UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LISET VÁZQUEZ ROMAGUERA

SEGMENTAÇÃO DO MIOCÁRDIO EM IMAGENS DE MRI CARDÍACA UTILIZANDO REDES NEURAIS CONVOLUTIVAS

MANAUS
2017

UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LISET VÁZQUEZ ROMAGUERA

SEGMENTAÇÃO DO MIOCÁRDIO EM IMAGENS DE MRI CARDÍACA UTILIZANDO REDES NEURAIS CONVOLUTIVAS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica na área de concentração Controle e Automação de Sistemas.

Orientadora: Prof[a]. Dra. Marly Guimarães Fernandes Costa
Co-orientador: Prof. Dr. Cícero Ferreira Fernandes Costa Filho

MANAUS
2017

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Romaguera, Liset Vázquez
R756s Segmentação do miocárdio em imagens de MRI cardíaca utilizando redes neurais convolutivas / Liset Vázquez Romaguera. 2017
153 f.: il. color; 31 cm.

Orientadora: Profa. Dra. Marly Guimarães Fernandes Costa
Coorientador: Prof. Dr. Cícero Ferreira Fernandes Costa Filho
Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Amazonas.

1. ressonância magnética cardíaca. 2. segmentação. 3. miocárdio. 4. aprendizado profundo. 5. redes neurais convolutivas. I. Costa, Profa. Dra. Marly Guimarães Fernandes II. Universidade Federal do Amazonas III. Título

LISET VÁZQUEZ ROMAGUERA

SEGMENTAÇÃO DO MIOCÁRDIO EM IMAGENS DE MRI CARDÍACA UTILIZANDO REDES NEURAIS CONVOLUTIVAS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica na área de concentração Controle e Automação de Sistemas.

Aprovado em 17 de abril de 2017.

BANCA EXAMINADORA

Profª. Dra. Marly Guimarães Fernandes Costa, Presidente

Universidade Federal do Amazonas- UFAM

Prof. Dr. José Raimundo Pereira, Membro

Universidade Federal do Amazonas- UFAM

Prof. Dr. Marco Antônio Gutierrez, Membro

Universidade de São Paulo- INCOR/HC/USP

Aos meus pais Lesbia e Carlos

AGRADECIMENTOS

Primeiramente a Deus, que a Ele seja toda a glória;

Aos meus orientadores, os professores Dra. Marly Guimarães Fernandes Costa e Dr. Cicero Fernandes Costa Filho pelo aprendizado adquirido e pela sua orientação;

Ao meu esposo Francisco, pelo seu grande amor, apoio e paciência comigo, por estar ao meu lado nos momentos de alegria e tristeza;

Aos meus pais e irmã, pela força que me dão todos os dias para seguir à frente;

À Universidade Federal do Amazonas, em especial ao Centro de Tecnologia Eletrônica da Informação – CETELI pela concessão de toda infraestrutura para a realização deste trabalho;

A CAPES pelo suporte financeiro.

RESUMO

As doenças cardiovasculares são a principal causa de morte a nível mundial. As tecnologias não invasivas de imageamento cardíaco, tais como a ressonância magnética, são ferramentas essenciais de apoio ao diagnóstico e monitoramento de diversas patologias. Um passo fundamental para a extração dos indicadores da função cardíaca é a segmentação dos contornos do endocárdio e do epicárdio na cavidade ventricular esquerda. Este processo, a maioria das vezes, é realizado manualmente pelos especialistas, o qual exige muito tempo e esforço, além de que é propenso a erros intra e inter-observadores. Esta dissertação desenvolve uma metodologia automática baseada em uma rede neural totalmente convolutiva para segmentar o miocárdio em imagens do eixo curto de ressonância magnética cardíaca. O banco de imagens utilizado é dividido em 10 conjuntos para propósitos de treinamento e teste. São avaliados seis métodos de otimização, a saber, o gradiente descendente estocástico, o gradiente acelerado de Nesterov, o RMSProp, o Adam, o AdaDelta e o AdaGrad. Os melhores resultados foram alcançados com o gradiente descendente estocástico e com o RMSProp. Com o gradiente descendente estocástico foi obtido um coeficiente Dice de 0,9055 e 0,9146, distância de Hausdorff de 10,5244 e 10,7240, sensibilidade de 0,9263 e 0,9135, especificidade de 0,9985 e 0,9986, para o endocárdio e epicárdio, respectivamente. Com o RMSProp foi obtido um coeficiente Dice de 0,9098 e 0,9167, distância de Hausdorff de 9,0421 e 9,7663, sensibilidade de 0,9200 e 0,9116, especificidade de 0,9988 e 0,9987, para o endocárdio e epicárdio, respectivamente.

Palavras chaves: ressonância magnética cardíaca, segmentação, miocárdio, ventrículo esquerdo, aprendizado profundo, redes neurais convolutivas.

ABSTRACT

Cardiovascular diseases are the leading cause of death worldwide. Noninvasive cardiac imaging technologies, such as magnetic resonance, are essential tools to support the diagnosis and monitoring of various pathologies. The previous step for the extraction of cardiac function indicators is the endocardium and epicardium contours segmentation in the left ventricular cavity. This process often is performed manually by the specialists, which requires a lot of time and effort, and is prone to intra and inter-observer errors. This dissertation develops an automatic methodology based on a fully convolutional neural network to segment the myocardium in short axis cardiac magnetic resonance images. The database used is divided into 10 sets for training and testing purposes. Six optimization methods are evaluated: stochastic gradient descend, Nesterov accelerated gradient, RMSProp, Adam, AdaDelta and AdaGrad. The best results were achieved with the stochastic gradient descend and RMSProp. With the former, a Dice coefficient of 0.9055 and 0.9146, Hausdorff distance of 10.5244 and 10.7240, sensitivity of 0.9263 and 0.9135, specificity of 0.9985 and 0.9986 were obtained for endocardium and epicardium, respectively. With RMSProp, a Dice coefficient of 0.9098 and 0.9167, Hausdorff distance of 9.0421 and 9.7663, sensitivity of 0.9200 and 0.9116, specificity of 0.9988 and 0.9987 were obtained for endocardium and epicardium, respectively.

Keywords: cardiac magnetic resonance, segmentation, myocardium, left ventricle, deep learning, convolutional neural networks.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

As doenças cardiovasculares representam a principal causa de morte a nível mundial, sendo responsáveis por 17,3 milhões de mortes por ano, número que deverá aumentar a mais de 23,6 milhões para o ano 2030 (MOZAFFARIAN et al., 2015). Atendendo a esses números, há uma crescente demanda de tecnologia capaz de fornecer informações qualitativas e quantitativas sobre a morfologia e função do coração, de forma que seu uso possