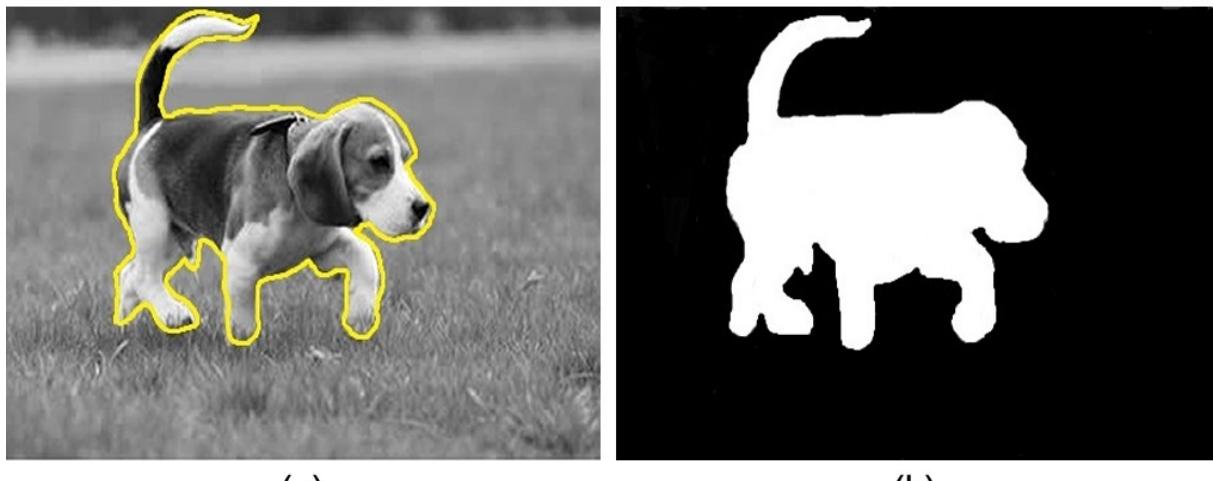
- Indica que as regiões adjacentes devem ser diferentes na propriedade (d).

A tarefa de segmentar uma imagem é complexa porque tenta traduzir para o computador um processo cognitivo realizado pela visão humana (BEUCHER e MEYER, 1982). Para o computador é difícil tratar o processo de segmentação automática, dado que as bordas das regiões a serem segmentadas normalmente não são muito nítidas, sendo muitas vezes irregulares e imprecisas, além de não existir um modelo padrão para a segmentação de imagens. Para tanto, diversos algoritmos podem ser empregados dependendo da complexidade do objeto alvo a ser segmentado, dentre alguns podemos citar: *Thresholding* (CHOW e KANEKO, 1972), *Split and Merge* (HARALICK e SHAPIRO, 1985), *Graph Cut* (BOYKOV e JOLLY, 2001), *Active Shape Model* (GINNEKEN et al., 2002), entre outros.

De modo geral, a segmentação é um processo empírico e adaptativo, que procura sempre se adequar às características particulares de cada tipo de imagem e ao objetivo que se pretende alcançar (PINTRO, 2008).

A Figura 5 ilustra o processo de segmentação em uma imagem em escala de cinza.

Figura 5 – Aplicação de segmentação em uma imagem em escala de cinza



Fonte: Autoria própria

Na Figura 5 (b) observa-se uma imagem binária separando duas partes, sendo em branco o cachorro e em preto, o fundo (*background*) como resultado do processo de segmentação.

Na área médica, em geral, as técnicas de segmentação são utilizadas para detectar estruturas, tais como: órgãos, lesões, tumores, ossos e tecidos, extraíndo seus contornos de uma forma eficiente, robusta e automatizada (ARAUJO, 2012).

Neste trabalho a segmentação será útil para a detecção do crânio fetal em imagens de ultrassom 2D separando-o do fundo da imagem.

2.4 DEEP LEARNING

O Aprendizado Profundo (em inglês, *Deep Learning*) é uma subárea de Aprendizado de Máquina (em inglês, *Machine Learning*) relacionado a algoritmos inspirados na estrutura e função do cérebro humano, denominados Redes Neurais Artificiais.

O *Deep Learning* é atualmente um dos mais eficientes métodos em Aprendizado de Máquina para classificação de imagens. A técnica possibilita que o computador aprenda automaticamente atributos complexos e relevantes a partir de um conjunto de treinamento (BENGIO et al., 2009).

Segundo Lecun, Bengio e Hinton (2015), o *Deep Learning* possibilita que modelos computacionais compostos por múltiplas camadas de processamento possam aprender representações de dados com múltiplos níveis de abstração. Esses métodos melhoraram drasticamente o estado da arte em diversas áreas, como por exemplo: reconhecimento de imagens (KRIZHEVSKY; SUTSKEVER; HINTON, 2012, SZEGEDY et al., 2014) e reconhecimento de fala (HINTON et al., 2012). Também superou outras técnicas de Aprendizado de Máquina para prever a atividade de possíveis moléculas de drogas (MA et al., 2015), na reconstrução de circuitos cerebrais (HELMSTAEDTER et al., 2013), entre outros domínios.

As Redes Neurais Convolucionais foram as primeiras técnicas de *Deep Learning* bem-sucedidas, nas quais várias camadas de uma hierarquia foram treinadas de maneira consistente (AREL et al., 2010). Esse sucesso só foi possível devido a recentes superações em muitos problemas, como por exemplo: no aumento da quantidade de dados disponíveis em muitas áreas (imagens, áudios, textos, etc), que são essenciais para o treinamento deste tipo de rede. Outra razão para a grande adoção foi o avanço no poder computacional das Unidades de Processamento Gráfico (*Graphics Processing Unit – GPU*), possibilitando a análise de grandes quantidades de dados, realizando operações com maior velocidade e viabilizando o uso de paralelização (NASCIMENTO, 2016).

2.4.1 Rede Neural Convolucional

Criada por Lecun et al. (1998), uma Rede Neural Convolucional, também conhecida como ConvNet, é um tipo de modelo de *Deep Learning* amplamente utilizado para tarefas de classificação e aprendizado através de imagens.

As CNNs são da família das redes neurais multicamadas, particularmente projetadas para uso em dados de duas dimensões, como imagens e vídeos.

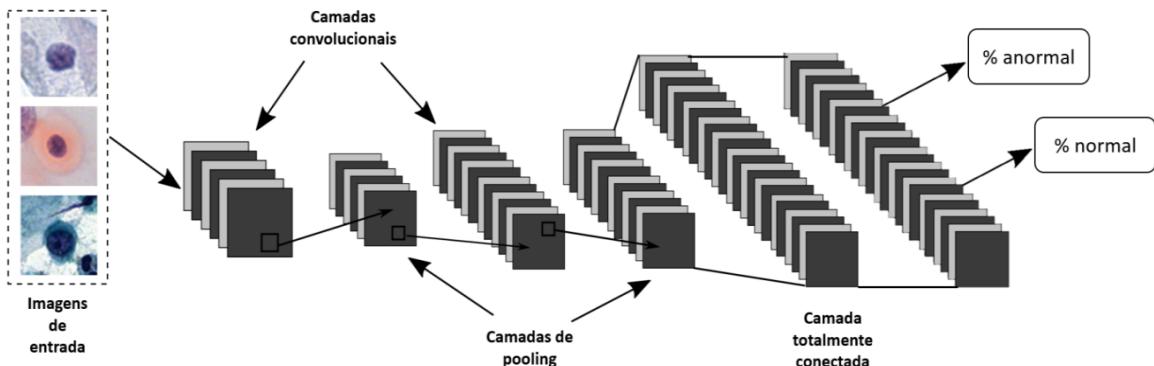
Normalmente a arquitetura de uma CNN consiste em diminuir a dimensão da imagem em cada camada da rede ao mesmo tempo em que as características não

conhecidas das camadas anteriores são extraídas. Então, a partir delas obtém-se a informação desejada para a classificação de objetos contidos na imagem.

As CNNs possuem diversos tipos de arquiteturas, seja para classificação como a *GoogLeNet* (SZEGEDY et al., 2015), para detecção como a *VGG Net* (SIMONYAN e ZISSERMAN, 2014), para reconhecimento como a *AlexNet* (Krizhevsky; Sutskever; Hinton, 2012) e para segmentação como a *V-Net* (Milletari; Navab; Ahmadi, 2016). Sendo esta última a arquitetura base para a concepção dos objetivos propostos neste trabalho.

A Figura 6 representa a arquitetura de uma rede CNN chamada *LeNet* proposta por Lecun et al. (1998), utilizada para a classificação de células, onde o resultado desta rede classificará a célula presente em uma imagem como anormal ou normal.

Figura 6 - Ilustração da arquitetura de uma LeNet que classifica imagens de entrada em célula anormal ou célula normal e suas três principais camadas: convolucional, de pooling e totalmente conectada



Fonte: Araújo et al. (2017)

A arquitetura básica de uma CNN é formada por três tipos de camadas, sendo: a camada de convolução, a de subamostragem (*pooling*) e a totalmente conectada (*fully connected layer*). Estas três camadas serão brevemente descritas a seguir.

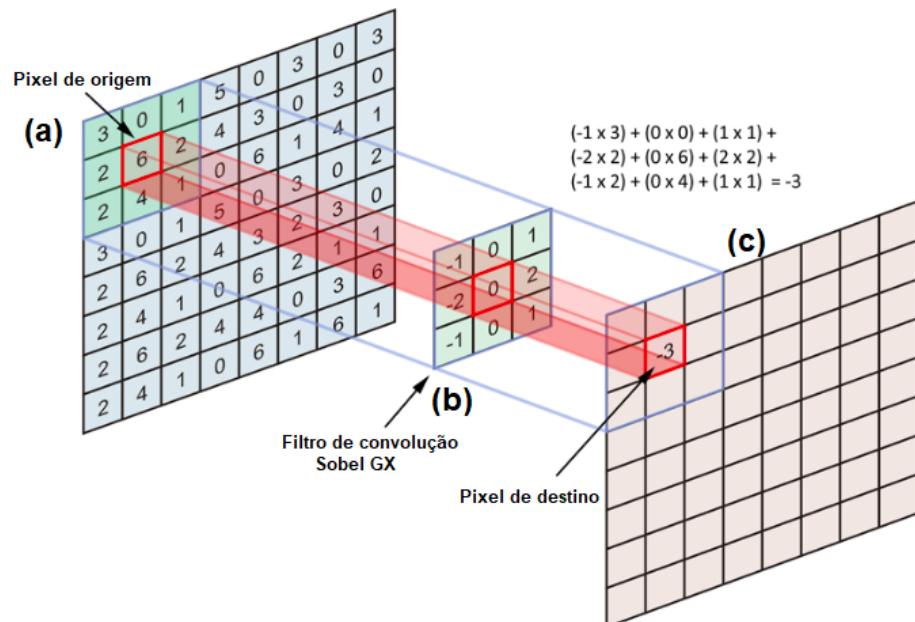
2.4.1.1 A camada convolucional

Essa camada envolve a maior parte do processamento computacional de uma CNN, sendo considerada a camada mais importante da rede. Sua principal função é extrair características pertinentes da entrada, podendo ser bordas, cores, texturas, entre outras (KARPATHY, 2017).

A camada convolucional possui filtros bidimensionais (*kernel*s) treináveis que são aplicados através de uma “janela deslizante” que percorre toda a imagem de entrada calculando o produto escalar entre os pesos do filtro e os *pixels* correspondentes da entrada. Esta operação é conhecida como convolução. A aplicação da convolução na imagem é dada por um número pré-definido de *pixels*, conhecido como passo (*stride*). Para cada filtro, um neurônio está conectado a apenas um subconjunto dos neurônios na camada anterior (LECUN; KAVUKCUOGLU; FARABET, 2010). Cada filtro é responsável por detectar um tipo de característica na imagem e cada filtro irá produzir um mapa de característica (*feature map*) distinto (HAFEMANN, 2014).

A Figura 7 ilustra o processo de convolução em uma imagem aplicando o filtro de Sobel proposto por Sobel e Feldman (1968), cujo objetivo é detectar bordas de objetos existentes numa determinada cena.

Figura 7 – Representação de uma operação de convolução utilizando um filtro de convolução Sobel GX

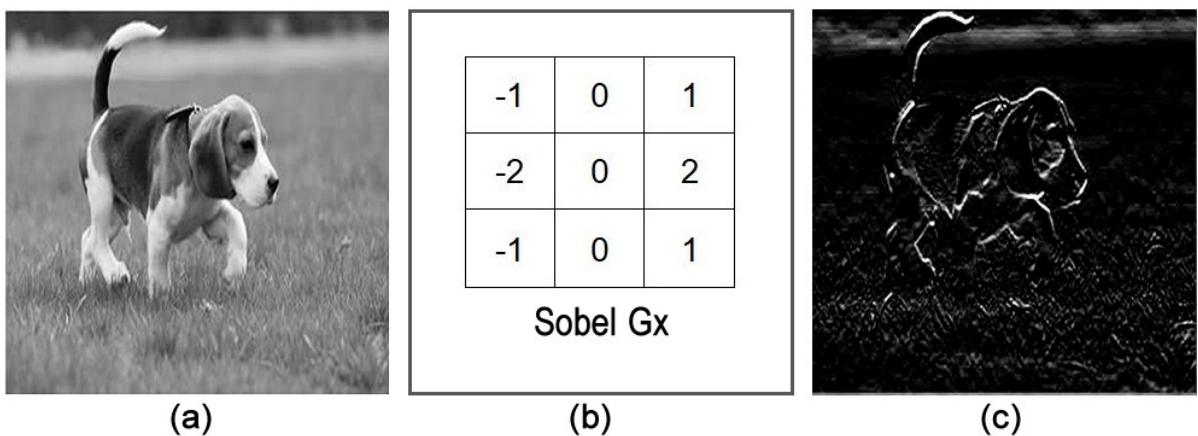


Fonte: Adaptado de Apple (2016)

A Figura 7 (a) representa uma imagem, em que o *kernel* irá percorrer toda a imagem aplicando o filtro de Sobel, Figura 7 (b). O valor resultante da multiplicação entre os valores do *pixel* com os do *kernel* será armazenado em uma matriz com a mesma dimensão da imagem original chamada de mapa de características, como ilustrado na Figura 7 (c). Esta será enviada às próximas camadas da rede neural convolucional.

A Figura 8 ilustra o resultado de uma convolução utilizando o filtro de Sobel em uma imagem em escala de cinza.

Figura 8 – Resultado da aplicação do Filtro de convolução Sobel Gx em uma imagem em escala de cinza



Fonte: Autoria própria

Conforme o resultado visto na Figura 8 (c), percebe-se que a aplicação do filtro Sobel Gx (b) extraí atributos significativos da imagem original (a), neste caso ressaltando as bordas existentes na direção vertical.

Frequentemente, após a operação de convolução, os valores resultantes passam por uma função de ativação. Uma função de ativação desempenha um papel importante na rede, decidindo quais *pixels* devem ser ativados em uma determinada camada.

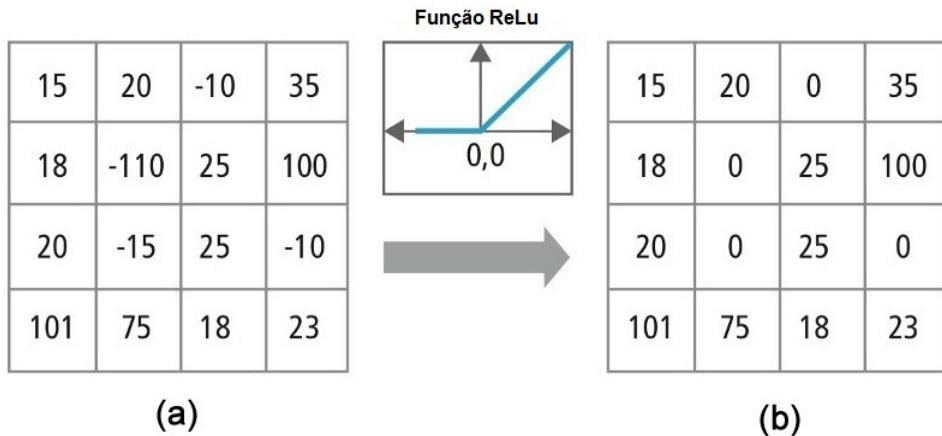
Dentre diversas funções de ativação a mais comumente utilizada no processo de convolução é a Unidade Linear Retificadora (do inglês *Rectified Linear Units* - ReLU) que foi introduzida por Krizhevsky, Sutskever e Hinton (2012). Matematicamente a função ReLU é dada pela Equação 1:

$$f(u) = \max(0, u) \quad (1)$$

A função retornará o valor 0 caso os valores sejam menores que ele e acima deste valor eles serão mantidos. Esta função é aplicada para cada *pixel* do mapa de características de forma a retificar os valores negativos.

A Figura 9 exemplifica a aplicação da função ReLU.

Figura 9 – Aplicação da função de ativação ReLu em um mapa de característica (4x4)



Fonte: Adaptado de Singhal (2017)

Conforme observado na Figura 9, todos os valores negativos existentes no mapa de característica (Figura 9 (a)) tornaram-se zero na (Figura 9 (b)).

Outras funções não lineares como *Tanh* ou *Sigmoid*, também podem ser usadas em vez de ReLU, porém na maioria dos casos a função ReLU obtém um desempenho superior (SINGHAL, 2017).

2.4.1.2 A camada de subamostragem

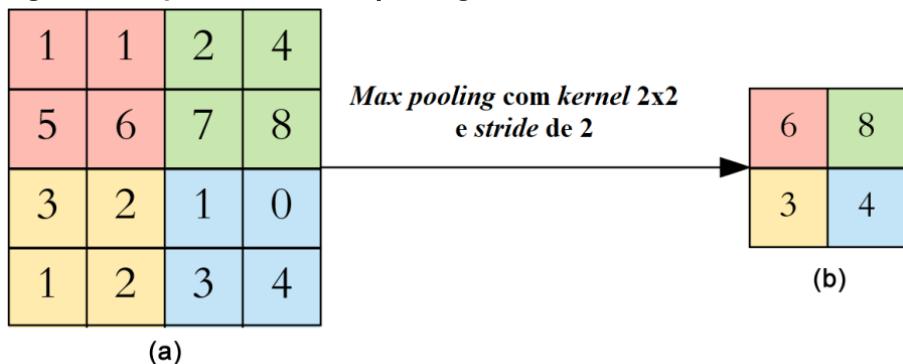
Uma CNN utiliza várias camadas de subamostragem (em inglês, *pooling*). As camadas de *pooling* são uma forma de *down-sampling*. *Down-sampling* é o processo de reduzir a taxa de amostragem de um sinal. Neste sentido, esta camada tem como objetivo reduzir progressivamente o tamanho espacial das matrizes resultantes da convolução. Com o *pooling* há uma redução no custo computacional da rede e também evita-se o *overfitting*⁴ (KARPATHY, 2017).

A técnica de *pooling* frequentemente utilizada nas redes CNN é o *max pooling*. Dada a escolha do *kernel* a ser utilizado, o *max pooling* consiste em substituir os valores da região tomada pelo *kernel* pelo seu valor máximo. Sua utilização é útil para eliminar valores desprezíveis e reduzir a dimensão da representação dos dados (GOODFELLOW et al., 2016).

⁴*overfitting* (em português, sobreajuste) é um termo estatístico utilizado para indicar que um modelo se adequa bem ao conjunto de dados previamente observado, mas que não é eficaz para prever resultados fora do seu conjunto de treinamento.

A Figura 10 exemplifica o resultado do *max pooling* em uma imagem (4×4) utilizando um *kernel* de tamanho (2×2) e *stride* de 2.

Figura 10 - Aplicação de *max pooling* utilizando *kernel* 2×2 e *stride* de 2



Fonte: Adaptado de Dertat (2017)

Conforme visto na Figura 10 (b), a aplicação do *max pooling* mantém o maior valor do *pixel* reduzindo o tamanho da imagem, mas mantendo informações importantes.

Outras funções de *pooling* comumente usadas são a média, a norma L2 e a média ponderada baseada na distância partindo do *pixel* central (AGGARWAL; HINNENBURG; KEIM, 2001).

2.4.1.3 A camada totalmente conectada

A camada totalmente conectada (*Fully Connected Layer* - FCL) é a última etapa de uma CNN, e nada mais é do que uma Rede Neural Artificial semelhante à Perceptron de Múltiplas Camadas (*MultiLayer Perceptron* - MLP). Seu objetivo é utilizar as características da imagem extraídas pelas camadas anteriores para classificá-la em uma classe pré-determinada. O significado do termo totalmente conectado se dá ao fato de que cada neurônio na camada anterior está ligado a cada neurônio na próxima camada, adicionando uma camada de saída com o número de neurônios em relação ao número de classes presente no respectivo experimento para realizar a classificação aplicando algum tipo de função de ativação (KARN, 2016).

A função de ativação mais utilizada nesta etapa é a *Softmax*. Esta função recebe um vetor de valores como entrada e produz a distribuição probabilística da imagem de entrada pertencer a cada uma das classes na qual a rede foi treinada (ARAÚJO et al., 2017). Matematicamente a função *Softmax* é dada pela Equação 2:

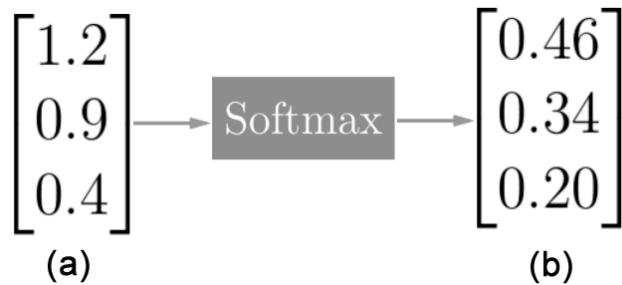
$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad (2)$$

em que z é o vetor de entrada para as camadas de saída. Por exemplo, se a rede tiver 10 unidades de saída então z terá 10 elementos, e j indexa as unidades de saída variando de $1, 2, \dots, K$.

A função *Softmax* faz uma restrição para que cada unidade fique com probabilidades entre 0 e 1 e também divide cada saída de forma que sua soma seja igual a 1.

A Figura 11 representa a aplicação da função *Softmax*.

Figura 11 – Representação da função Softmax



Fonte: SEWAK, KARIM e PUJARI (2018)

A saída da função *Softmax* é equivalente a uma distribuição de probabilidade categórica (Figura 11 (b)), ou seja, ela informa a probabilidade de cada uma das classes ser verdadeira (YANG, 2017).

2.4.2 Técnicas para Otimização de CNN

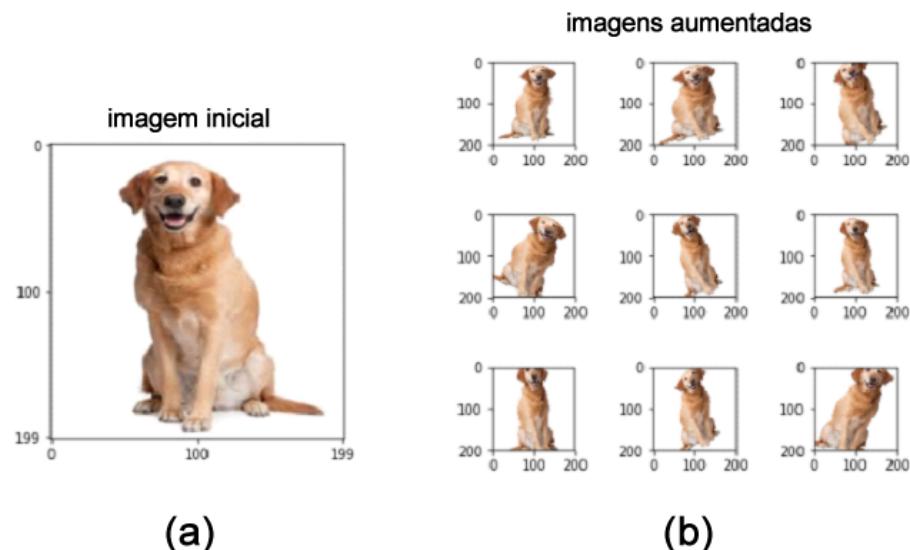
Nesta seção serão abordadas algumas das técnicas utilizadas para melhorar o desempenho durante o treinamento de uma CNN. Estas técnicas serão brevemente descritas a seguir.

2.4.2.1 Data augmentation

A técnica de aumento dos dados (em inglês, *data augmentation*) consiste em aumentar o *dataset*⁵ de treinamento gerando novas imagens artificiais a partir de transformações geométricas como rotação, translação, injeções de ruído, alteração na cor, brilho e contraste da imagem, recorte, entre outros, criando assim novas representações a partir das imagens originais existentes no *dataset* (LEE et al., 2017).

A Figura 12 ilustra a aplicação da técnica *data augmentation* em uma imagem colorida.

Figura 12 - Aplicação de *data augmentation* em uma imagem colorida



Fonte: Adaptado de Sudhakar (2018)

Conforme visto na Figura 12, em (a) representa-se uma imagem original contida no *dataset* e em (b) mostra-se o resultado da aplicação de *data augmentation* gerando nove novas representações a partir da imagem original.

Esta técnica é bastante utilizada quando se têm um *dataset* com poucas imagens, desta forma tornando-se eficiente contra o *overfitting* dos dados.

⁵ Em visão computacional, um *dataset* (em português, conjunto de dados) é um conjunto de imagens utilizado para treinamento e validação de algoritmos.

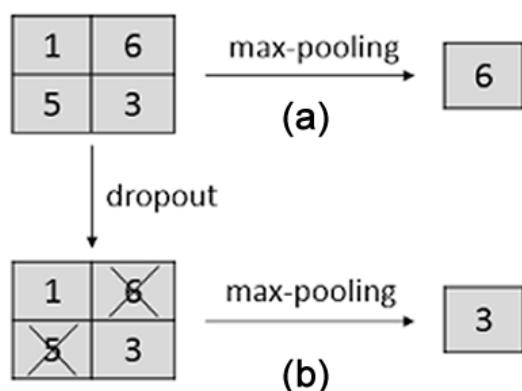
2.4.2.2 Dropout

A técnica de *dropout* deriva do termo em inglês “*dropping out*” (em português, abandonar), foi proposta por Hinton et al. (2012) com objetivo de reduzir o *overfitting* da rede durante o seu treinamento. O seu funcionamento consiste em desativar um conjunto de neurônios da camada totalmente conectada a cada iteração do treinamento da rede. A desativação é realizada de acordo com uma probabilidade p que varia no intervalo entre $[0, 1]$, sendo que quanto maior a taxa, maior a quantidade de neurônios que não são ajustados durante o treinamento da rede. Com menos ativações o problema de *overfitting* é reduzido, forçando cada camada da rede a se especializar em uma determinada característica de forma mais distinta. Em resumo, o *dropout* intenciona a rede a não ficar dependente de um, ou da combinação de vários neurônios durante a etapa de treinamento.

Esta técnica é comumente utilizada na camada totalmente conectada de uma CNN obtendo um bom desempenho. Entretanto, ela pode ser aplicada em outras camadas da rede como a camada de *pooling* e convolucional (WU e GU, 2015).

A Figura 13 ilustra a aplicação da técnica *dropout* numa etapa de *max pooling* previamente explicada na seção 2.4.1.2.

Figura 13 – Representação da aplicação de (a) Max pooling e (b) Max pooling dropout



Fonte: Adaptado de WU e GU (2015)

Conforme visto na Figura 13 (a), o resultado do *max pooling* resultará sempre na ativação do maior valor, neste caso sendo o valor 6. Na Figura 13 (b) com a desativação aleatória dada pela técnica de *dropout*, possibilita-se que outros valores sejam ativados.

2.4.2.3 Batch Normalization

A técnica conhecida como Normalização em Lote (em inglês, *Batch Normalization* - BN) foi desenvolvida por Ioffe e Szegedy (2015) para o tratamento de problemas de inicialização da rede. Um dos fatores que dificultam o treinamento é que os valores das funções de ativações das camadas anteriores estão sempre mudando. Neste sentido, a Normalização em Lote normaliza a saída de uma camada de ativação para seguir uma distribuição Gaussiana normal, e como consequência, acelerando o processo de aprendizado da rede.

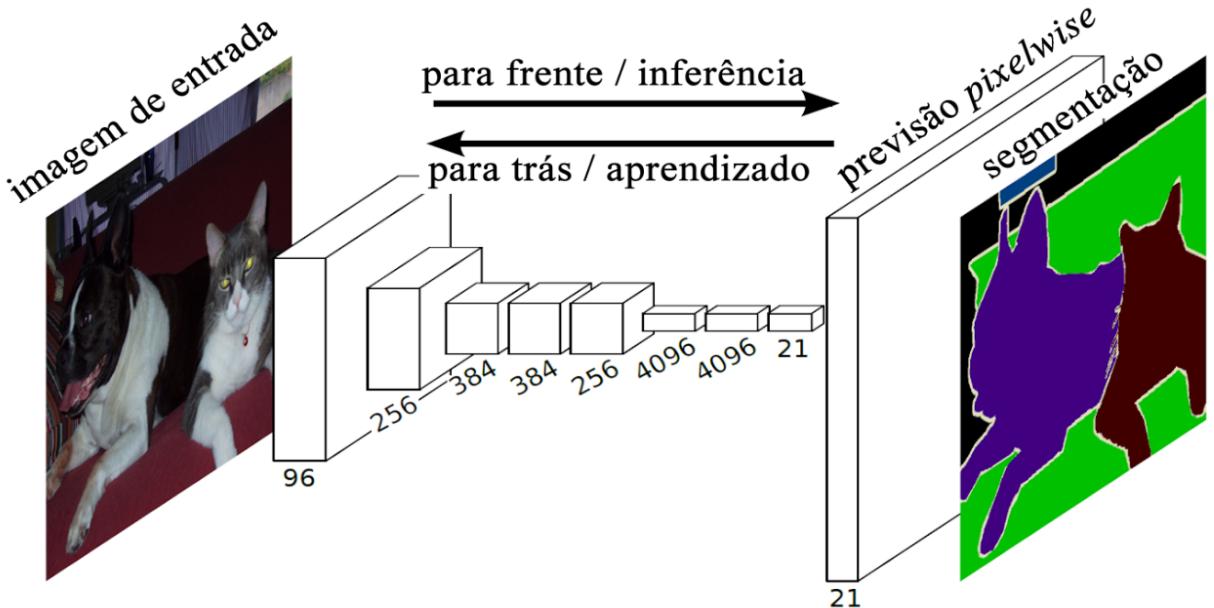
Em algumas circunstâncias a utilização da técnica de *batch normalization* despreza a necessidade da utilização da técnica de *dropout* nas redes convolucionais, diminuindo o tempo de treinamento da rede (LEE; GALLAGHER; TU, 2016).

2.4.3 Rede Completamente Convolucional

Criada por Long, Shelhamer e Darrell (2015), a Rede Completamente Convolucional é muito similar a uma rede CNN. Porém, na arquitetura da FCN a camada totalmente conectada, que é tipicamente utilizada para classificação, é substituída por outra camada de convolução com um grande “campo receptivo” utilizado para classificar cada *pixel* da imagem. A ideia desta abordagem é capturar o contexto global da cena, ou seja, quais são os objetos existentes na imagem e qual a sua localização.

A Figura 14 apresenta a arquitetura de uma rede FCN utilizada para tarefa de segmentação semântica (*semantic segmentation*), isto é, classificar cada *pixel* da imagem de entrada de acordo com a classe que ele pertence, sendo: gato, cachorro, sofá, janela ou fundo (*background*).

Figura 14 – Representação esquemática da arquitetura da rede FCN proposta por Long, Shelhamer e Darrell (2015)



Fonte: Adaptado de Long, Shelhamer e Darrell (2015)

Conforme a arquitetura apresentada na Figura 14, existem várias camadas de convolução que produzirão mapas de características de diferentes profundidades. No final da rede, encontra-se a previsão *pixelwise* (*pixelwise prediction*) que também é um tipo de camada de convolução e que irá fazer uma predição *pixel-a-pixel*, isto é, atribuindo cada *pixel* a uma respectiva classe. Conforme visto na Figura 14, o tamanho do *pixelwise prediction* é 21 devido à existência de 21 classes distintas no *dataset* utilizado pelos autores da rede.

A arquitetura de uma rede FCN pode ser dividida em duas partes principais, o caminho de *downsampling* e o caminho de *upsampling*, os quais serão brevemente descritos a seguir:

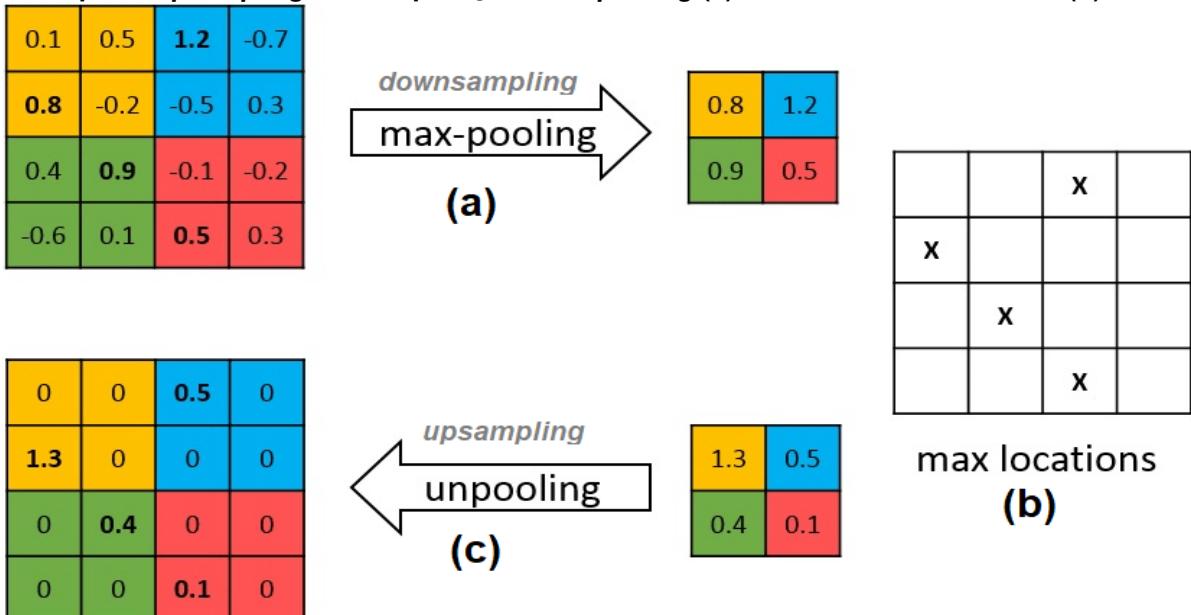
- **O caminho de *downsampling*:** reduz a dimensão da imagem utilizando etapas de *pooling*;
- **O caminho de *upsampling*:** é o oposto das camadas de *pooling*. Em sua forma mais simples ele restitui a imagem para a sua resolução original.

Uma técnica mais avançada de *upsampling* é o *unpooling* que consegue reverter a operação de *maxpooling* lembrando a localização dos máximos (*max location*) nas camadas *maxpooling* e copia este valor para o local exato na camada de *unpooling* (ZEILER; FERGUS, 2014).

Desta forma, a rede consegue reduzir a imagem utilizando camadas intermediárias e posteriormente reconstruí-la com interpolação e outras convoluções.

A Figura 15 representa a operação de *unpooling*.

Figura 15 – Imagem ilustrativa da etapa de *downsampling* com a aplicação de *maxpooling* (a) e da etapa de *upsampling* com a aplicação de *unpooling* (c) utilizando o *max location* (b)



Fonte: Adaptado de David e Netanyahu (2016)

Conforme descrito na Figura 15, durante a operação de *max-pooling* (a) a localização dos *max locations* (b) é armazenada de modo que, durante o *unpooling*, o valor aproximado seja restaurado para o local exato e os locais restantes sejam definidos como zero (DAVID e NETANYAHU, 2016).

Uma FCN é bastante utilizada para segmentação de imagens, devido ao fato que este tipo de arquitetura consegue segmentar a entrada da rede ao mesmo tempo em que a classifica, não sendo necessária outra técnica para seccionar a imagem.

Um dos problemas deste tipo de arquitetura de rede é que ao propagar por várias camadas convolucionais e camadas de *pooling* alternadas, a resolução de saída do mapa de característica é reduzida. Portanto, as estimativas obtidas pela rede geralmente estarão em baixa resolução, resultando em limites de objetos relativamente imprecisos (LE, 2017).

2.4.3.1 Rede completamente convolucional U-Net

Criada por Ronneberger, Fischer e Brox (2015), esta é uma rede totalmente convolucional sendo desenvolvida para a segmentação de imagens biomédicas. Os autores a projetaram para trabalhar com um número pequeno de imagens de treinamento e obter uma alta precisão na segmentação.

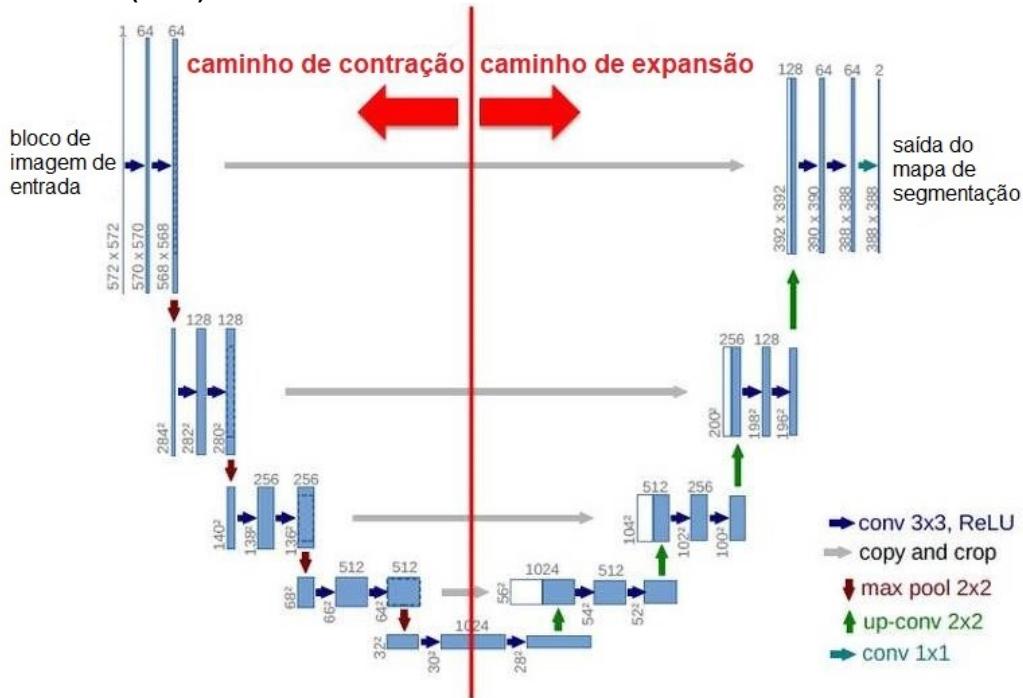
A U-Net é uma rede só com camadas de convolução e não têm a camada totalmente conectada. Este tipo de arquitetura conhecida como *Encoder-Decoder* é formado por camadas de contração, estruturadas ao lado esquerdo da rede, e ao lado direito é composta com camadas de expansão, apresentando uma simetria entre as camadas dos dois lados, no formato da letra "U", conforme pode ser observado na Figura 16 (ALMEIDA, 2017). Como a rede consegue trabalhar com poucas imagens de treinamento, os autores projetaram esta arquitetura para utilizar *data augmentation* (ver subseção 2.4.2.1), sendo especialmente útil, pois geralmente é difícil a obtenção de um grande *dataset* de imagens médicas.

Segundo Ronneberger, Fischer e Brox (2015) a principal estratégia que diferencia a U-Net das outras arquiteturas FCN é a combinação entre os mapas de características do estágio de contração e seus correspondentes simétricos no estágio de expansão, permitindo a propagação de informações de contexto para os mapas de características de alta resolução.

A Figura 16 ilustra a arquitetura da rede U-Net, em que cada caixinha azul presente na imagem corresponde a um mapa de característica multicanal (*multi-channel feature map*). O número de cada canal está descrito no valor acima de cada caixa. No canto inferior esquerdo é dada a dimensão $x-y$ da imagem. As caixas brancas representam a cópia dos mapas de características (*feature maps*) e cada flecha com sua respectiva cor representa uma operação diferente, conforme descrito no canto direito inferior da figura (RONNEBERGER; FISCHER; BROX, 2015). Na parte direita da rede as flechas verdes referem-se ao caminho de expansão onde é utilizado a operação de *up-convolution*, também chamada de *de-convolution*⁶ ou *transposed convolution*.

⁶ Diversos autores expressam que o nome “*de-convolution*” gera mal entendimento, pois o termo “*de-convolution*” matematicamente é definido como o inverso de uma convolução, o que é diferente do que esta operação faz (DUMOULIN e VISIN, 2016).

Figura 16 – Representação esquemática da arquitetura da rede U-Net proposta por Ronneberger, Fischer e Brox (2015)



Fonte: Adaptado de Ronneberger, Fischer e Brox (2015)

A operação de *up-convolution* é uma operação inversa que realiza uma convolução regular, mas reverte sua transformação espacial, dado que uma convolução normal, dependendo dos parâmetros utilizados como: *stride*, tamanho do *kernel* ou *padding*, acabam por reduzir a dimensão da imagem de entrada. Neste sentido, a operação *up-convolution* é utilizada para retornar a dimensão da imagem que ela tinha anteriormente, antes da operação de convolução.

Segundo Ronneberger, Fischer e Brox (2015), o caminho de contração (parte esquerda da rede) consiste na repetição de duas convoluções com *kernel* 3x3. Cada uma é seguida por uma unidade linear retificada (ReLU) e uma operação de *max pooling* com *kernel* 2x2 e *stride* de 2 para *downsampling*. Em cada etapa de *downsampling*, dobra-se o número de mapas de características. No lado direito da rede, cada passo do caminho expansivo consiste em um *upsampling* do mapa de características seguido por uma *up-convolution* com *kernel* 2x2 que dividirá a quantidade destes mapas pela metade. Uma concatenação é feita com o mapa de característica recortado do caminho de contração (dado pelas flechas na cor cinza, conforme ilustrado na Figura 16) e duas convoluções, com *kernel* 3x3 cada uma, seguidas por uma ReLU. Conforme descrito pelos autores, este recorte é necessário devido à perda de *pixels* de borda em toda convolução. Na camada final, uma

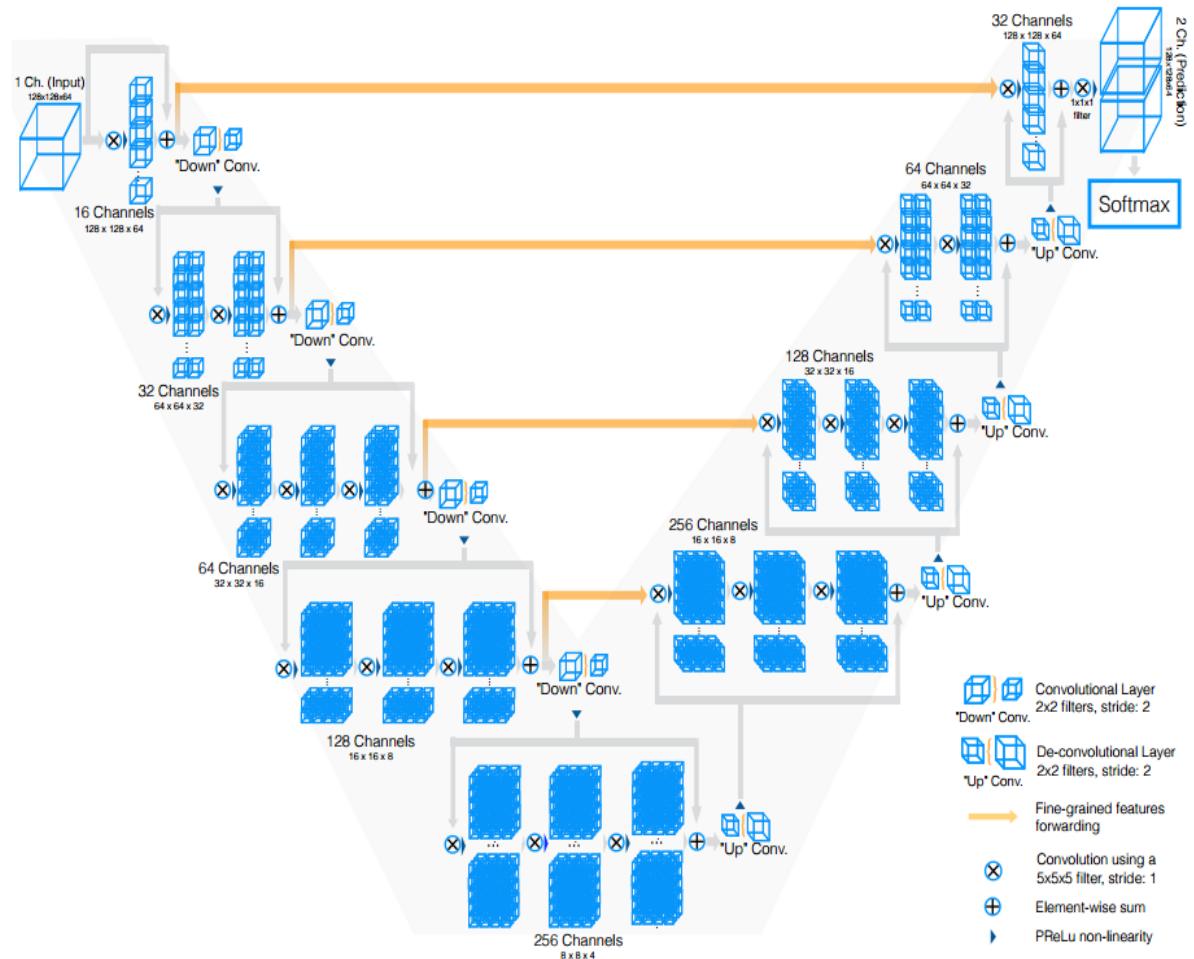
convolução com *kernel* 1x1 com uma função de ativação *Softmax* é usada para realizar a classificação *pixel-a-pixel* da imagem de saída.

2.4.3.2 Rede completamente convolucional V-Net

Criada por Milletari, Navab e Ahmadi (2016) esta rede possui algumas similaridades com a arquitetura da rede Completamente Convolucional U-Net. Os autores projetaram esta rede para a segmentação de imagens de ressonância magnética da próstata tridimensionais (3D), baseando-se em uma rede neural totalmente convolucional volumétrica (*volumetric fully convolutional neural network*).

A arquitetura desta rede realiza convoluções com o objetivo de extrair características dos dados e, assim como a U-Net, esta rede pode ser descrita em duas partes, a esquerda (parte de compressão) e a direita (parte de descompressão).

Figura 17 - Representação esquemática da arquitetura da rede V-Net proposta por Milletari, Navab e Ahmadi (2016)



Fonte: Milletari, Navab e Ahmadi (2016)

A parte esquerda da rede consiste em um caminho de compressão. Este lado é dividido em diferentes estágios que operam em diferentes resoluções, estes estágios possuem de uma a três camadas convolucionais. Conforme observado na Figura 17, as convoluções em cada estágio usam *kernels* volumétricos com $5 \times 5 \times 5$ *voxels*. À medida que os dados prosseguem por diferentes estágios ao longo do caminho de compactação, sua resolução é reduzida. Isso é realizado por convoluções com $2 \times 2 \times 2$ *voxels* com *kernel* largo e utilizando um *stride* de 2.

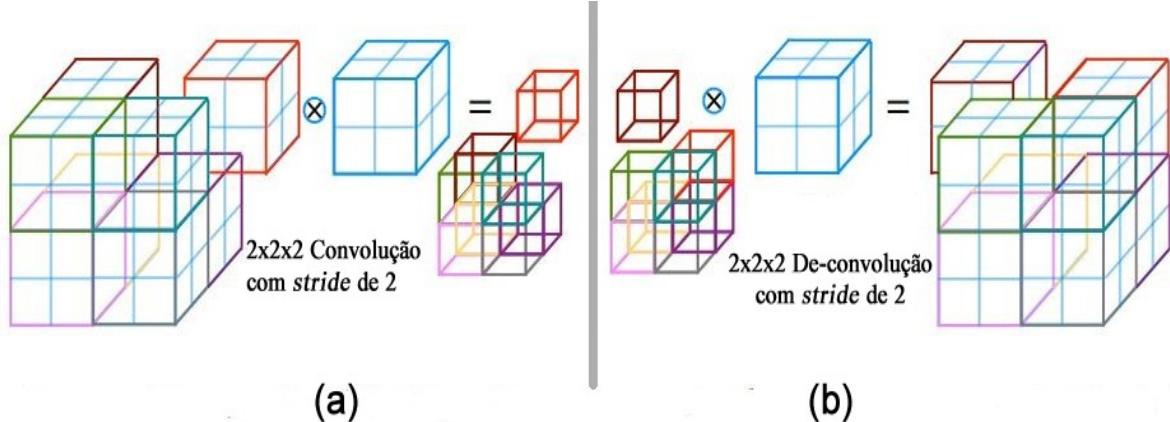
Conforme visto na Figura 18 (a), a cada operação o tamanho dos mapas de características são reduzidos pela metade, esta estratégia tem propósito similar às camadas de *pooling*, as quais não são aplicadas nesta rede. Os autores também aplicaram a função de ativação PReLU (*Parametric ReLU*) proposta por He et al. (2015), em toda a rede. A PReLU é uma derivação da função ReLU (ver seção 2.4.1.1) que aprende os parâmetros de forma adaptativa obtendo uma precisão superior com um custo computacional equivalente (HE et al., 2015).

O *downsampling* realizado pela rede permite reduzir o tamanho do sinal apresentado como entrada e aumentar o campo receptivo das características que estão sendo computadas nas camadas de rede. Cada um dos estágios presente na parte esquerda da rede calcula um número de características que é duas vezes maior que o da camada anterior (MILLETARI; NAVAB; AHMADI, 2016).

Já o lado direito da rede extrai as características e expande os mapas de características (*feature maps*) com baixa resolução, com o objetivo de juntar e montar as informações necessárias para realizar a segmentação. Os dois mapas de características computados pela última camada convolucional terão o tamanho de *kernel* $1 \times 1 \times 1$ e produzirão saídas do mesmo tamanho que o volume de entrada. Estas que serão processadas através de uma camada *Softmax* que gerará a probabilidade de cada *voxel* pertencer ao *background* ou ao *foreground* (MILLETARI; NAVAB; AHMADI, 2016).

Depois de cada etapa do lado direito da rede, uma operação de *de-convolução* é utilizada para aumentar o tamanho da saída conforme ilustrado na Figura 18 (b) seguidos de uma a três camadas convolucionais envolvendo metade do número de *kernels* $5 \times 5 \times 5$ resultantes da camada anterior (MILLETARI; NAVAB; AHMADI, 2016).

Figura 18 – (a) operação de convolução com kernel 2x2x2 e stride de 2, (b) operação de de-convolução com kernel 2x2x2 e stride de 2



Fonte: Adaptado de Milletari, Navab e Ahmadi (2016)

Similarmente a estratégia descrita em Springenberg et al. (2014), os autores encaminharam as características dos estágios iniciais da parte esquerda da CNN para a parte direita, conforme representado esquematicamente na Figura 17 pelas setas horizontais nas cores laranja. Estas conexões juntam detalhes refinados (*fine-grained*) que seriam perdidos no caminho de compressão, resultando em um aprimoramento da qualidade final da segmentação e melhorando o tempo de convergência do modelo (MILLETARI; NAVAB; AHMADI, 2016).

2.5 MÉTRICAS DE DESEMPENHO

Na literatura existem diversas métricas para a avaliação de segmentação de imagens. As mais comuns incluem as taxas de sucesso, as métricas de similaridade e as métricas de distância (ROMAGUERA et al., 2017).

Para obter a taxa de sucesso das redes descritas neste trabalho foi necessário medir a qualidade da segmentação gerada após sua execução, comparando-as com as imagens de *ground truth*. A *ground truth* será considerada a imagem de referência que representa a segmentação ideal. Neste trabalho a segmentação ideal foi concebida manualmente por um ultrassonografista treinado.

As métricas quantitativas utilizadas são baseadas no conceito de verdadeiro/falso e positivo/negativo conhecido como Matriz de Confusão, como mostrado na Tabela 1, em que cada *pixel* da imagem segmentada é comparado com o respectivo *pixel* presente na imagem de *ground truth* e, então, é calculada a porcentagem de erros e acertos da imagem segmentada.

Tabela 1 – Matriz de Confusão

	POSITIVO	NEGATIVO
POSITIVO	Verdadeiro Positivo (VP)	Falso Negativo (FN)
NEGATIVO	Falso Positivo (FP)	Verdadeiro Negativo (VN)

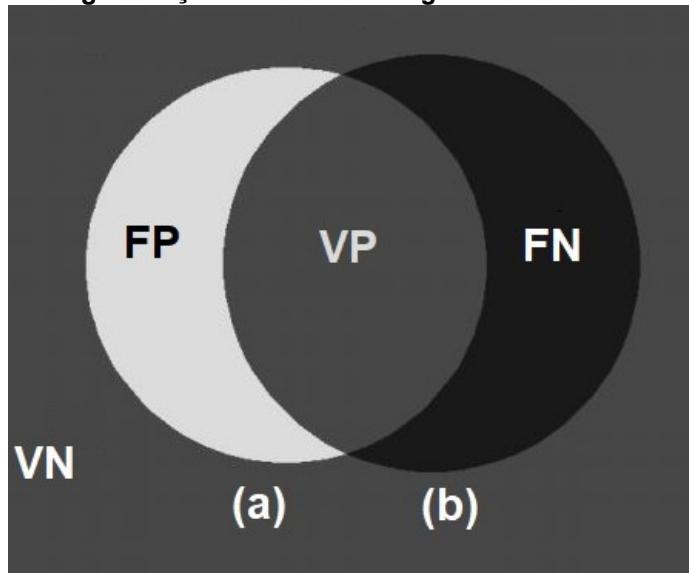
Fonte: Adaptado de Sguario (2015)

Em um problema de segmentação de imagens quatro situações podem ocorrer, sendo (SGUARIO, 2015):

- **Verdadeiro Positivo (VP):** representa o total de *pixels* corretamente segmentados, onde quanto maior o valor desta taxa melhor será o desempenho do algoritmo;
- **Falso Positivo (FP):** representa um *pixel* erroneamente classificado como pertencente ao objeto alvo, mas na realidade não pertencente;
- **Verdadeiro Negativo (VN):** o *pixel* é predito corretamente como não pertencente ao alvo;
- **Falso Negativo (FN):** mede a taxa de *pixels* pertencentes ao alvo que tenham sido classificados como não pertencentes pelo algoritmo de segmentação.

A Figura 19 ilustra as definições de VP, FP, VN e FN em um diagrama de Venn.

Figura 19 - Diagrama de Venn baseado na comparação da segmentação automática e a *ground truth*



Fonte: Adaptado de Safarzadeh (2013)

Onde na Figura 19 (a) representa a imagem gerada pela segmentação automática e (b) representa a imagem de *ground truth*.

2.5.1 Taxas de Sucesso

As taxas de sucesso baseiam-se pelos resultados gerados a partir de uma matriz de confusão. Algumas destas métricas são a sensibilidade (*sensitivity*) também conhecida como *recall*, a especificidade (*specificity*), a precisão (*precision*) e a acurácia (*accuracy*), as quais serão brevemente descritas a seguir:

- **Sensibilidade:** A sensibilidade mede o total de *pixels* positivos no *ground truth* que também são identificados como positivos pela segmentação que está sendo avaliada (ROMAGUERA et al., 2017). A sensibilidade é dada pela Equação 3:

$$\text{sensibilidade} = \frac{|VP|}{|VP| + |FN|} \quad (3)$$

- **Especificidade:** A especificidade mede o total de *pixels* negativos no *ground truth* que também são considerados como negativos pela segmentação (ROMAGUERA et al., 2017). A especificidade é dada pela Equação 4:

$$\text{especificidade} = \frac{|VN|}{|VN| + |FP|} \quad (4)$$

- **Precisão:** A precisão mede o total de *pixels* positivos no *ground truth* que também são considerados como positivos pela segmentação, dividida pelo número total de *pixels* identificados como pertencentes ao conjunto positivo (a soma de verdadeiros positivos e falsos positivos, que são os *pixels* incorretamente rotulados como pertencente ao conjunto) (POWERS, 2011). A precisão é dada pela Equação 5:

$$\text{precisão} = \frac{|VP|}{|VP| + |FP|} \quad (5)$$

- **Acurácia:** A acurácia é relativa aos acertos na classificação. É a soma de verdadeiros positivos e verdadeiros negativos divididos pelo todo. Essa métrica mostra como o classificador se saiu de uma maneira geral (POWERS, 2011). A acurácia é dada pela Equação 6:

$$\text{acurácia} = \frac{|VP| + |VN|}{|VP| + |VN| + |FP| + |FN|} \quad (6)$$

2.5.2 Métricas de Similaridade

As métricas de similaridade medem quão bem duas segmentações se sobrepõem. Duas medidas de similaridade comumente usadas são os Coeficientes de Similaridade Dice e o de Jaccard, os quais serão descritos a seguir:

- **Coeficiente de Similaridade Dice:** O Coeficiente de Similaridade Dice (*Dice Similarity Coefficient - DSC*) (DICE, 1945) é definido como a intersecção entre duas regiões sendo G e A , em que, $|G|$ é a *ground truth* e $|A|$ é a imagem segmentada pelo algoritmo. O simbolo \cap representa a intersecção destas duas máscaras binárias (conjuntos), dividido pelo volume médio dessas duas regiões (SAFARZADEH, 2013). O coeficiente de similaridade Dice é dado pela Equação 7:

$$DSC = \frac{2|G \cap A|}{|G| + |A|} = \frac{2(VP)}{(VP + FP) + (VP + FN)} \quad (7)$$

- **Coeficiente de Similaridade Jaccard:** O Coeficiente de Similaridade Jaccard (*Jaccard Similarity Coefficient – JSC*) é definido como a intersecção entre duas regiões sendo G e A onde, G é a *ground truth* e A é a imagem segmentada pelo algoritmo, o simbolo \cap representa a intersecção destas duas máscaras binárias, sendo dividida pela sua união, dado pelo símbolo \cup (SAFARZADEH, 2013). O coeficiente de similaridade Jaccard é dado pela Equação 8:

$$JSC = \frac{|G \cap A|}{|G \cup A|} = \frac{VP}{VP + FP + FN} \quad (8)$$

O resultado de ambas as métricas variam de 0 a 1, em que 0 indica que os conjuntos não têm regiões comuns e 1 indica que os conjuntos são equivalentes (SHATTUCK et al., 2001).

2.5.3 Métrica de Distância

As métricas de distância medem o quanto distante dois subconjuntos do espaço estão um do outro. Dentre as diversas métricas de distância existentes na literatura, neste trabalho será utilizada apenas a Distância de Hausdorff, a qual será descrita a seguir.

2.5.3.1 Distância de Hausdorff

A distância de Hausdorff (*Hausdorff Distance* - HD) proposta por Huttenlocher, Klanderman e Ruckridge (1993) é um método matemático que mede o quanto distante dois subconjuntos de um espaço métrico estão um do outro (SANTOS, 2015).

Matematicamente, a distância de Hausdorff pode ser demonstrada como: Seja a e b quaisquer pontos de conjuntos distintos $A = a_1, a_2, \dots, a_n$ e $B = b_1, b_2, \dots, b_n$, a distância entre os pontos do conjunto será dada pela distância euclidiana, sendo:

$$d(a, b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2} = \| a - b \| \quad (9)$$

A distância de Hausdorff sendo definida como:

$$H(A, B) = \max[h(A, B), h(B, A)] \quad (10)$$

em que:

$$h(A, B) = \max_{a \in A} \min_{b \in B} \| a - b \|; \quad (11)$$

$$h(B, A) = \max_{b \in B} \min_{a \in A} \| b - a \|. \quad (12)$$

A distância direta de Hausdorff dada pela função de $h(A, B)$ de A para B , identifica o ponto de $a \in A$ de maior distância ao ponto mais próximo do conjunto B . Intuitivamente, se $h(A, B) = d$, qualquer ponto a possui distância menor ou igual a d , no que tange à mínima distância aos pontos do conjunto B (HUTTENLOCHER; KLANDERMAN; RUCKLIDGE, 1993).

2.6 CONSIDERAÇÕES FINAIS

Ao longo deste capítulo, apresentaram-se todos os conceitos teóricos considerados importantes para o desenvolvimento deste trabalho. Entendê-los é de suma importância para a compreensão da metodologia que foi utilizada. Nele, foram introduzidos os conceitos básicos sobre o aparelho de ultrassom e a geração da imagem de ultrassonografia. Posteriormente foram explicados os conceitos fundamentais da área de processamento e segmentação de imagens. Seguidamente foram introduzidos os conceitos necessários sobre *deep learning*, em que foram abordados os conceitos introdutórios sobre as redes neurais convolucionais, algumas das principais técnicas para a sua otimização. Também foram abordados os conceitos sobre redes neurais completamente convolucionais, além de um detalhamento da rede que foi utilizada como base para o presente trabalho. Na última seção foram abordadas algumas métricas de desempenho que foram necessárias para avaliação da segmentação. No próximo capítulo são apresentados alguns dos trabalhos recentes relacionados com o tema proposto.

3 TRABALHOS RELACIONADOS

Devido à relevância do problema, recentemente alguns pesquisadores têm se dedicado ao desenvolvimento de novas abordagens utilizando Aprendizado Profundo para a segmentação de órgãos e ossos fetais em imagens de ultrassonografia bidimensionais. Neste contexto, o presente capítulo apresenta alguns dos recentes trabalhos relacionados.

Li et al. (2017) propuseram a utilização de uma rede FCN com uma estrutura do tipo *Encoder-Decoder* para a segmentação automática do líquido amniótico e tecidos fetais em imagens de ultrassom 2D em pontos de vista distintos e posições arbitrárias. Resumidamente, a estrutura *Encoder-Decoder* da rede pode ser dividida em duas etapas: a primeira etapa é a camada de *Encoder* que consiste no processo da redução da entrada da rede, ou seja, a cada etapa da rede a dimensão espacial da imagem de entrada vai diminuindo. Neste passo os autores utilizaram as primeiras 10 camadas de convolução da rede VGG16 proposta por Simonyan e Zisserman (2014) como a base da estrutura. A segunda etapa é a camada de *Decoder* da rede, que consiste em reconstruir (decodificar) a entrada diminuída para sua dimensão original. Para a avaliação automática do tamanho e volume do corpo fetal e do líquido amniótico, três classes foram definidas: corpo fetal, líquido amniótico e fundo, as quais foram anotadas manualmente. Estas segmentações manuais foram realizadas por médicos especialistas com vasta experiência em imagens de ultrassonografia. Os resultados das segmentações geradas pela rede foram comparados com as anotações manuais obtendo a taxa de acerto de 93% global, 67% na precisão da classe especificada como corpo fetal e 78% na precisão da classe especificada como líquido amniótico.

O trabalho de Sobhaninia et al. (2019) propôs uma rede multitarefa (*Multi-Task network*) profunda baseada na estrutura da rede Link-Net proposta por Chaurasia e Culurciello (2017) para a segmentação e medição de crânios fetais em imagens de ultrassonografia bidimensionais. A rede utilizada consiste na estrutura do tipo *Encoder-Decoder* que pode ser dividida em duas metades. Segundo os autores, a rede é alimentada com três imagens em diferentes resoluções em suas camadas iniciais. A primeira metade da rede (*Encoder*) contém blocos de codificação chamados de *ResBlock* que incluem convoluções com *maxpooling* para o *downsampling* das imagens. A segunda metade (*Decoder*) é composta por blocos de decodificação que

são responsáveis por realizar o *upsampling* dos mapas de características e construir a saída final de segmentação. Os autores utilizaram um *dataset* contendo 999 imagens de ultrassons. Além disso, foram geradas 8823 imagens sintéticas com a técnica de *data augmentation*, para o treinamento da rede. Segundo os autores, os resultados obtidos demonstraram que a aprendizagem multitarefa leva a melhorias notáveis em uma rede de tarefa única e produz resultados de segmentação mais precisos.

O trabalho proposto por Sundaresan et al. (2017) consiste na localização automática do coração fetal em *frames* de vídeo de ultrassonografia (ecocardiografia). Além de classificar como pertencente a um dos três planos de visualização padrão: (visão da via de saída do ventrículo esquerdo,visão de três vasos e visão das quatro cavidades do coração), o trabalho visa auxiliar a identificação de anomalias cardíacas e de cardiopatias congênitas. Para a realização do projeto, os autores utilizaram FCN's, sendo empregadas três variações deste tipo de arquitetura: uma sendo a VGG projetada por Simonyan e Zisserman (2014) muito profunda com 16 camadas (*layers*) denominada (FCN-16*stride*); uma variação da VGG com 8 camadas (FCN-8*stride*); e outra com 32 camadas (FCN-32*stride*), em que suas camadas totalmente conectadas foram convertidas para camadas convolucionais. Para a análise do resultado obtido pela rede, em cada *frame* foi realizada uma anotação manual dada por um ultrassonografista experiente, onde o mesmo segmenta manualmente o centro do coração e atribui uma etiqueta indicando a qual plano de visualização o mesmo pertence. Com a análise realizada através das métricas de desempenho utilizadas pelos autores entre a segmentação obtida pela rede e da segmentação realizada pelo ultrassonografista, a rede que obteve melhor desempenho foi a FCN-16*stride*, tendo erro de classificação de 23,48%.

O trabalho de Sinclair et al. (2018) objetivou a utilização de uma rede FCN para medir automaticamente a circunferência da cabeça do feto e o diâmetro biparietal em imagens de ultrassonografia 2D. A rede FCN foi treinada com um *dataset* com aproximadamente 2.724 imagens da cabeça com anotações providas por 45 diferentes ultrassonografistas durante exames de rotina. A rede projetada pelos autores utilizou em sua arquitetura uma FCN VGG-16 contendo 16 camadas. Subsequentemente, nos dados de teste, uma elipse é ajustada aos contornos da segmentação para imitar o procedimento de medição realizado pelos ultrassonografistas. Após a segmentação, a circunferência da cabeça foi mensurada

utilizando a métrica *Ramanujan Approximation II* proposta por Barnard, Pearce e Schovanec (2001). Os autores compararam a eficiência do método com a segmentação manual realizada por profissionais da área de ultrassonografia em um subconjunto de 100 imagens presentes no teste. Segundo os autores, os resultados demonstraram que o modelo executa em um nível semelhante a um especialista humano e aprende a produzir segmentações precisas a partir de um grande conjunto de dados com anotações provenientes de diversos ultrassonografistas.

O trabalho de Wu et al. (2017) propôs uma customização na rede FCN, nomeando-a de *cascade FCN* (casFCN) para a segmentação do crânio e abdômen fetal. Os resultados experimentais foram aplicados em duas bases de imagens de ultrassonografia 2D, sendo uma consistindo apenas de crâneos e a outra apenas de abdômen. Os autores descreveram a arquitetura da rede proposta em três partes. Primeiramente foi adaptada uma rede popular FCN-8*stride* descrita em Long, Shelhamer e Darrell (2015) para realizar a segmentação da região desejada de forma eficiente. O segundo passo foi a implantação de uma pequena FCN com *Auto-Context model* para aperfeiçoar o mapa de predição (*prediction map*). Em terceiro, ao invés de usar paralelização foi utilizado um operador de somatório no mapa de predição do *Auto-Context model*. Os autores aplicaram três métricas de avaliação, sendo: *Dice coefficient similarity*, *Average Distance of Boundaries* e *Jaccard index*. Os resultados publicados por Wu et al. (2017) foram superiores tanto na segmentação do abdômen como na do crânio em comparação com outras abordagens comumente utilizadas na literatura como, por exemplo, a CNN, FCN-8s e a rede U-Net proposta por Ronneberger, Fischer e Brox (2015).

No trabalho proposto por Jang et al. (2018) foi utilizado uma CNN para classificar as principais estruturas anatômicas abdominais do feto nos possíveis planos de visão, sendo eles: bolha estomacal, líquido amniótico e veia umbilical, para posteriormente utilizar a transformada de *Hough transform-based ellipse detection* proposta por Bennett, Burridge e Saito (1999) para a detecção e medição automática da circunferência abdominal do feto. O método abordado no artigo pode ser descrito em três etapas principais: na primeira etapa é realizada a detecção da estrutura anatômica usando uma CNN, em que a saída da rede classificará e realizará uma segmentação semântica, isto é, classificará cada *pixel* da imagem de acordo com a classe que ele pertence. Isso é realizado atribuindo uma cor a cada *pixel* de acordo com a sua respectiva classe. Na segunda etapa é utilizada a transformada de *Hough*

para a detecção automática de elipses da região abdominal fetal. A terceira e última etapa consiste na verificação do plano de aceitação usando outra CNN para avaliação dos resultados obtidos pela rede, comparando-os com os realizados manualmente pelos especialistas. As imagens de treinamento foram utilizadas para gerar a verificação de aceitação pela segunda CNN e ajustar os parâmetros heurísticos para a detecção de elipse. Segundo os autores, os resultados obtidos pela rede mostraram um desempenho estável na maioria dos casos e até mesmo para imagens de ultrassonografias deterioradas por sombras e artefatos. No entanto, a rede necessita de um *dataset* de treinamento volumoso para obter resultados satisfatórios na classificação.

O trabalho de Ravishankar et al. (2016) utilizou uma abordagem híbrida combinando métodos populares de análise de textura com *Deep Learning* para a medição e detecção automática do abdômen fetal em imagens de ultrassonografia 2D. Esta abordagem utilizou uma CNN e HOG-GBM (*Histogram of Oriented Gradients - HOG and Gradient Boosting Machine - GBM*) como estimativa final de probabilidade. Conforme descrito no artigo, os autores estudaram quatro principais algoritmos descritores de características usados na literatura em imagens de ultrassonografia, sendo eles: *Haar-Features*, *Gray Level Co-Occurrence Matrix*, *Local Binary Patterns* e HOG. O desempenho destes descritores foi estudado em combinação com três classificadores bem estabelecidos na literatura, sendo eles: *Support Vector Machine*, *Random Forest* e o *GBM*. De acordo com as análises, o desempenho da combinação do *GBM* e *HOG* forneceu a menor taxa de erros de classificação. Os autores demonstraram que a abordagem híbrida realizando uma média entre as detecções obtidas pela CNN com as obtidas pelo HOG-GBM superou outras características de textura de última geração e de classificadores convencionais no problema de reconhecimento entre regiões do abdômen versus não abdômen.

Visando uma análise comparativa entre os trabalhos relacionados, o Quadro 1 apresenta as respectivas técnicas, métricas de avaliações e *datasets* utilizados.

Quadro 1- Quadro comparativo entre os trabalhos correlatos

Autores	Objetivo proposto	Rede proposta	Dataset	Métricas de Avaliação	Comparação de trabalhos
Sobhaninia et al., (2019)	Segmentação e medição de crânios fetais em imagens de ultrassonografia 2D.	<i>Multi-Task Deep Network</i> baseada na estrutura de rede Link-Net	Público	Coeficiente Dice, Diferença, Diferença Absoluta e Distância de Hausdorff	Comparou o resultado com o trabalho de Heuvel et al. (2018)
Li et al., (2017)	Segmentação automática do líquido amniótico e tecidos fetais em imagens de ultrassom 2D	FCN	Privado	Acurácia Global, Acurácia Média e Índice Jaccard	-
Sundaresan et al., (2017)	Localização automática do coração fetal em frames de vídeo de ultrassonografia	FCN	Público	Distância Euclidiana, Curva ROC, Verdadeiro e Falso Positivo	Comparou o resultado com o trabalho de Bridge; Ioannou e Noble (2017)
Sinclair et al., (2018)	Medição automática da circunferência da cabeça do feto e o diâmetro biparietal em imagens de ultrassonografia 2D	FCN	Privado	Coeficiente Dice, Estimação Biométrica (HC e BPD), Bland-Altman	Comparou o resultado com os trabalhos de Rueda et al., (2014) e Wu et al., (2017)
Wu et al. (2017)	Segmentação do crânio e abdômen fetal em imagens de ultrassonografia 2D.	<i>cascade FCN</i> (<i>casFCN</i>)	Privado	Coeficiente Dice, Índice Jaccard e Average Distance of Boundaries (Adb)	Comparou com as redes CNN, FCN-8s e U-Net de Ronneberger, Fischer e Brox (2015).
Jang et al. (2018)	Classificar estruturas anatômicas do Feto em Imagens de ultrassom 2D	CNN	Privado	Coeficiente Dice	-
Ravishankar et al. (2016)	Medição e detecção automática do abdômen fetal em imagens de ultrassonografia 2D	CNN e HOG-GBM	Privado	Coeficiente Dice Médio	Comparou com o método GBM de Becker et al., (2013)

Fonte: Autoria própria

3.1 CONSIDERAÇÕES FINAIS

Este capítulo descreveu alguns dos trabalhos recentes existentes na literatura que são voltados a área de segmentação de órgãos e ossos fetais em imagens de ultrassonografia utilizando técnicas de Aprendizado Profundo. O conteúdo apresentado descreveu de forma sucinta os principais pontos abordados em cada artigo. Contudo, não foi possível realizar uma comparação efetiva em relação ao desempenho dos trabalhos, já que na maioria dos casos utilizam bases diferentes.

Conforme os trabalhos recentes descritos, percebe-se que a rede FCN está sendo comumente utilizada para soluções de problemas na área. Em relação às métricas, o Coeficiente de Similaridade Dice foi o mais recorrente nos trabalhos revisados. Outro ponto a ser observado é que a maioria dos *datasets* utilizados pelos autores é de procedência privada, isto dificulta a comparação do presente trabalho com o destes autores.

A seguir, o Capítulo 4 descreverá a metodologia proposta para o desenvolvimento do presente trabalho.

4 METODOLOGIA

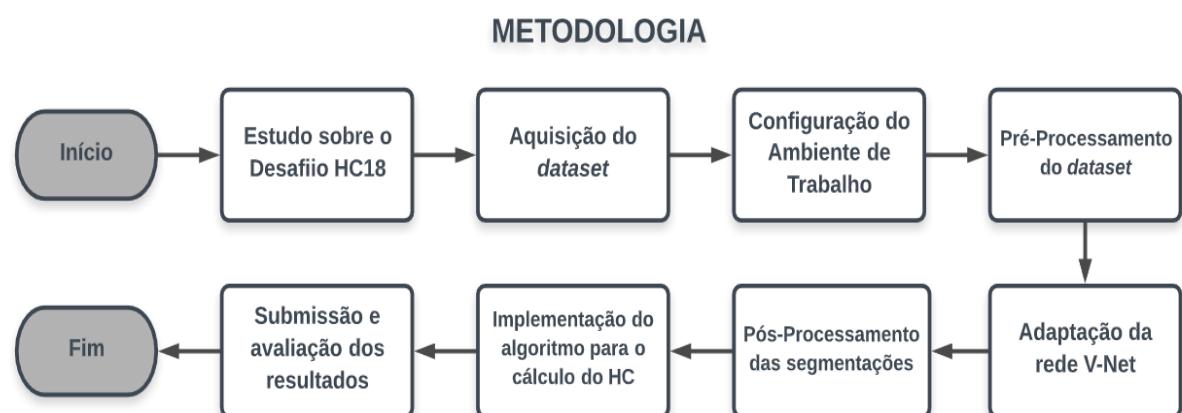
Este capítulo descreve a metodologia utilizada para o desenvolvimento do presente trabalho. Dessa maneira, a Seção 4.1 aborda as informações sobre a metodologia proposta para a realização do trabalho. Na seção 4.1.1 são abordados os estudos sobre o desafio HC18. Na seção 4.1.2, serão apresentadas as principais ferramentas utilizadas para a realização dos experimentos. Na seção 4.1.3, é descrita a etapa de pré-processamento realizada no *dataset*. Na seção 4.1.4 são abordadas as alterações impostas na arquitetura da rede V-Net para a segmentação de crânios fetais. Na seção 4.1.5 é descrita a etapa de pós-processamento realizada nas segmentações ineficientes. Na seção 4.1.6 será abordado o código criado para os cálculos do HC. E, por fim, na seção 4.1.7 será abordada a submissão dos resultados ao desafio.

4.1 METODOLOGIA PROPOSTA

A metodologia utilizada para a realização dos objetivos propostos é formada pelas seguintes etapas: estudo sobre o desafio HC18, configuração do ambiente de trabalho, pré-processamento do *dataset*, adaptação da rede V-Net, aplicação de pós-processamento, implementação do algoritmo para o cálculo do HC e submissão e avaliação dos resultados.

A Figura 20 descreve em diagrama de blocos as etapas descritas acima.

Figura 20 - Metodologia proposta



Fonte: Autoria própria

As etapas presentes no diagrama da Figura 20 serão descritas a seguir.

4.1.1 Estudos sobre o Desafio HC18

O *Grand Challenges in Biomedical Image Analysis*⁷ (em português, Grandes Desafios em Análise de Imagem Biomédica) é uma plataforma que possibilita hospedagem de desafios científicos com o intuito de facilitar acesso aos pesquisadores a diferentes desafios relacionados a imagens médicas, visando a criação de melhores abordagens comparadas as já existentes na literatura. Segundo dados estatísticos do próprio site⁸, para o mês de janeiro de 2019 existiam 20.749 usuários cadastrados no sistema, tendo 102 desafios ocultos e 35 desafios públicos, em que um destes é o desafio HC18, o qual será descrito a seguir.

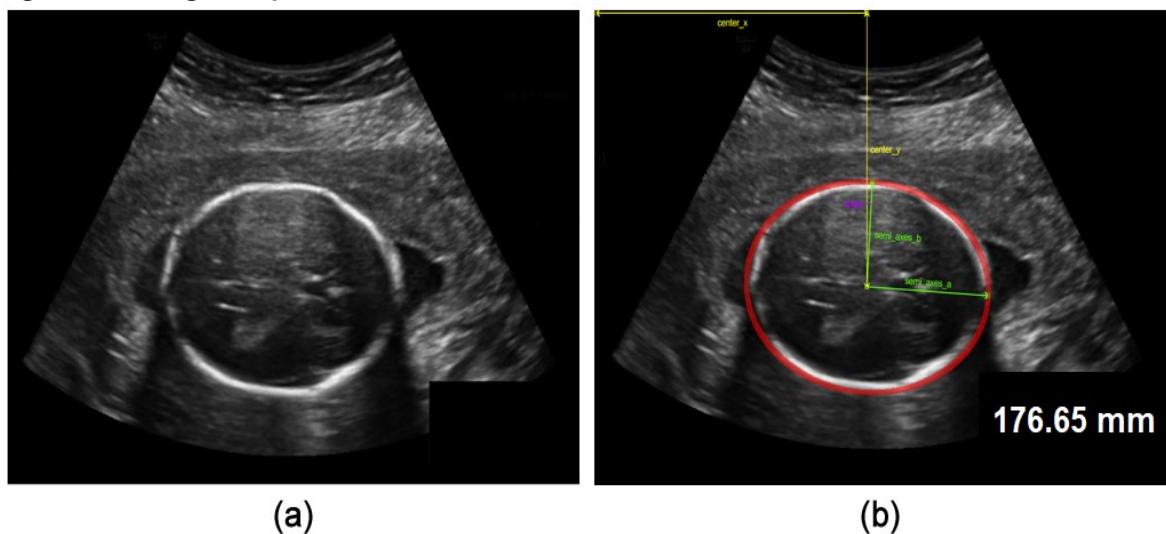
4.1.1.1 O desafio HC18

Durante a gestação, a ultrassonografia é usada para medir a biometria fetal. Uma das medidas extraídas é a circunferência da cabeça do feto (*head circumference* - HC). O HC pode ser usado para estimar a idade gestacional e monitorar o crescimento do feto. O HC é medido em uma seção transversal específica da cabeça fetal, que é chamada de plano padrão (GRAND-CHALLENGES©, 2019). O desafio HC18⁸ representado pela Figura 21, propõe a construção de um algoritmo para a medição automática da circunferência da cabeça do feto em imagens de ultrassom 2D. O desafio contém 372 inscritos e mais de 750 submissões (dados referentes ao mês de julho de 2019).

⁷ <https://grand-challenge.org/>

⁸ <https://hc18.grand-challenge.org/>

Figura 21 - Imagem representativa do desafio HC18



Fonte: Adaptado de HC18 (2018)

Dada uma imagem de ultrassom 2D conforme visto na Figura 21 (a), o objetivo é segmentar o crânio fetal conforme demonstrado pela elipse na cor vermelha presente na Figura 21 (b) e então calcular cinco medidas: centro x (*center x*), centro y (*center y*), semieixo a (*semi axes a*), semieixo b (*semi axes b*) e ângulo (*angle*), as quais serão descritas na seção 4.1.6.

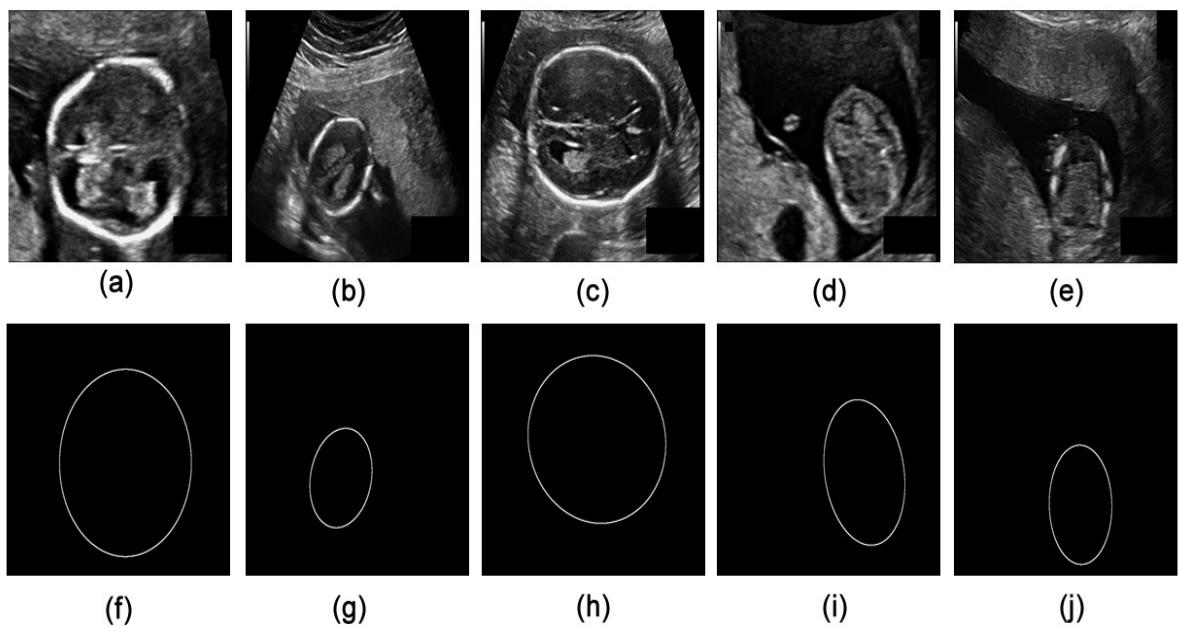
4.1.1.2 O dataset

O *dataset* utilizado para o trabalho foi disponibilizado *online*⁹ por Heuvel et al. (2018) para a realização do desafio. O mesmo conta com 1.334 imagens de ultrassons 2D, sendo 999 para treinamento do algoritmo e 335 para o teste. Todas as imagens estão no formato PNG (*Portable Network Graphic*) em escala de cinza, tendo a resolução 800 por 540 *pixels* com um tamanho de *pixel* variando entre 0,052 a 0,326 mm (milímetros). O *dataset* contém ainda 999 imagens com a anotação manual da circunferência da cabeça para cada HC, feita por um ultrassonografista treinado. O *dataset* possui dois arquivos no formato CSV (*Comma-Separated Values*), um contendo a medida do perímetro cefálico e o tamanho do *pixel* para cada HC anotado no conjunto de treinamento e o outro arquivo contendo o tamanho do *pixel* de cada imagem do conjunto de teste, o qual será necessário para os cálculos do HC para posterior submissão ao desafio.

A Figura 22 ilustra algumas das imagens presentes no *dataset* de treinamento.

⁹ <http://doi.org/10.5281/zenodo.1327317>

Figura 22 - Representação das imagens de ultrassonografia e as *ground truth* contidas no dataset



Fonte: Autoria própria

Na Figura 22, (a)-(e) representam-se cinco imagens de crânios fetais gerados por meio da ultrassonografia, presentes no *dataset* disponibilizado, e de (f)-(j) são as chamadas *ground truth*, as quais são as segmentações manuais das imagens (a)-(e) realizadas por um ultrassonografista treinado.

4.1.1.3 Processo para a submissão e avaliação

Para a submissão do desafio deve-se enviar um arquivo CSV contendo seis colunas e 336 linhas. A primeira coluna contém o nome da imagem de ultrassom e seu formato, as outras cinco devem descrever as dimensões da elipse gerada com a segmentação do crânio fetal, sendo que as quatro primeiras colunas deverão ser descritas em milímetros e a última coluna em radianos.

A Tabela 1 exemplifica a estrutura do arquivo que deve ser enviado para a submissão dos resultados.

Tabela 1 - Exemplo de tabela para a submissão dos resultados

filename	center_x_mm	center_y_mm	semi_axes_a_mm	semi_axes_b_mm	angle_rad
388_HC.png	72.378073183	60.803647248	32.482252220	23.374422688	23.374422688

Fonte: HC18 (2018)

A avaliação do desafio é realizada de forma automática, o ranqueamento é dado pela média obtida em cada métrica usada, sendo: *Abs Difference*, Coeficiente de Similaridade Dice (*Dice Similarity Coefficient*), Distância de Hausdorff (*Hausdorff Distance*). Essas três métricas foram descritas na Seção 2.5.

4.1.2 Configuração do Ambiente de Trabalho

O desenvolvimento deste trabalho foi baseado em linguagens e bibliotecas abertas para o público em geral, e que são amplamente utilizadas para aplicações gerais em Aprendizado Profundo.

Na configuração do ambiente de trabalho foi preciso instalar um conjunto de bibliotecas e pacotes de *software* para tornar possível a construção e execução da rede. As principais bibliotecas utilizadas foram o Tensorflow¹⁰, cuDNN¹¹ e o Keras¹². Todas as demais dependências foram instaladas com o auxílio da plataforma Anaconda¹³. A linguagem de programação utilizada na implementação da rede foi a linguagem Python¹⁴ e para a etapa do pré-processamento do *dataset* foi utilizada a linguagem MATLAB®¹⁵. Já na etapa de pós-processamento, foi utilizada Python OpenCV¹⁶. Estas sete principais ferramentas necessárias para a construção do ambiente de trabalho serão brevemente introduzidas a seguir:

- **Python:** Python é uma linguagem de programação interpretada, orientada a objetos, de alto nível com tipagem dinâmica (ROSSUM, 1995). Um dos aspectos que torna esta linguagem uma escolha tão popular na área de Aprendizado de Máquina e Aprendizado Profundo é a abundância de bibliotecas e *frameworks* que facilitam a codificação e economizam tempo de desenvolvimento;
- **TensorFlow:** O TensorFlow é uma biblioteca de código aberto para computação numérica e Aprendizado de Máquina disponibilizado pelo *Google Brain Team* em novembro de 2015. Devido a sua ampla documentação, o

¹⁰ <https://www.tensorflow.org/>

¹¹ <https://developer.nvidia.com/cudnn>

¹² <https://keras.io/>

¹³ <https://www.anaconda.com/>

¹⁴ <https://www.python.org/>

¹⁵ <https://www.mathworks.com/products/matlab.html>

¹⁶ <https://pypi.org/project/opencv-python/>

TensorFlow tornou-se uma das bibliotecas mais utilizadas para *Deep Learning* e *Machine Learning*. Sua arquitetura flexível permite a fácil implantação de computação em várias plataformas como desktops, GPUs, entre outros (ABADI et al., 2016);

- **Keras:** O Keras é uma biblioteca de alto nível escrita na linguagem de programação Python, sendo desenvolvida para a rápida prototipagem de modelos de *Deep Learning*. Esta biblioteca utiliza como base o TensorFlow ou Theano¹⁷, fornecendo uma fácil implementação de vários componentes de uma rede neural, como: camadas de convolução, *pooling*, funções de ativação, etc, além de diversas ferramentas para facilitar o trabalho com dados e imagens (CHOLLET, 2017);
- **NVIDIA cuDNN:** A biblioteca de rede neural profunda NVIDIA CUDA® (NVIDIA CUDA Deep Neural Network - cuDNN) fornece a funcionalidade de aceleração via GPU para operações comuns em redes neurais profundas, como por exemplo: camadas de convolução, *pooling*, normalização e funções de ativação, aumentando drasticamente a velocidade de processamento da rede e consequentemente diminuindo o seu tempo de treinamento. O uso da cuDNN possibilita a aceleração dos principais *frameworks*, sendo: Caffe, Caffe2, Chainer, Keras, MATLAB®, MxNet, TensorFlow e o PyTorch (NVIDIA DEVELOPER, 2018);
- **Anaconda:** Anaconda é uma distribuição Python que visa simplificar a instalação e gerenciamento de pacotes. Sua distribuição fornece mais de 1.000 pacotes, o que elimina a necessidade de instalar cada biblioteca de forma independente. Outra funcionalidade da Anaconda é a criação de diferentes ambientes Python de forma fácil, cada um contendo suas próprias configurações (ANACONDA INC, 2018);
- **MATLAB®:** O MATLAB® é um software voltado para cálculos numéricos e gráficos científicos. Sua linguagem permite que soluções de problemas sejam escritos quase precisamente como são escritos matematicamente. O seu ambiente de programação é de fácil utilização, além de conter diversos *toolboxes* que são bibliotecas auxiliares para várias áreas da ciência e tecnologia (VIEIRA, 2004);

¹⁷ <http://deeplearning.net/software/theano/>

- **OpenCV:** sigla para *Open Source Computer Vision Library* (em português, Biblioteca de Computação Visual em Código Aberto), o OpenCV é uma das principais bibliotecas multiplataforma de código aberto contando com mais de 500 funções para o uso em visão computacional, tratamento de imagens, reconstrução 3D, Aprendizado de Máquina, entre outras (OPENCV, 2019).

O Quadro 2 apresenta as versões das principais ferramentas necessárias para a construção do ambiente de trabalho.

Quadro 2 - Versões utilizadas

Aplicação	Versão
Python	3.6.6
Tensorflow-Gpu	1.10.0
Keras	2.2.4
Cudnn	7.1.4
Anaconda	4.5.11
Matlab®	2017b
OpenCV Python	3.1.0

Fonte: Autoria própria

4.1.2.1 Configurações de máquina

Os tópicos seguintes especificam as configurações da máquina utilizadas para a realização do trabalho:

- **Processador:** Intel(R) Core (TM) i7-3370K CPU 3.50GHz 3.90GHz;
- **Memória RAM:** 8 GB;
- **Placa de Vídeo:** NVIDIA GeForce GTX TITAN – 6GB dedicada;
- **Sistema Operacional:** *Microsoft Windows 10 PRO 64 bits.*

A Tabela 2 representa as configurações específicas da placa de vídeo utilizada.

Tabela 2 - Configurações específicas da placa de vídeo GTX TITAN

Parâmetros	Valores
Total de Núcleos CUDA	2688
Clock Padrão GPU	836Mhz
Quantidade de memória	6GB
Clock memória	6008Mhz
Interface de memória	364-bit GDDR5

Fonte: Autoria própria

4.1.3 Pré-Processamento do *Dataset*

Nesta etapa foram realizadas duas alterações no *dataset*. A primeira alteração foi a redução da dimensão das imagens de treinamento e de teste para a resolução de 512 por 512 *pixels*, a fim de diminuir o custo computacional durante o treinamento da rede. A segunda alteração foi realizada nas imagens de *ground truth*, em que foi criado um algoritmo para preencher as 900 imagens, fazendo-as parecer como elipses sólidas. Para o preenchimento das elipses foi escolhida a linguagem de programação MATLAB®, em que a mesma conta com diversas funções existentes para a área de processamento de imagens, dentre elas a função *imfill()* foi necessária para atender os objetivos, esta que será brevemente descrita a seguir:

- ***imfill()*:** A função *imfill()* executa uma operação de preenchimento por inundação (*flood- fill*) em imagens binárias ou em escala de cinza. Esta função utiliza um algoritmo baseado em reconstrução morfológica descrito em (SOILLE, 1999) (MATLAB, 2019).

A Figura 23 ilustra o código na linguagem MATLAB® com utilização da função *imfill()* para o preenchimento das elipses.

Figura 23 - Representação do código utilizado para o preenchimento das elipses

```
% author: Everton Leonardo Skeika

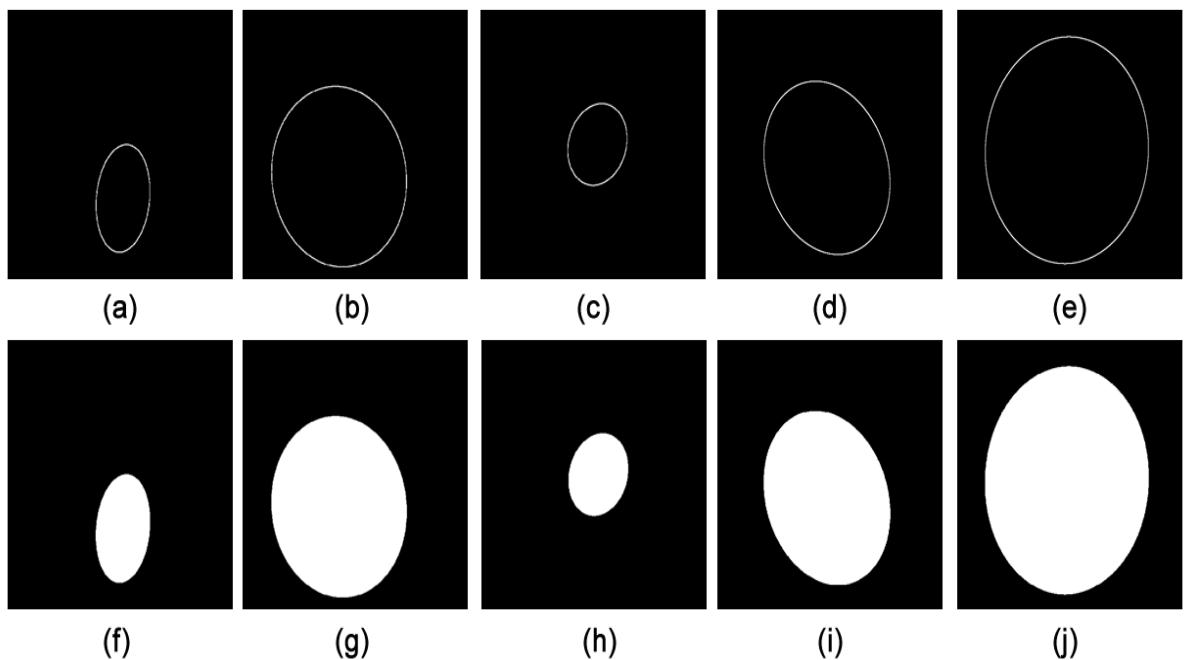
location = 'C:\Users\Everton\Documents\ground truth original\*.png';
ds = imageDatastore(location);
n=0;
while hasdata(ds)

    img = read(ds);
    img = imfill(img, 'holes');
    imwrite(img, sprintf('%d.png', n));
    n=n+1;
end
```

Fonte: Autoria própria

A Figura 24 ilustra o resultado do preenchimento das imagens de *ground truth*.

Figura 24 - Resultado do preenchimento das *ground truth*



Fonte: Autoria própria

Conforme pode ser observado, a Figura 24 (a)-(e) representam as segmentações originais previamente existentes no *dataset* e (f)-(j) representa o resultado obtido pelo algoritmo de preenchimento.

Esta alteração foi necessária devido aos resultados iniciais obtidos pela rede proposta, na qual as elipses resultantes da segmentação ficaram com suas bordas incompletas. Neste sentido, seria impraticável o cálculo da elipse de uma forma precisa, pois não haveria como delimitar a área faltante com exatidão.

Devido ao fato do desafio HC18 não disponibilizar as *ground truth* de teste para a avaliação dos resultados, foram retiradas aleatoriamente 297 imagens das 999 (aprox. 30%) existentes no treinamento com suas respectivas *ground truth* previamente preenchidas para tornar possível a comparação dos resultados obtidos pela rede e posteriormente, com a análise do desempenho viabilizar o projeto de possíveis melhorias na arquitetura, aprimorando os resultados da segmentação.

4.1.4 Adaptação da Rede V-Net

A rede utilizada para a realização das segmentações foi uma adaptação da rede original V-Net projetada por Milletari, Navab e Ahmadi (2016). Esta segue a arquitetura original descrita na seção 2.4.3.2, exceto pelas mudanças dos *kernels* de convolução para a segmentação de imagens em 2D, pela inclusão de *dropout* (ver

subseção 2.4.2.2), *batch normalization* (ver subseção 2.4.2.3), e o uso da função *sigmoid* em vez da *softmax* na última camada da rede. O código base utilizado está disponível *online*¹⁸ sob a licença (*GNU General Public License v3.0 - GPL*)¹⁹.

Inicialmente foram realizadas alterações básicas na rede para a execução do *dataset* com as imagens em resolução 512 por 512 *pixels*. A rede foi treinada com diversas épocas (*epoch*) e passos (*steps*) visando encontrar bons valores para o seu treinamento. Em Aprendizado Profundo, uma época significa uma iteração sobre todo o *dataset* de treinamento, e um passo condiz ao treinamento de um *batch* (lote) por vez.

A Tabela 3 descreve a taxa de acurácia (*accuracy*) e perda (*loss*) durante o treinamento da rede e o tempo de execução em relação ao número de épocas de acordo com a configuração de máquina utilizada, previamente descrita na subseção (4.1.2.1).

Tabela 3 – Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação ao número de épocas e de passos utilizados para o treinamento da rede V-Net

Rede	Passos por época	Épocas	Acurácia (máxima)	Perda (mínima)	Tempo de Execução
V-Net base	100	100	0.9874	0.0211	~4 minutos
	100	500	0.9942	0.0097	~17 minutos
	500	100	0.9941	0.0099	~3 horas
	500	300	0.9970	0.0049	~8,33 horas
	1000	100	0.9964	0.0061	~5,5 horas
	1000	300	0.9920	0,0117	~16,25 horas
	2000	300	0.9978	0.0036	~36,00 horas

Fonte: Autoria própria

Conforme apresentado na Tabela 3, percebe-se que a rede treinada com 2000 passos e 300 épocas obteve a maior taxa de acurácia e menor valor de perda durante o treinamento, porém conforme avaliação das segmentações obtidas pelas métricas de desempenho (Tabela 4) o melhor resultado das segmentações foi obtido com 1000 passos e 300 épocas. Este valor será utilizado como padrão para o treinamento da rede em suas novas edições.

¹⁸ <https://github.com/FENGShuanglang/2D-Vnet-Keras>

¹⁹ Esta licença permite que os códigos abertos estejam livres para serem modificados, compartilhados, entre outros.

A Tabela 4 descreve os resultados médios obtidos através das métricas de precisão, especificidade, sensibilidade, acurácia, Dice e Jaccard entre as segmentações obtidas pela rede nos diversos tempos de treinamentos.

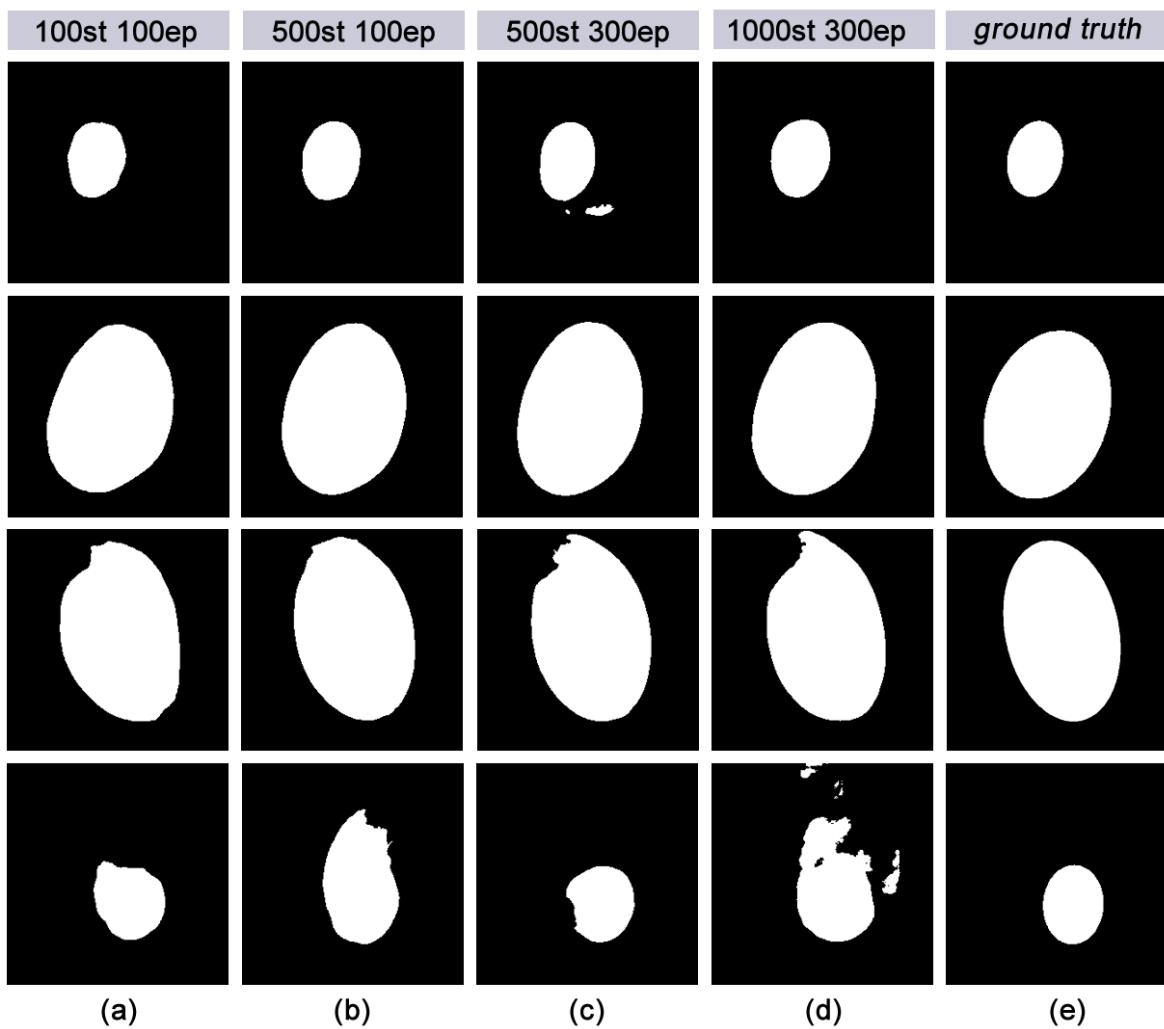
Tabela 4 - Tabela comparativa dos resultados das métricas de desempenho em relação ao número de épocas e de passos utilizados para o treinamento da rede V-Net

V-Net base	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
100st 100ep	0.97826	0.99023	0.96784	0.98609	0.97258	0.94741
100st 500ep	0.97516	0.99068	0.97790	0.98748	0.97620	0.95382
500st 100ep	0.97418	0.99059	0.97624	0.98697	0.97462	0.95136
500st 300ep	0.97697	0.99132	0.97897	0.98754	0.97552	0.95334
1000st 100ep	0.97468	0.99062	0.97850	0.98756	0.97611	0.95406
1000st 300ep	0.97699	0.99200	0.97674	0.98778	0.97651	0.95447
2000st 300ep	0.97723	0.99199	0.97643	0.98764	0.97648	0.95444

Fonte: Autoria própria

Embora os valores iniciais obtidos pela rede sejam considerados bons, existe a necessidade de aprimorá-la para resolver problemas relacionados ao *overfitting*. A Figura 25 ilustra alguns dos resultados das segmentações obtidas pela rede nos diferentes tempos de treinamento.

Figura 25- Resultados da segmentação obtidos pela rede V-Net 2D nos diferentes tempos de treinamento



Fonte: Autoria própria

Conforme apresentado na Figura 25, alguns resultados das segmentações obtidas pela rede são ineficientes, essa ineficiência está relacionada ao *overfitting* da rede e pela falta de generalização das imagens do treinamento. Visando aprimorar a sua performance foram aplicadas algumas alterações e análises na arquitetura da rede V-Net 2D utilizada como base, as quais serão descritas a seguir.

4.1.4.1 Uso de *batch normalization*

Embora esta técnica não tenha sido descrita na rede 3D original projetada por Milletari, Navab e Ahmadi (2016), ela foi aplicada na rede V-Net 2D utilizada como base para o presente trabalho.

Neste caso, primeiramente foi analisado o desempenho durante o treinamento da rede V-Net 2D com e sem a técnica *Batch Normalization*. A Tabela 5 descreve a maior taxa de acurácia, menor perda e o tempo total de processamento durante o treinamento da rede em cada caso.

Tabela 5 - Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação à aplicação da técnica *batch normalization*

V-Net base	Acurácia (máxima)	Perda (mínima)	Tempo de Execução (horas)
Sem BN	0.99760	0.00289	~14,33
Com BN	0,99210	0,01170	~16,25

Fonte: Autoria própria

Embora os valores de acurácia, perda e até o tempo de processamento sejam melhores sem a técnica de BN, os resultados obtidos pelas métricas nas comparações entre os resultados das segmentações e as *ground truth* (Tabela 6) foram piores do que com a sua utilização, ou seja, sem a técnica de BN o *overfitting* da rede aumentou.

Tabela 6 - Tabela comparativa dos resultados das métricas de desempenho em relação ao uso da técnica *batch normalization*

V-Net	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
Sem BN	0.97473	0.99086	0.97911	0.98697	0.97647	0.95438
Com BN	0.97699	0.99200	0.97674	0.98778	0.97651	0.95447

Fonte: Autoria própria

Conforme os resultados superiores de precisão, especificidade, acurácia, Dice e Jaccard mostrados na Tabela 6, tornou-se válido a permanência desta técnica na arquitetura da rede.

Outro ponto estudado foi o posicionamento da técnica de BN no código disposto. A técnica é aplicada antes da função de ativação PReLU. Trabalhos como o de Ham, Kim e Kim (2017) abordam alterações na posição do uso de BN. Portanto, foi testado se o uso da técnica de BN após a função de ativação PReLU poderia melhorar o desempenho da rede. A Tabela 7 apresenta os valores obtidos pelas métricas de avaliação em relação à reestruturação da técnica de BN empregada na rede.

Tabela 7 - Tabela comparativa dos resultados das métricas de desempenho em relação à reestruturação do *batch normalization*

V-Net	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
Com BN	0.97699	0.99200	0.97674	0.98778	0.97651	0.95447
BN editado	0.97722	0.99138	0.97863	0.98807	0.97766	0.95662

Fonte: Autoria própria

Conforme os valores apresentados na Tabela 7, o uso do BN após a função de ativação PReLU aprimorou ainda mais o desempenho nas segmentações e, portanto, esta alteração foi agregada na arquitetura da rede.

4.1.4.2 Alteração na camada de ativação

A última camada de ativação de uma rede FCN tem a importante tarefa de gerar a probabilidade de cada *pixel* da imagem de saída pertencer ao *background* ou *foreground*. Tanto a rede U-Net quanto a V-Net em suas versões originais utilizaram a função *Softmax*. No entanto, na rede V-Net 2D utilizada como base, foi empregado o uso da função *sigmoid*.

Visando uma análise comparativa foram testadas outras funções na última camada da rede, sendo elas: *softmax* e *hard_sigmoid*. Vale ressaltar que os testes foram aplicados na rede já com as alterações impostas nos tópicos anteriores. A Tabela 8 apresenta a taxa de acurácia, perda e o tempo de execução durante o treinamento da rede de acordo com o tipo de função aplicada em sua última camada.

Tabela 8 - Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação a alteração da função de ativação da última camada da rede

V-Net	Acurácia (máxima)	Perda (mínima)	Tempo de Execução (horas)
<i>softmax</i>	0.31000	0.53267	~18,25
<i>sigmoid</i> (padrão)	0.99760	0.00407	~17,33
<i>hard_sigmoid</i>	0.99761	0.00408	~15,91

Fonte: Autoria própria

Observa-se pelos resultados apresentados na Tabela 8 que a função de ativação *softmax* obteve um desempenho extremamente ineficiente no treinamento da rede, além de ter um custo computacional maior. Já a função *hard_sigmoid* obteve

um custo computacional menor durante o treinamento, tendo um desempenho muito semelhante à função *sigmoid*.

Visando analisar a eficiência das segmentações a Tabela 9 descreve os resultados obtidos através das métricas de precisão, especificidade, sensibilidade, acurácia, Dice e Jaccard entre as segmentações obtidas de acordo com a função de ativação utilizada na última camada.

Tabela 9 - Tabela comparativa dos resultados das métricas de desempenho em relação ao tipo de função de ativação aplicada na última camada da rede

V-Net	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
<i>softmax</i>	-	-	-	-	-	-
<i>sigmoid</i>	0.97722	0.99138	0.97863	0.98807	0.97766	0.95662
<i>hard_sigmoid</i>	0.97681	0.99135	0.97863	0.98809	0.97774	0.95665

Fonte: Autoria própria

Como apresentado na Tabela 9, não foi possível realizar as comparações entre as segmentações obtidas pela rede com a função *softmax* devido ao fato da baixa taxa de acurácia durante o seu treinamento, já que os resultados das segmentações geraram apenas imagens pretas. Já com a utilização da função *hard_sigmoid* além da diminuição do treinamento da rede, houve uma pequena melhora na acurácia das segmentações e, portanto, a função *sigmoid* foi substituída por ela.

4.1.4.3 Alteração da função de ativação

A função de Ativação empregada na rede original é a PReLU, sendo aplicada em todas as etapas de convolução da rede. Visando aprimorar ainda mais o desempenho das segmentações foram realizados testes com outras quatro funções de ativação, sendo elas: ReLU, ELU (*Exponential Linear Units*), LeakyReLU e ThresholdedReLU. A Tabela 10 descreve o maior valor da acurácia e menor valor de perda obtido durante o treinamento da rede para cada função de ativação.

Tabela 10 - Tabela comparativa dos resultados obtidos para cada função de ativação

Função de Ativação	Acurácia (max)	Perda (min)
PReLU (padrão)	0.99761	0.00408
ReLU	0.99760	0.00402
ELU	0.99760	0.00400
LeakyReLU	0.99720	0.00463
ThresholdedReLU	0.97890	0.01330

Fonte: Autoria própria

Conforme os valores observados na Tabela 10, percebe-se que as funções de ativação ReLU e ELU se assemelharam muito com o desempenho da função de ativação padrão PReLU da rede V-Net.

A Tabela 11 mostra os valores médios obtidos pelas métricas de desempenho na análise dos resultados obtidos pela segmentação.

Tabela 11 - Tabela comparativa dos resultados das métricas de desempenho em relação as Funções de Ativação aplicadas

Função de Ativação	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
PReLU (padrão)	0.97681	0.99135	0.97863	0.98809	0.97774	0.95665
ReLU	0.97843	0.99178	0.97715	0.98794	0.97753	0.95633
ELU	0.97747	0.99106	0.97875	0.98812	0.97786	0.95693
LeakyReLU	0.97088	0.98988	0.97972	0.98730	0.97445	0.95152
ThresholdedReLU	0.77519	0.90632	0.86119	0.88165	0.78566	0.67029

Fonte: Autoria própria

Conforme os valores descritos na Tabela 11, a função de ativação ELU obteve o melhor valor nas métricas médias de acurácia, Dice e Jaccard. Portanto, tornou-se válido a alteração da função de PReLU para ELU.

4.1.4.4 Aplicação de *dropout*

Uma das técnicas que visa diminuir o *overfitting* é a técnica de *dropout* (ver subseção 2.4.2.2). A rede utilizada como base aplicou esta técnica no caminho de contração. Inicialmente foi retirado o *dropout* em toda rede para analisar se a técnica estava sendo útil no desempenho. Foi verificado que o *overfitting* aumentou e, como consequência, o desempenho das segmentações pioraram, conforme apresentado na

Tabela 12 e Tabela 13. Neste sentido, foram realizados testes com a inclusão desta técnica também no caminho de expansão, sendo aplicada após todas as etapas de convolução existentes na rede. A rede foi treinada com valores de *dropout* de 0.2, 0.5 e 0.7, ou seja, 20%, 50% e 70% de ativações mantidas.

A Tabela 12 apresenta o maior valor de acurácia e menor valor de perda obtido durante o treinamento da rede para cada valor de *dropout* utilizado.

Tabela 12 - Tabela comparativa da Acurácia, Perda e o Tempo de execução em relação aos valores de dropout

V-Net valor <i>dropout</i>	Acurácia (max)	Perda (min)	Tempo de Execução (horas)
sem	0.99760	0.00392	~13,66
0.2	0.99630	0.00620	~13,90
0.5	0.99090	0.01508	~13,92
0.7	0.96280	0.06385	~15,00

Fonte: Autoria própria

A Tabela 13 descreve os valores médios obtidos pelas métricas de desempenho na análise dos resultados obtidos pela segmentação com o uso da técnica de *dropout*.

Tabela 13- Tabela comparativa dos resultados das métricas de desempenho em relação aos valores de dropout

V-Net valor <i>dropout</i>	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
sem	0.97902	0.99212	0.97615	0.98793	0.97730	0.95593
0.2	0.97950	0.99215	0.97595	0.98802	0.97747	0.95621
0.5	0.98643	0.99491	0.96032	0.98597	0.97263	0.94752
0.7	0.98652	0.99586	0.95968	0.98563	0.97245	0.94698

Fonte: Autoria própria

Conforme descrito na Tabela 13, o aumento do uso do *dropout* não superou o desempenho nas segmentações da rede com as alterações descritas nos tópicos anteriores, em que o maior valor médio da acurácia foi de 0.98812. Neste sentido, será mantido o *dropout* apenas nas camadas de contração.

4.1.4.5 Alteração no *data augmentation*

Embora a rede use *data augmentation* (ver subseção 2.4.2.1), foi necessário realizar algumas alterações para tentar minimizar o seu *overfitting*. Para aumentar a generalização do *dataset* de treinamento foram aplicadas as seguintes operações: *rotation_range*, *zoom_range*, *width_shift*, *height_shift*, *shear_range*, *horizontal_flip* e *vertical_flip*, as quais serão brevemente descritas a seguir:

- ***rotation_range***: aleatoriamente rotaciona a imagem em valores de 0 a 180 graus;
- ***zoom_range***: aleatoriamente aumenta as imagens de acordo com o valor utilizado;
- ***width_shift***: aleatoriamente desloca a imagem em direção ao eixo horizontal;
- ***height_shift***: aleatoriamente desloca a imagem em direção ao eixo vertical;
- ***shear_range***: aleatoriamente distorce a imagem aplicando a transformação *shear*;
- ***horizontal_flip***: aleatoriamente gira a imagem horizontalmente, a entrada é dada por um valor booleano *True* ou *False*;
- ***vertical_flip***: aleatoriamente gira a imagem verticalmente, a entrada é dada por um valor booleano *True* ou *False*.

A Tabela 14 descreve os valores utilizados para a geração das novas imagens sintéticas para o treinamento da rede, vale ressaltar que foram realizados vários testes empíricos, onde foram utilizados diversos valores e aplicados outras operações de aumento de dados. No entanto, a rede que obteve o melhor desempenho foi com os valores descritos nesta tabela.

Tabela 14 - Tabela demonstrativa das operações e valores utilizados para a geração de imagens sintéticas pela técnica de *data augmentation*

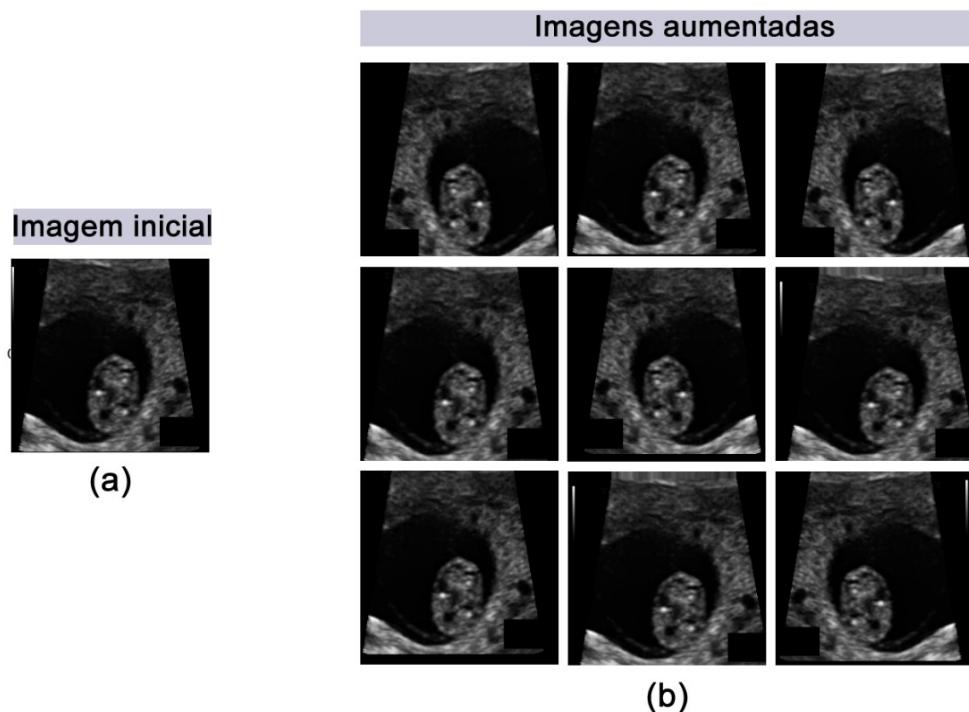
Operações	Valor
<i>rotation_range</i>	0.15
<i>width_shift_range</i>	0.05
<i>height_shift_range</i>	0.05
<i>shear_range</i>	0.10
<i>zoom_range</i>	0.10
<i>horizontal_flip</i>	<i>True</i>
<i>vertical_flip</i>	<i>True</i>

Fonte: Autoria própria

Conforme observado na Tabela 14 foi utilizado valores entre 0.05 a 0.15, pois o *dataset* utilizado não contém imagens tão distintas uma das outras. Isto quer dizer que, os novos dados sintéticos gerados a partir das transformações das amostras de treinamento tiveram uma grande semelhança com as amostras originais.

A Figura 26 ilustra algumas das imagens sintéticas geradas pelas operações do *data augmentation* a partir de uma imagem referência contida no *dataset* utilizado.

Figura 26- Exemplo de aplicação de *data augmentation* no *dataset* utilizado (a) Imagem original. (b) Imagens aumentadas



Fonte: Autoria própria

Além disso, foram realizados testes com a alteração do valor do *batch* (em português, lote) de 2 para 4 e 6 ou seja, este processo aumentou ainda mais a quantidade de imagens artificiais que foram geradas para o treinamento da rede, e como consequência, houve um aumento considerável do custo computacional. A Tabela 15 descreve a análise do desempenho da rede com e sem a técnica de *data augmentation* e com as alterações no valor do *batch*.

Tabela 15 - Tabela comparativa dos resultados obtidos com as edições no *data augmentation*

<i>Data augmentation</i>	<i>Batch</i>	Acurácia (max)	Perda (min)	Tempo de Execução (horas)
sem	-	0.99760	0.00392	~12,66
com	2(padrão)	0.99760	0.00400	~16,00
com	4	0.99750	0.00403	~23,34
com	6	0.99770	0.00370	~31,75

Fonte: Autoria própria

Pelos valores descritos pela Tabela 15 percebe-se que a alteração do *batch* de 2 para 4 obteve um aumento superior no tempo de processamento da rede em cerca de 45% e de 2 para 6 em cerca de 95%.

A Tabela 16 apresenta os valores médios obtidos pelas métricas de desempenho entre os resultados obtidos pela segmentação em relação ao tamanho do *batch* utilizado para a geração das imagens sintéticas.

Tabela 16- Tabela comparativa dos resultados das métricas de desempenho em relação às edições no tamanho do *batch*

<i>Batch</i>	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
sem	0.97717	0.99096	0.92121	0.97825	0.93339	0.93339
2(padrão)	0.97577	0.99052	0.98059	0.98816	0.97794	0.95707
4	0.98009	0.99209	0.97725	0.98859	0.97842	0.95805
6	0.97768	0.99125	0.97869	0.98832	0.97793	0.95713

Fonte: Autoria própria

Pelos valores previamente apresentados pela Tabela 16, percebe-se que a alteração do *batch* de 2 para 4 obteve um aumento circunstancial na performance das segmentações. Neste sentido, torna-se válida esta alteração na arquitetura da rede.

4.1.4.6 Alteração na função de perda

Uma função de perda tem o objetivo de medir a diferença entre a saída prevista e a saída real obtida pelo treinamento da rede em antecipação. Quanto maior o valor da perda menos preciso será o modelo.

Milletari, Navab e Ahmadi, (2016) em seu trabalho propuseram uma nova função de perda baseada na métrica de similaridade Dice para a rede V-Net. Visando uma melhora no desempenho foram realizados testes com outras funções de perda, como por exemplo: a função *binary_crossentropy* utilizada pela rede U-Net, a função Tversky *loss* proposta por Salehi, Erdogan e Gholipour (2017), a função Focal *loss* proposta por Lin et al. (2017) e a Lovasz-Softmax *loss* projetada por Berman, Rannen e Blaschko (2018).

A Tabela 17 apresenta os valores médios obtidos pelas métricas de desempenho em relação à cada função de perda empregada na arquitetura da rede.

Tabela 17- Tabela comparativa dos resultados das métricas de desempenho em relação ao uso da função de Perda

Função de Perda	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
Dice (padrão)	0.98009	0.99209	0.97725	0.98859	0.97842	0.95805
Binary_crossentropy	0.97605	0.99100	0.97730	0.98744	0.97637	0.95414
Focal	0.98282	0.99334	0.97501	0.98861	0.97868	0.95849
Tversky	0.97608	0.99057	0.98194	0.98866	0.97876	0.95866
Lovasz-Softmax	0.98386	0.99324	0.97155	0.98821	0.97717	0.95607

Fonte: Autoria própria

Pelos valores previamente descritos pela Tabela 17, percebe-se que o uso da função de perda Tversky *loss* obteve um melhor resultado nas segmentações dos crânios fetais neste sentido, ela substituirá a função de perda Dice.

4.1.4.7 Alteração na profundidade da rede

A última das modificações empregadas foi realizar testes com a rede mais ou menos densa “profunda”, por padrão a rede contém cinco estágios. Foram realizados testes com outros estágios a fim de analisar se a profundidade da rede influência no desempenho das segmentações, neste caso foram realizadas quatro edições na

profundidade da rede. A Tabela 18 descreve a taxa de acurácia, perda e o tempo de execução durante o treinamento da rede de acordo com a quantidade de estágios empregada.

Tabela 18 - Tabela comparativa dos resultados obtidos com as edições no número de estágios da rede.

V-Net Nº estágios	Acurácia (max)	Perda (min)	Tempo de Execução (horas)
2	0.91540	0.13778	~7,16
4	0.99720	0.00459	~16,75
5 (padrão)	0.99710	0.00351	~24,17
6	0.99710	0.00359	~29,70
7	0.99710	0.00358	~53,35

Fonte: Autoria própria

Conforme descrito na Tabela 18, a alteração na profundidade da rede para estágios maiores que o padrão obteve um aumento expressivo no custo computacional devido ao fato de que quanto mais profunda é a rede mais operações de convoluções, entre outras, serão aplicadas. Vale ressaltar que com valores acima de 7 estágios, devido a quantidade expressiva de operações seria necessária uma placa de vídeo com uma quantidade de memória superior à utilizada para suportar o treinamento da rede.

A Tabela 19 mostra os valores médios alcançados pelas métricas de desempenho entre os resultados obtidos pela segmentação de acordo com a profundidade da rede empregada.

Tabela 19 - Tabela comparativa dos resultados das métricas de desempenho em relação aos números de estágios da rede

V-Net Nº estágios	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
2	0.79584	0.92224	0.86899	0.90482	0.81605	0.70813
4	0.96605	0.98857	0.97814	0.98672	0.97158	0.94838
5 (padrão)	0.97608	0.99057	0.98194	0.98866	0.97876	0.95866
6	0.97379	0.98972	0.98256	0.98828	0.97777	0.95707
7	0.97666	0.99062	0.98169	0.98877	0.97894	0.95899

Fonte: Autoria própria

Pelos valores contidos na Tabela 19, percebe-se que a alteração na profundidade da rede levou a um aumento na performance das segmentações, de

modo que a rede contendo 7 estágios atingiu os maiores valores em quase todas as métricas. Neste sentido, esta alteração foi empregada na arquitetura da rede.

4.1.5 Aplicação de Pós-Processamento

Mesmo com as alterações na estrutura da rede previamente descritas na subseção 4.1.4, nem todas as segmentações geradas pela rede foram eficientes. Neste sentido, visando uma melhora nas imagens obtidas foi aplicado um pós-processamento.

Para a realização do pós-processamento foi criado um algoritmo que, dada uma imagem segmentada defeituosa, seja gerada uma nova elipse encaixando-a aos seus contornos e posteriormente preenchendo-a. Para a criação do algoritmo foi utilizada a linguagem de programação Python OpenCV devido a facilidade de programação e pelas inúmeras funções existentes para diversos tipos de problemas, dentre elas as funções *Canny()*, *findContours()* e a *fitEllipse()* foram essenciais para a realização do pós-processamento, as quais serão brevemente descritas a seguir:

- ***Canny()*:** esta função visa encontrar bordas em uma imagem aplicando o detector de bordas Canny desenvolvido por John F. Canny em 1986 (CANNY, 1986). Este algoritmo é considerado um dos melhores métodos de detecção de bordas, fornecendo uma detecção precisa dos contornos do objeto (OPENCV, 2019);
- ***findContours()*:** esta função recupera contornos da imagem binária usando o algoritmo proposto por Suzuki et al. (1985). Os contornos são uma ferramenta útil para análise de formas e detecção e reconhecimento de objetos (OPENCV, 2019);
- ***fitEllipse()*:** esta função utiliza o algoritmo proposto por Fitzgibbon e Fisher (1995) para calcular a elipse que melhor se encaixa (em um sentido de mínimos quadrados) a um melhor conjunto de pontos 2D encontrado. A função retorna um retângulo rotacionado no qual a elipse está inscrita (OPENCV, 2019).

A Figura 27 descreve o código construído para a realização do pós-processamento nas segmentações imprecisas.

Figura 27 - Algoritmo na linguagem Python para a realização do pós-processamento nas segmentações imprecisas

```

canny = cv.Canny(imagem, threshold, threshold * 2) #algoritmo de Canny

#encontrar os contornos
_, contorno, _ = cv.findContours(canny, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)

#encontra as elipses em cada contorno
minRect = [None]*len(contorno)
minEllipse = [None]*len(contorno)
for i, c in enumerate(contorno):
    minRect[i] = cv.minAreaRect(c)
    if c.shape[0] > 5:
        minEllipse[i] = cv.fitEllipse(c)

drawing = np.zeros((canny.shape[0], canny.shape[1], 1), dtype=np.uint8)

for i, c in enumerate(contorno):

    if c.shape[0] > 5:
        #prenche as elipses
        cv.ellipse(drawing, minEllipse[i], (255,255,255), -1)

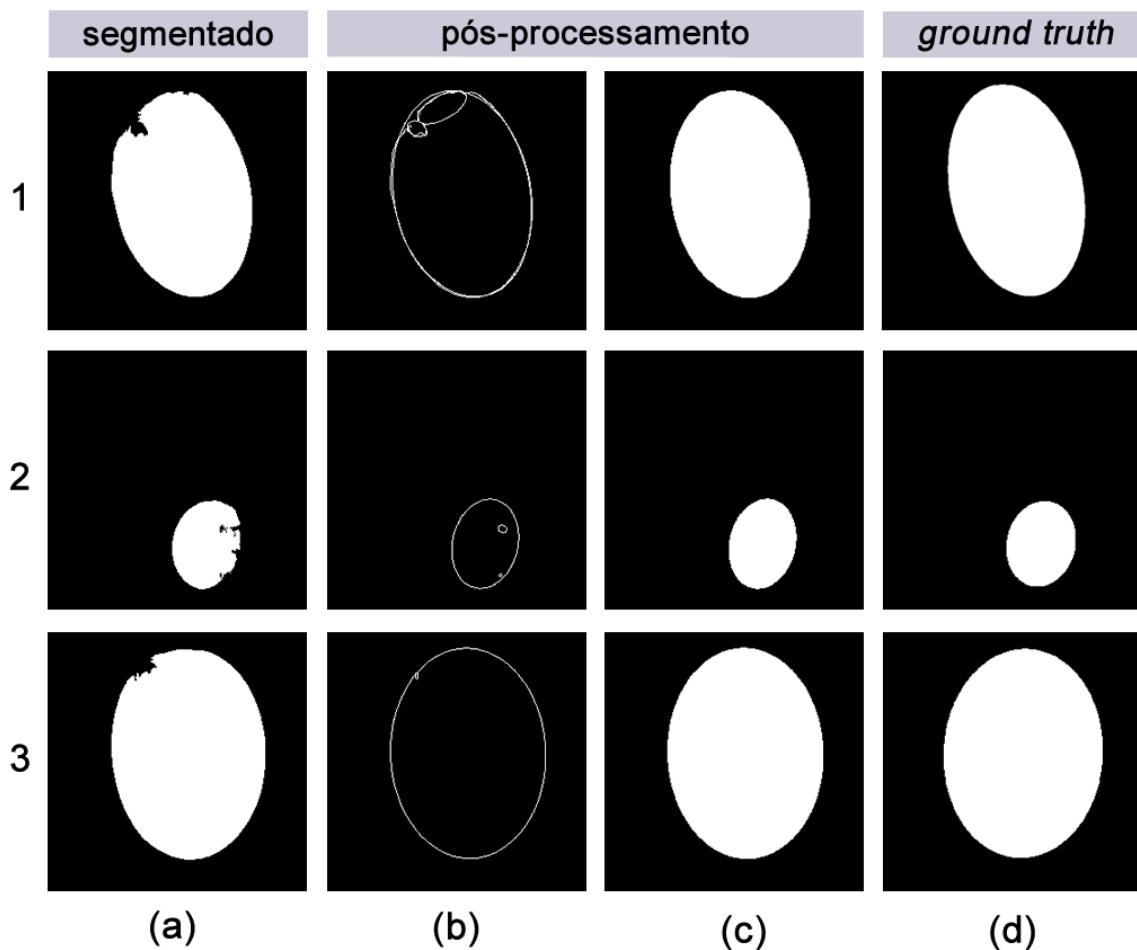
cv.imwrite('saida.png',drawing)

```

Fonte: Autoria própria

A Figura 28 ilustra os resultados do algoritmo aplicado a uma segmentação ineficiente obtida após o treinamento da rede.

Figura 28- Figura ilustrativa dos resultados obtidos após o algoritmo de pós-processamento



Fonte: Autoria própria

Conforme pode ser observado pela Figura 28, em (a) ilustra o resultado falho da segmentação obtida pela rede V-Net adaptada, em (b) representa o encaixe de uma nova elipse aos contornos da segmentação ineficiente, em (c) o resultado do preenchimento da nova elipse e em (d) ilustra a *ground truth* utilizada como métrica avaliativa.

A Tabela 20 realiza uma comparação entre as segmentações presentes na Figura 28 com e sem a técnica de pós-processamento.

Tabela 20- Tabela comparativa dos resultados entre as segmentações (spp - sem pós-processamento) e (cpp - com pós-processamento)

Figura 28	Precisão	Especificidade	Sensibilidade	Acurácia	Dice	Jaccard
1 (a) spp	0.97141	0.98591	0.94939	0.97366	0.96027	0.92358
1 (c) cpp	0.96452	0.98198	0.97137	0.97842	0.96794	0.93786
2 (a) spp	0.96616	0.99751	0.95385	0.99448	0.95996	0.92301
2 (c) cpp	0.96269	0.99716	0.98110	0.99605	0.97181	0.94516
3 (a) spp	0.98691	0.99182	0.96430	0.98108	0.97548	0.95213
3 (c) cpp	0.98683	0.99167	0.97540	0.98532	0.98108	0.96287

Fonte: Autoria própria

Conforme os valores descritos na Tabela 20, percebe-se que o emprego do pós-processamento melhorou as segmentações dados aos resultados obtidos pelas métricas de avaliação sensibilidade, acurácia, Dice e Jaccard em comparação com as *ground truth*, porém, perde-se a precisão da segmentação conforme os valores obtidos pelas métricas de precisão e especificidade. Isto ocorre devido ao fato de que é gerada uma elipse uniforme aos contornos, o que acaba por descartar alguns pontos de contorno válidos obtidos pela segmentação falha.

Visando uma comparação geral, a Tabela 21 apresenta o resultado das segmentações obtidas pela rede alterada conforme processos descritos na seção 4.1.4.

Tabela 21- Tabela comparativa dos resultados gerais entre as segmentações (spp - sem pós-processamento) e (cpp - com pós-processamento)

V-net Adaptada	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
spp	0.97666	0.99062	0.98169	0.98877	0.97894	0.95899
cpp	0.97677	0.99065	0.98207	0.98881	0.97918	0.95942

Fonte: Autoria própria

Conforme os resultados apresentados na Tabela 21, percebe-se que o uso de pós-processamento aprimorou o desempenho das segmentações proporcionado um aumento dos valores em todas as métricas abordadas.

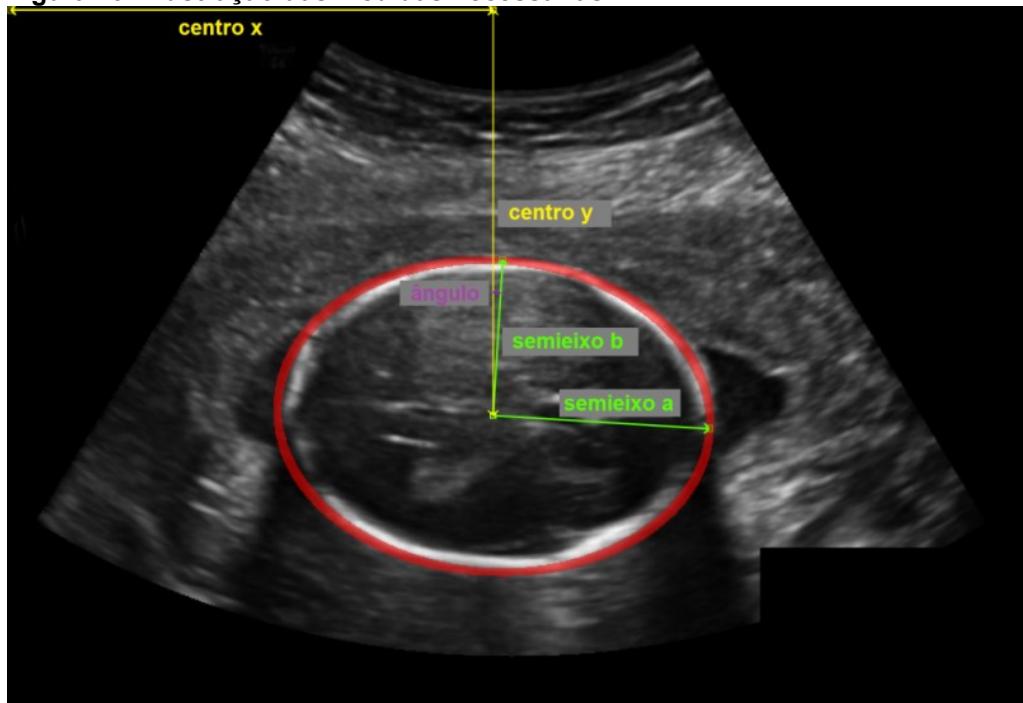
4.1.6 Implementação do Algoritmo para Cálculo do HC

Para a submissão dos resultados foi necessário calcular cinco medidas de cada elipse gerada pela segmentação da rede, sendo: centro x, centro y, semieixo a, semieixo b e ângulo, as quais serão brevemente descritas a seguir:

- **Centro x:** compreende o valor da distância em milímetros do *pixel* inicial presente no eixo x da imagem até o *pixel* que compõe o centro da elipse;
- **Centro y:** compreende o valor da distância em milímetros do *pixel* inicial do eixo y da imagem até o *pixel* que compõe o centro da elipse;
- **Semieixo a:** encontrado o centro da elipse, a medida semieixo a compreende o maior valor do raio desde a distância do centro da elipse até o ponto mais distante dele;
- **Semieixo b:** compreende o menor valor do raio, desde a distância do centro da elipse até o ponto mais próximo dele;
- **Ângulo:** compreende o valor do ângulo em radianos entre o vetor centro y e o vetor semieixo b.

A Figura 29 ilustra as cinco medidas descritas para o cálculo na elipse.

Figura 29 - Ilustração das medidas necessárias



Fonte: Adaptado de HEUVEL et al. (2018)

Conforme visto na Figura 29, a elipse na cor vermelha representa a segmentação do crânio fetal em uma imagem de ultrassonografia, as setas na cor verde representam o semieixo a e b, em amarelo o centro x e y, e na cor roxa o ângulo.

Para a construção do algoritmo para os cálculos da elipse foi escolhida a linguagem de programação Python OpenCV, sendo utilizadas a função *findContours()* e a função *fitEllipse()* previamente descrita na subseção 4.1.5.

A Figura 30 descreve o código construído para o cálculo do HC.

Figura 30 – Algoritmo na linguagem Python para o cálculo dos valores da elipse
Author: Everton Leonardo Skeika

```
imagem, valorpixel = np.loadtxt('test_set_pixel_size.csv',
delimiter = ',', unpack = True, dtype = 'str')
i=0
for i in range(355):

    img = cv2.imread(imagem[i],0)
    ret,thresh = cv2.threshold(img,127,255,0)
    im2, contours, hierarchy = cv2.findContours(thresh,
cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    cnt = contours[0]
    (x,y), (MA,ma), angle = cv2.fitEllipse(cnt)

    x = x *float (valorpixel[i])
    y = y *float (valorpixel[i])
    MA = (MA *float (valorpixel[i]))/2
    ma = (ma *float (valorpixel[i]))/2
    angle = math.cos(math.radians(angle))

    with open('hc_result.csv', 'at', newline='') as csvfile:
        spamwriter = csv.writer(csvfile , delimiter=',',
        quotechar='|', quoting=csv.QUOTE_MINIMAL)
        spamwriter.writerow([imagem[i],round(x,9),
        round(y,9), round(MA,9),
        round(ma,9), round(angle,9)])
csvfile.close()
```

Fonte: Autoria própria

Após a execução do código, gera-se o arquivo no formato CSV com todas as medidas sob os padrões necessários para a submissão.

4.1.7 Submissão ao desafio HC18

Após todas as alterações realizadas na rede, a mesma foi treinada para a base de imagens completa. Neste caso foi alterado o número de épocas e passos da rede para um melhor treinamento. Após o treinamento as imagens segmentadas resultantes foram passadas pela etapa de pré-processamento e posteriormente suas resoluções foram convertidas para o seu valor original de 800 por 540 *pixels*, para então ser aplicado o algoritmo para o cálculo dos valores da circunferência da cabeça do feto segmentada.

4.2 CONSIDERAÇÕES FINAIS

Neste capítulo apresentou-se a metodologia proposta necessária para a execução do presente trabalho. Inicialmente foi realizado um estudo sobre o desafio HC18, posteriormente com a aquisição do *dataset* utilizado foi aplicado técnicas de pré-processamento no mesmo. Também foram descritas todas as ferramentas necessárias para a execução da metodologia proposta, bem como a configuração de máquina utilizada.

Posteriormente, com a aquisição da rede empregada como base, foram realizadas diversas alterações em sua estrutura para evitar o problema de *overfitting*. Algumas das alterações realizadas foram: a adaptação da técnica de *batch normalization*; em cada camada de convolução empregada na rede, foi alterada a função de PreLu para ELU; na camada de ativação final existente na rede foi realizada a substituição de *sigmoid* para *hard_sigmoid*; foi avaliada a técnica de *dropout*; foram realizados vários testes com outras funções de perda existentes na literatura e, conforme os resultados, foi alterada para função de perda Tversky *loss*. Dado o *dataset* utilizado, foram realizadas alterações na técnica de *data augmentation* e no aumento do *batch* visando uma maior generalização das imagens. E, por último, foram realizados testes com a rede menos densa "profunda" e mais densa, com as análises dos resultados foi visto que o aumento na profundidade da rede obteve impactos satisfatórios na segmentação das imagens.

Embora todas as edições realizadas na rede, a mesma retornou algumas imagens mal segmentadas. Neste sentido, foi aplicada uma etapa de pós-processamento, ou seja, foi criado um algoritmo na linguagem Python OpenCV para

anexar uma elipse uniforme nas segmentações ineficientes e posteriormente a mesma é preenchida. Testes realizados constataram que o emprego da técnica obteve uma melhora na acurácia geral das segmentações.

Por último, foi projetado um algoritmo na linguagem Python OpenCV em que cada imagem contendo o crânio fetal segmentado seja extraída as medidas necessárias para posterior submissão ao desafio.

No próximo capítulo serão abordados os resultados obtidos pela metodologia proposta, além de discussões sobre os problemas enfrentados ao decorrer do trabalho.

5 RESULTADOS E DISCUSSÕES

Neste capítulo são apresentados os resultados obtidos, além de discussões sobre os principais problemas encontrados durante a execução da metodologia proposta no Capítulo 4.

Com a submissão do arquivo CSV contendo todas as cinco medidas da cabeça do feto (conforme processo descrito na subseção 4.1.7), pela avaliação automática com as métricas do desafio foi obtido a 46^a posição geral com a rede V-Net adaptada (posição referente ao mês de julho de 2019).

A fim de uma análise comparativa, a Tabela 22 apresenta os valores obtidos pelo primeiro colocado no desafio com os valores da rede V-Net adaptada, e a V-Net base.

Tabela 22- Tabela comparativa entre o 1º colocado no desafio com os resultados obtidos pela V-Net adaptada e a V-Net base

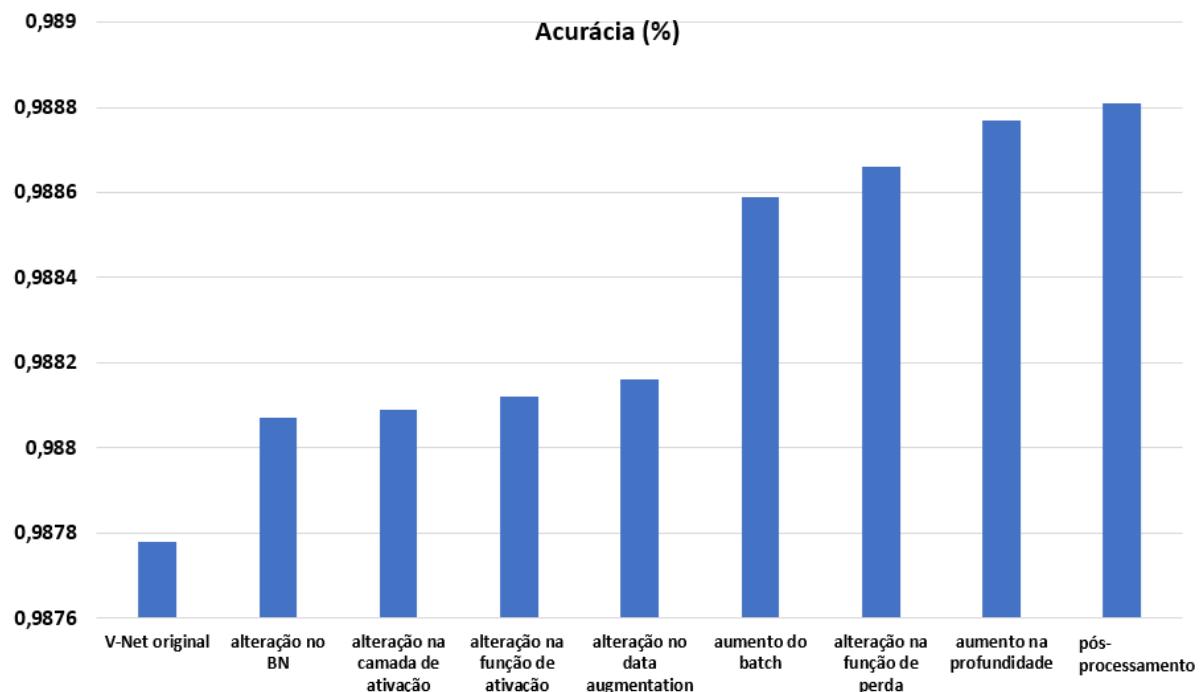
Posição HC18	<i>Abs difference</i> (mm) (média) \pm std	Dice (%) (média) \pm std	<i>Mean difference</i> (mm) \pm std	Distância de Haussdorf (mm) (média) \pm std
1 ^a	1.72 \pm 1.55	98.10 \pm 0.93	-0.05 \pm 2.31	1.17 \pm 0.64
46 ^a	1.89 \pm 1.67	90.62 \pm 2.45	0.74 \pm 2.41	6.65 \pm 3.24
312 ^a	2.18 \pm 2.37	90.63 \pm 2.45	0.83 \pm 3.12	6.67 \pm 3.22

Fonte: Autoria própria

Embora o resultado obtido pelo trabalho proposto não alcançar as primeiras posições, vale ressaltar que o desafio conta com mais de 760 submissões (quantidade referente ao mês de julho de 2019). O resultado com a V-Net base neste desafio ficou na 312^a posição. Neste sentido, o emprego da metodologia proposta, bem como as alterações na arquitetura da rede foram responsáveis pelo ganho de desempenho.

A Figura 31 ilustra o gráfico de evolução da rede utilizada como base após cada processo abordado na subseção 4.1.4 do presente trabalho.

Figura 31 - Gráfico de evolução do valor médio de acurácia de acordo com as alterações impostas na arquitetura da rede



Fonte: Autoria própria

Conforme o gráfico da acurácia ilustrado pela Figura 31, percebe-se uma boa evolução no desempenho da rede ao passar de cada processo de modificação em sua arquitetura. O aumento no conjunto de imagens sintéticas gerados pelo *data augmentation* e o aumento da profundidade da rede tiveram uma grande contribuição em sua performance.

No entanto, um dos fatores limitantes na melhora do desempenho foi a quantidade de memória de vídeo utilizada, pois o aumento do *batch* e da profundidade da rede acarretaram por utilizar todos os 6 GB de memória dedicada da placa de vídeo e, como consequência, impedindo o emprego de outras técnicas na arquitetura da rede. Este fator também impediu a utilização da rede U-Net como base, conforme demonstrado a seguir.

Visando uma comparação foi empregada a rede U-Net 2D disponível no sitio²⁰. Assim como a V-Net utilizada como base, esta rede foi construída por meio da biblioteca Keras, sendo utilizado o Tensorflow como *backend* com cuDNN para o processamento da rede via GPU. A rede foi treinada com o mesmo *dataset* e a mesma quantidade de passos e épocas.

²⁰ <https://github.com/zhixuhao/unet>

A Tabela 23 apresenta uma comparação entre a rede U-Net, a rede V-Net base e V-Net adaptada entre os resultados de acurácia, perda e o tempo de execução em relação ao treinamento de cada rede.

Tabela 23- Tabela comparativa da Acurácia, Perda e o Tempo de execução entre a rede U-Net base, V-Net base e a V-Net adaptada

Rede	Passos	Épocas	Acurácia (máxima)	Perda (mínima)	Tempo de Execução (horas)
U-Net base	1000	300	0,99790	0,00480	~33,10
V-Net base	1000	300	0,99210	0,01170	~16,25
V-Net aptada	1000	300	0.99750	0.00399	~54,35

Fonte: Autoria própria

Vale ressaltar que a rede U-Net é extremamente utilizada na literatura obtendo bons resultados em diversos tipos de segmentações, além de ter diversos trabalhos empregando outras técnicas em sua arquitetura de rede. No entanto, a rede U-Net utilizada como base não obteve um bom resultado nas segmentações dos crânios fetais conforme analisado pelos valores das métricas contidos na Tabela 24.

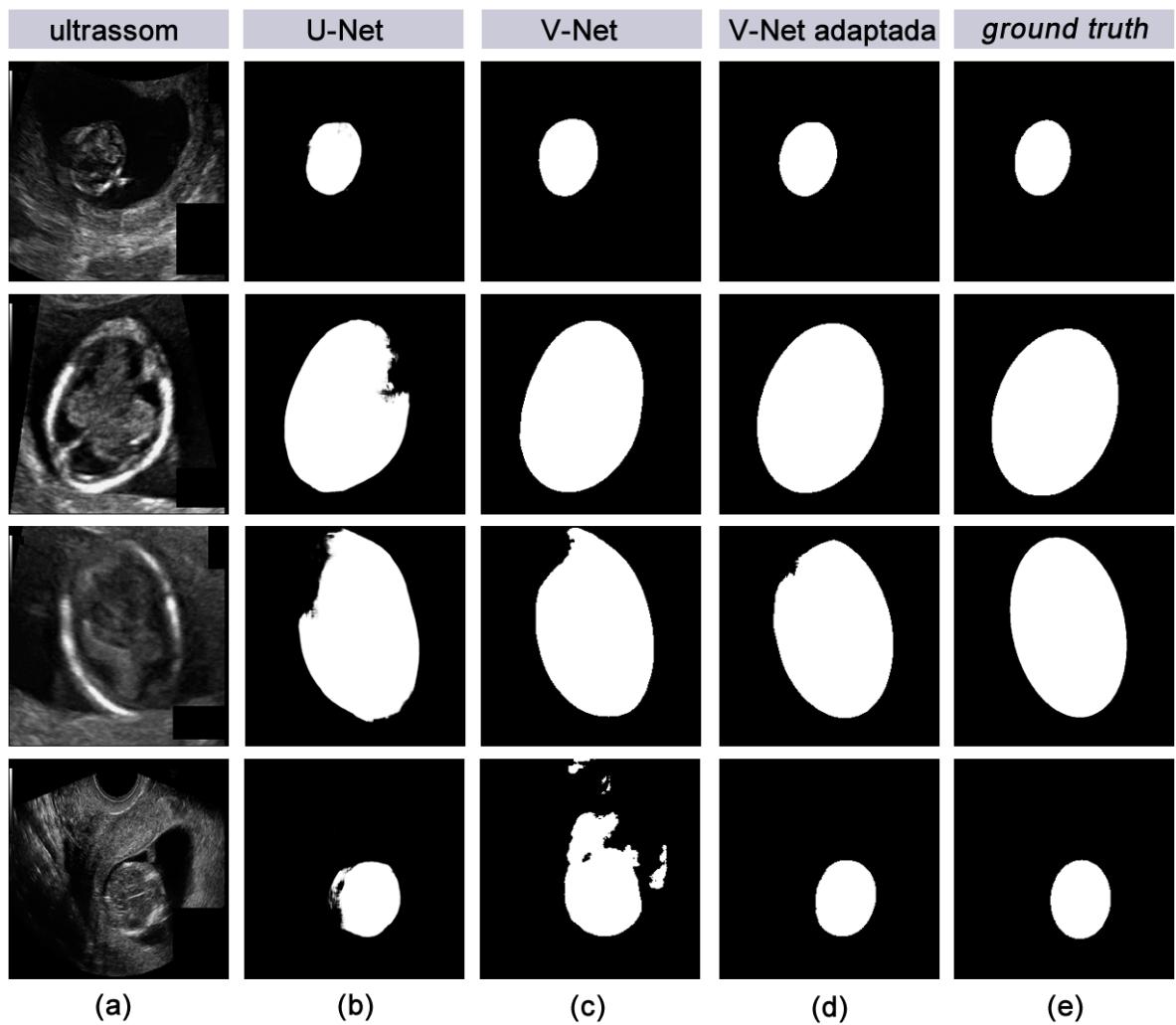
Tabela 24- Tabela comparativa dos resultados das métricas de desempenho entre a rede U-Net base, V-Net base e a V-Net adaptada

Rede	Precisão (média)	Especificidade (média)	Sensibilidade (média)	Acurácia (média)	Dice (média)	Jaccard (média)
U-Net original	0.97603	0.99038	0.97040	0.98587	0.97256	0.94770
V-Net original	0.97699	0.99200	0.97674	0.98778	0.97651	0.95447
V-Net aptada	0.97957	0.99176	0.97890	0.98868	0.97902	0.95909

Fonte: Autoria própria

Objetivando uma análise visual, a Figura 32 ilustra alguns dos resultados obtidos pela segmentação da rede U-Net, V-Net e V-Net adaptada, todas sem a aplicação de pós-processamento.

Figura 32 - Imagem comparativa entre alguns dos resultados obtidos pela rede U-Net, V-Net utilizada como base e a V-Net adaptada



Fonte: Autoria própria

Neste trabalho, a rede U-Net não foi escolhida devido ao seu alto custo computacional e pela maior quantidade de memória de vídeo necessária. Conforme demonstrado na Tabela 23 seu um tempo de execução em horas é praticamente o dobro comparado com o tempo da rede V-Net utilizada como base.

Uma das principais diferenças entre as duas redes é que a técnica usada na rede U-Net para reduzir as resoluções entre os estágios de contração (lado esquerdo da rede) é a técnica de *maxpooling*. No caso da V-Net, como esta foi projetada para imagens tridimensionais, as resoluções são diminuídas através de uma convolução com *stride* de 2. Tal estratégia acaba por utilizar menos memória, já que a sua escassez é um grande problema para a maioria dos modelos 3D.

Outro ponto que foi descartado na metodologia foi a inclusão de algoritmos para pré-processamento nas imagens presentes no *dataset*, algumas técnicas foram

empregadas para tentar realçar as bordas dos crânios fetais ou melhorar a qualidade das imagens. Por exemplo, o uso de *histogram matching*, equalização de histograma, histograma adaptativo, filtro da média e filtro *high boost*. Todas as técnicas citadas pioraram drasticamente o desempenho do treinamento da rede, e consequentemente as segmentações.

Outro ponto negativo foi que o *dataset* utilizado para a realização da metodologia proposta é considerado recente, ou seja, não foram encontrados trabalhos na literatura que o utilizaram. Isto impossibilitou uma comparação de desempenho com a rede V-Net adaptada.

5.1 CONSIDERAÇÕES FINAIS

Neste capítulo foram abordados os resultados obtidos pela metodologia proposta ao desafio HC18. Conforme os resultados observou-se uma melhora no desempenho da rede V-Net após sua adaptação. Além disso, foi realizada uma comparação sucinta entre a rede U-Net e V-Net, nela observou-se que o custo computacional da U-Net é superior. Também foram relatadas algumas das dificuldades encontradas durante a execução da metodologia, além de uma breve comparação com trabalhos relacionados. No próximo capítulo será abordada a conclusão do presente trabalho e algumas sugestões para trabalhos futuros.

6 CONCLUSÃO

O uso da ultrassonografia é imprescindível para a medição da biometria fetal durante o processo de gestação. No entanto, a avaliação manual das medidas é subjetiva e depende muito da experiência do avaliador. Neste sentido, torna-se necessário o emprego de técnicas computacionais para obtenção de melhores resultados.

Dado a problemática, esta dissertação apresentou a adaptação de um método computacional baseado em Aprendizado Profundo para a segmentação de crânios fetais em imagens de ultrassom bidimensionais. Deste modo, foi proposta uma metodologia para a adaptação de uma Rede Neural Completamente Convolucional para a realização do problema disposto.

Para a realização da metodologia foi utilizada como base a rede V-Net que foi projetada para segmentação de imagens em 3D. Foram realizadas várias alterações na arquitetura para aumentar o desempenho e diminuir o problema de *overfitting*, como por exemplo: a alteração do uso de *batch normalization*, a alteração nas funções de ativação em cada convolução da rede de PreLu para ELU.

Na última camada de ativação da rede foi realizada a alteração da função *sigmoid* para *hard_sigmoid*, foi editado o uso da técnica de *dropout*, além do emprego de uma recente função de perda ao invés da dice *loss*. Devido à baixa quantidade de imagens do *dataset* também foi realizado alterações na técnica de *data augmentation* sendo elevado o número de *batch* gerados para a disponibilização de uma maior quantidade de dados sintéticos para o treinamento da rede. Além disso, a profundidade da rede foi aumentada até o limite suportado pela placa de vídeo utilizada. Posteriormente, com algumas das segmentações imprecisas, foi criado um algoritmo para realizar o pós-processamento e melhorar a qualidade das mesmas. Por último foi criado um algoritmo para o cálculo das medidas da circunferência da cabeça fetal.

Os resultados obtidos pela metodologia proposta constataram que é possível a adaptação da arquitetura da Rede Neural Completamente Convolucional V-Net que foi projetada para segmentação de imagens tridimensionais seja utilizada para o emprego em imagens bidimensionais. No entanto, conforme resultados obtidos pela classificação no desafio percebe-se que ainda há a necessidade de aprimorar a rede utilizando outras técnicas para uma melhor segmentação dos crânios fetais.

6.1 TRABALHOS FUTUROS

O trabalho realizado até aqui marca um ponto de partida em uma série de trabalhos futuros que podem ser feitos a fim de explorar ainda mais a arquitetura da rede V-Net para a segmentação de crânios fetais em imagens de ultrassom bidimensionais. A seguir, algumas sugestões para trabalhos futuros são listadas:

- Aumentar o conjunto de treinamento agregando outros *datasets* de crânios fetais;
- Treinar a rede com o *dataset* com resolução de imagem original, pois isto eliminaria a necessidade de redimensioná-la e, como consequência, não haveria deformações e perda de qualidade;
- Utilizar uma técnica mais precisa na etapa de pós-processamento para consertar as segmentações ineficientes;
- Adaptar a rede para realizar o seu treinamento seguindo os conceitos de validação cruzada;
- Aplicar outras técnicas disponíveis na literatura na arquitetura da rede V-Net, dentre algumas dessas podemos citar: a agregação de *Recurrent Residual Convolution* conforme descrito no trabalho de Alom et al. (2018), a inclusão de *Attention Gates* pelo trabalho proposto por Oktay et al. (2018), o uso de *Pyramid Dilated Convolution Unit* conforme apresentado no trabalho de Zhang et al. (2018).

REFERÊNCIAS

- ABADI, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. In: **12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)**. Savannah, GA. USENIX Association, p. 265–283, 2016.
- ABUHAMAD, A.; et al. **Ultrasound in obstetrics and gynecology**: A practical approach. 1. ed. 2014.
- AGGARWAL, C. C.; HINNEBURG, A; KEIM, D. A. On the surprising behavior of distance metrics in high dimensional space. In: **International conference on database theory**. Springer, Berlin, Heidelberg, p. 420-434, 2001.
- AL-KARMI, A. M.; DINNO, M. A.; STOLTZ, D. A.; CRUM, L. A.; MATTHEWS, J.C. Calcium and effects of ultrasound on frog skin. **Ultrasound in Med. & Biol.**, 20(1):73-81, 1994.
- ALMEIDA C. F. P. **Mapeamento da Distribuição Populacional Através da Detecção de Áreas Edificadas em Imagens de Regiões Heterogêneas do Google Earth Usando Deep Learning**, Tese de Doutorado, Programa de Pós-graduação em Informática da PUC-Rio. Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2017.
- ALOM, M. Z.; et al. Recurrent residual convolutional neural network based on u-net (R2U-net) for medical image segmentation. **arXiv preprint arXiv:1802.06955**, 2018.
- ALPERT, S.; et al. Image segmentation by probabilistic bottom-up aggregation and cue integration. **IEEE transactions on pattern analysis and machine intelligence**, v. 34, n. 2, p. 315-327, 2012.
- ANACONDA INC. **Anaconda Documentation: Release 2.0**. 2018. Disponível em:<<https://readthedocs.com/projects/continuumio-docs/downloads/pdf/latest/>>. Acesso em: 8 dez. 2018.
- APPLE. **Performing Convolution Operations**. 2016. Disponível em:<<https://developer.apple.com/library/content/documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html>>. Acesso em: 14 nov. 2018.
- ARAUJO, A. F. D.; et al. Análise e caracterização de lesões de pele para auxílio ao diagnóstico médico. **Avanços em Visão Computacional**, 2012.
- ARAÚJO, F. H.D.; et al. **Redes Neurais Convolucionais com Tensorflow: Teoria e Prática**. SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. III Escola Regional de Informática do Piauí. Livro Anais-Artigos e Minicursos, v. 1, p. 382-406, 2017.
- AREL, I.; et al. Deep machine learning-a new frontier in artificial intelligence research. **IEEE computational intelligence magazine**, v. 5, n. 4, p. 13-18, 2010.

- BARNARD, R. W.; PEARCE, K; SCHOVANEC, L. Inequalities for the perimeter of an ellipse. **Journal of mathematical analysis and applications**, v. 260, n. 2, p. 295-306, 2001.
- BECKER, C. et al. Supervised feature learning for curvilinear structure segmentation. In: **International conference on medical image computing and computer-assisted intervention**. Springer, Berlin, Heidelberg, 2013. p. 526-533.
- BENGIO, Y.; et al. Learning deep architectures for AI. **Foundations and trends in Machine Learning**, v. 2, n. 1, p. 1-127, 2009.
- BENNETT, N.; BURRIDGE, R.; SAITO, N. A method to detect and characterize ellipses using the Hough transform. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 21, n. 7, p. 652-657, 1999.
- BERMAN, M.; RANNEN T., A.; BLASCHKO, M., B. The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In: **IEEE/CVF Conference on Computer Vision and Pattern Recognition**. Salt Lake City, UT, p. 4413-4421, 2018.
- BEUCHER, S.; MEYER, F. **Image Analysis and mathematical Morphology**. London: London Academic Press. 1982.
- BOYKOV, Y. Y.; JOLLY, M. P. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In: **Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001**. Vancouver, BC, Canada, p. 105-112 vol.1, 2001.
- BRIDGE, C. P.; IOANNOU, C.; NOBLE, J. A. Automated annotation and quantitative description of ultrasound videos of the fetal heart. **Medical image analysis**, v. 36, p. 147-161, 2017.
- CANNY, J. A computational approach to edge detection. In: **Readings in computer vision**. Morgan Kaufmann, p. 184-203, 1987.
- CARNEIRO, G.; et al. Detection of fetal anomalies from ultrasound images using a constrained probabilistic boosting tree. **IEEE Trans. Medical Imaging**, v. 27, n. 9, p. 1342-1355, 2008.
- CHAURASIA, A.; CULURCIELLO, E. Linknet: Exploiting encoder representations for efficient semantic segmentation. In: **IEEE Visual Communications and Image Processing (VCIP)**. IEEE, p. 1-4, 2017.
- CHOLLET, F. **Deep learning with python**. Manning Publications Co., 2017.
- CHOW, C. K.; KANEKO, T. Automatic boundary detection of the left ventricle from cineangiograms. **Computers and biomedical research**, v. 5, n. 4, p. 388-410, 1972.
- CRÓSTA, A. P. **Processamento Digital de Imagens de Sensoriamento Remoto**, Campinas, SP, UNICAMP, ed. rev., 1993.

DAVID, O. E.; NETANYAHU, N. S. DeepPainter: painter classification using deep convolutional autoencoders. In: **International Conference on Artificial Neural Networks**. Springer, Cham, p. 20-28, 2016.

DALVI, R. F. **Detecção e Classificação de Câncer a partir de Mamografias Digitalizadas e Redes Neurais Convolucionais**. 106f. (Dissertação de Mestrado) Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo, Espírito Santo, 2018.

DERTAT, A. **Applied Deep Learning - Part 4: Convolutional Neural Networks**. 8 nov. 2017. Disponível em: < <https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2> >. Acesso em: 15 nov. 2018.

DICE, L. R. Measures of the amount of ecologic association between species. **Ecology**, v. 26, n. 3, p. 297-302, 1945.

DUMOULIN, V.; VISIN, F. A guide to convolution arithmetic for deep learning. **arXiv preprint arXiv:1603.07285**, 2016.

FITZGIBBON, A. W.; FISHER, R. B. A Buyer's Guide to Conic Fitting. **Proc.5th British Machine Vision Conference**, Birmingham, pp. 513-522, 1995.

GRAND-CHALLENGES©. **Automated measurement of fetal head circumference**. 2018. Disponível em: < <https://hc18.grand-challenge.org/> >. Acesso em: 15 jan. 2019.

GINNEKEN, V.B.; et al. Active shape model segmentation with optimal features. **IEEE transactions on medical imaging**, v. 21, n. 8, p. 924-933, 2002.

GIRSHICK, R.; et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: **IEEE Conference on Computer Vision and Pattern Recognition**. Columbus, OH, p. 580-587, 2014.

GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing**. 2. ed. New Jersey: Pearson Prentice Hall, 2001.

GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing**, 3. ed. New Jersey: Pearson Prentice Hall, 2008.

GOODFELLOW, I.; et al. **Deep learning**. 1. ed. Cambridge: MIT press, 2016.

GUARIGLIA S. N. **Breve história da ultrassonografia**. 2016. Disponível em: < <http://daimagem.com.br/site/breve-historia-da-ultrassonografia/> >. Acesso em: 1 nov. 2018.

HAFEMANN, L. G. **An analysis of deep neural networks for texture classification**. 89 f. Dissertação (mestrado) - Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná, Curitiba, 2014.

HAN, D.; KIM, J.; KIM, J. Deep pyramidal residual networks. In: **EEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Honolulu, HI, p. 6307-6315, 2017.

HARALICK, R. M.; SHAPIRO, L. G. Image segmentation techniques. In: **Applications of Artificial Intelligence II**. International Society for Optics and Photonics, p. 2-10, 1985.

HAUFF, P.; REINHARDT, M.; FOSTER, S. Ultrasound basics. In: **Molecular Imaging I**. Springer, Berlin, Heidelberg, p. 91-107, 2008.

HE, K.; et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: **IEEE International Conference on Computer Vision (ICCV)**. p. 1026-1034, 2015.

HELMSTAEDTER, M.; et al. Connectomic reconstruction of the inner plexiform layer in the mouse retina. **Nature**, v. 500, n. 7461, p. 168, 2013.

HEUVEL, V. D. T. L.; et al. (2018). **Automated measurement of fetal head circumference [Data set]**. Zenodo. <http://doi.org/10.5281/zenodo.1327317>
Acesso em: 5 set. 2018.

HINTON, G. E.; et al. Improving neural networks by preventing co-adaptation of feature detectors. **arXiv preprint arXiv:1207.0580**, 2012.

HUTTENLOCHER, D.; KLANDERMAN, G.; RUCKLIDGE, W. Comparing Images Using the Hausdorff Distance. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 15, n. 9, p. 850–863, 1993.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. **arXiv preprint arXiv:1502.03167**, 2015.

JARDIM, S. M.; FIGUEIREDO, M. A. Segmentation of fetal ultrasound images. **Ultrasound in medicine & biology**, v. 31, n. 2, p. 243-250, 2005.

JANG, J.; et al. Automatic Estimation of Fetal Abdominal Circumference from Ultrasound Images. **IEEE journal of biomedical and health informatics**, v. 22, n. 5, p. 1512-1520, 2018.

KARPATY, A. **Convolutional neural networks for visual recognition**. 2017.
Disponível em:< <http://cs231n.github.io/convolutional-networks/>>. Acesso em: 14 nov. 2018.

KARN, U. **An Intuitive Explanation of Convolutional Neural Networks**. 2016.
Disponível em: <<https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets>>. Acesso em: 17 nov. 2018.

KREMKAU, F. W. **Diagnóstico por ultra-som: princípios e instrumentos**. 4. ed. São Paulo: ArtMed, 1996.

- KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: **Advances in neural information processing systems**. USA: Curran Associates Inc. p. 1097–1105. 2012.
- LECUN, Y.; et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278-2324, 1998.
- LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015.
- LECUN, Y.; KAVUKCUOGLU, K.; FARABET, C. Convolutional networks and applications in vision. In: **IEEE International Symposium on Circuits and Systems (ISCAS)**. Paris, p. 253-256. 2010.
- LEE, J. **How to do Semantic Segmentation using Deep learning**. 2018. Disponível em: <<https://medium.com/nanonets/how-to-do-image-segmentation-using-deep-learning-c673cc5862ef>>. Acesso em: 15 nov. 2018.
- LEE, H.; et al. Fully automated deep learning system for bone age assessment. **Journal of digital imaging**, v. 30, n. 4, p. 427-441, 2017.
- LEE, C.; GALLAGHER, P. W.; TU, Z. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In: **Artificial Intelligence and Statistics**. 2016. p. 464-472.
- LEVAL. **Aparelho de ultrassom**. 2006. Disponível em: <<https://commons.wikimedia.org/wiki/File:Aparelhodeultrassom.jpg>>. Acesso em: 1 nov. 2018.
- LI, Y.; et al. Automatic fetal body and amniotic fluid segmentation from fetal ultrasound images by encoder-decoder network with inner layers. In: **Engineering in Medicine and Biology Society (EMBC), 2017 39th Annual International Conference of the IEEE**. IEEE, p. 1485-1488, 2017.
- LIN, T.; et al. Focal loss for dense object detection. In: **IEEE International Conference on Computer Vision (ICCV)**. Venice, p. 2999-3007, 2017.
- LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Boston, MA, p. 3431-3440, 2015.
- LU, W.; TAN, J.; FLOYD, R. Automated fetal head detection and measurement in ultrasound images by iterative randomized Hough transform. **Ultrasound in medicine & biology**, v. 31, n. 7, p. 929-936, 2005.
- MA, J.; et al. Deep neural nets as a method for quantitative structure–activity relationships. **Journal of chemical information and modeling**, v. 55, n. 2, p. 263-274, 2015.

MACIEL, C. D.; PEREIRA, W. C. A. Modelagem e processamento de speckle em imagem ultra-sônica: uma revisão. **RBE-Caderno de Engenharia Biomédica**, v. 13, n. 1, p. 71-89, 1997.

MARQUES FILHO, O.; VIEIRA NETO, H. **Processamento Digital de Imagens**, Rio de Janeiro: Brasport, 1999.

MILLETARI, F.; NAVAB, N.; AHMADI, S. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: **Fourth International Conference on 3D Vision (3DV)**. Stanford, CA, p. 565-571, 2016.

MOIGNE, L. J.; TILTON, J. C. Refining image segmentation by integration of edge and region data. **IEEE Transactions on Geoscience and Remote Sensing**, v. 33, n. 3, p. 605-615, 1995.

NASCIMENTO, P. P. M. **Applications of deep learning techniques on NILM**. 85f. Dissertação Programa de Pós-graduação em Engenharia Elétrica, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro 2016.

NVIDIA DEVELOPER. **NVIDIA cuDNN**. 2018. Disponível em: <<https://developer.nvidia.com/cudnn>>. Acesso em: 13 dez. 2018.

OBGYN W. G. **CRL Crown rump length 12 weeks ecografia Dr. Wolfgang Moroder**. 2006. Disponível em: <https://commons.wikimedia.org/wiki/File:CRL_Crown_rump_length_12_weeks_ecografia_Dr._Wolfgang_Moroder.jpg>. Acesso em: 1 nov. 2018.

OKTAY, O.; et al. Attention u-net: Learning where to look for the pancreas. **arXiv preprint arXiv:1804.03999**, 2018.

OPENCV. **Structural Analysis and Shape Descriptors**. Disponível em: <https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html>. Acesso em: 2 fev. 2019.

OPENCV. **OpenCV manual online**. Disponível em: <<https://docs.opencv.org/>>. Acesso em: 15 jul. 2019.

PEDRINI, H.; SCHWARTZ, W. **Análise de Imagens Digitais: Princípios, algoritmos e Aplicações**, 1. ed. São Paulo :Thomson Learning, 2008.

PINTRO, F. **Análise Morfológica dos Eritrócitos nas Doenças Hematológicas através da Aplicação de Redes Neurais Artificiais no Processamento de Imagens Digitais**. 98 f. Dissertação (Mestrado em Modelagem Matemática) – Departamento de Física, Estatística e Matemática, Universidade Regional do Noroeste do Estado do Rio Grande do Sul, Ijuí, 2008.

POWERS, D. M. W. Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. **J. Mach. Learn. Technol.**, v. 2, n. 1, p. 37-63, 2011.

RAMOS, P. Z. A. **Segmentação de imagens ultrassonográficas para detecção de nódulos.** 138 f. Tese de Doutorado, Escola de Engenharia de São Carlos Departamento de Engenharia Elétrica, Universidade de São Paulo, São Paulo, 2010.

RAVISHANKAR, H.; et al. Hybrid approach for automatic segmentation of fetal abdomen from ultrasound images using deep learning. In: **IEEE 13th International Symposium on Biomedical Imaging (ISBI).** Prague, p. 779-782, 2016.

REN, S.; et al. Faster r-cnn: Towards real-time object detection with region proposal networks. In: **Advances in neural information processing systems.** p. 91-99, 2015.

ROBIN, X.; et al. pROC: an open-source package for R and S+ to analyze and compare ROC curves. **BMC bioinformatics**, v. 12, n. 1, p. 77, 2011.

RONNEBERGER, O.; FISCHER, P; BROX, T. U-Net: Convolutional networks for biomedical image segmentation. In: **International Conference on Medical image computing and computer-assisted intervention.** Springer, Cham, p. 234-241, 2015.

ROMAGUERA, L. V. **Segmentação do miocárdio em imagens de MRI cardíaca utilizando redes neurais convolutivas.** 154 f. Dissertação (Mestrado) - Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal do Amazonas. Manaus, 2017.

ROSSUM, G. **Python Reference Manual.** Amsterdam, The Netherlands, The Netherlands, 1995.

RUEDA, S.; et al. Evaluation and comparison of current fetal ultrasound image segmentation methods for biometric measurements: a grand challenge. **IEEE Transactions on medical imaging**, v. 33, n. 4, p. 797-813, 2014.

SAFARZADEH, K. G. **Segmentation Validation Framework.** 2013. Department of Biomedical Engineering, Linköping University, 2013.

SALEHI, S. S. M.; ERDOGMUS, D.; GHOLIPOUR, A. Tversky loss function for image segmentation using 3D fully convolutional deep networks. In: **International Workshop on Machine Learning in Medical Imaging.** Springer, Cham, p. 379-387, 2017.

SANCHES, I. J. **Sobreposição de Imagens de Termografia e Ressonância Magnética: Uma Nova Modalidade de Imagem Médica Tridimensional,** Tese de Doutorado, Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná, Curitiba, 2009.

SANDERS, R. C.; JAMES, A. E. **The principles and practice of ultrasonography in obstetrics and gynecology.** 1. ed. United States: Radiology and Nuclear Medicine, 1985.

SANTOS, A. D. P. **Controle de qualidade cartográfica: metodologias para a avaliação da acurácia posicional em dados espaciais.** Tese de Doutorado, Programa de Pós-graduação em Engenharia Civil, Universidade Federal de Viçosa. Viçosa, 2015.

SEWAK, M.; KARIM, M. R.; PUJARI, P. **Practical Convolutional Neural Networks: Implement advanced deep learning models using Python.** Packt Publishing Ltd, 2018.

SGUARIO, M. L. **Uma nova abordagem do método Level Set baseada em conhecimento a priori da forma.** Tese de Doutorado, Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2015.

SHATTUCK, D. W.; et al. Magnetic resonance image tissue classification using a partial volume model. **NeuroImage**, v. 13, n. 5, p. 856-876, 2001.

SINCLAIR, M.; et al. Human-level performance on automatic head biometrics in fetal ultrasound using fully convolutional neural networks. **arXiv preprint arXiv:1804.09102**, 2018.

SINGHAL, H. **Convolutional Neural Network with TensorFlow implementation.** 2017. Disponível em: < <https://medium.com/data-science-group-iitr/building-a-convolutional-neural-network-in-python-with-tensorflow-d251c3ca8117>>. Acesso em: 15 nov. 2018.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014.

SOBEL, I.; FELDMAN, G. A 3x3 isotropic gradient operator for image processing. **A talk at the Stanford Artificial Project in**, p. 271-272, 1968.

SOBHANINIA, Z.; et al. Fetal Ultrasound Image Segmentation for Measuring Biometric Parameters Using Multi-Task Deep Learning. **arXiv preprint arXiv:1909.00273**, 2019.

SOILLE, P. **Morphological Image Analysis: Principles and Applications**, Springer-Verlag, pp. 173–174, 1999.

SPRINGENBERG, J. T.; et al. Striving for simplicity: The all convolutional net. **arXiv preprint arXiv:1412.6806**, 2014.

SUDHAKAR, S. **Custom Image Augmentation.** 2018. Disponível em:< <https://towardsdatascience.com/image-augmentation-14a0aaf0498>>. Acesso em: 13 jan. 2019.

SUNDARESAN, V.; et al. Automated characterization of the fetal heart in ultrasound images using fully convolutional neural networks. In: **IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)**, Melbourne, VIC, p. 671-674, 2017.

- SUZUKI, S.; et al. Topological structural analysis of digitized binary images by border following. **Computer vision, graphics, and image processing**, v. 30, n. 1, p. 32-46, 1985.
- SZEGEDY, C.; et al. Going deeper with convolutions. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Boston, MA, p. 1-9, 2015.
- VIEIRA, J. M. N. Matlab num instante. **Universidade de Aveiro, Portugal**, 2004.
- ZAYED, N. M.; et al. Wavelet segmentation for fetal ultrasound images. In: **Proceedings of the 44th IEEE 2001 Midwest Symposium on Circuits and Systems. MWSCAS 2001 (Cat. No.01CH37257)**, Dayton, OH, USA, p. 501-504 vol.1, 2001.
- ZHANG, Q.; et al. Image segmentation with pyramid dilated convolution based on ResNet and U-Net. In: **International Conference on Neural Information Processing**. Springer, Cham, p. 364-372, 2017.
- ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. In: **European conference on computer vision**. Springer, Cham, 2014. p. 818-833.
- YANG, J. **ReLU and Softmax Activation Functions**. 2017. Disponível em: <<https://github.com/Kulbear/deep-learning-nano-foundation/wiki/ReLU-and-Softmax-Activation-Functions>>. Acesso em: 5 dez. 2018.
- WU, H.; GU, X. Towards dropout training for convolutional neural networks. **Neural Networks**, v. 71, p. 1-10, 2015.
- WU, L.; et al. FUIQA: Fetal ultrasound image quality assessment with deep convolutional networks. **IEEE transactions on cybernetics**, v. 47, n. 5, p. 1336-1349, 2017.
- WU, L.; et al. Cascaded fully convolutional networks for automatic prenatal ultrasound image segmentation. In: **IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)**. IEEE, p. 663-666, 2017.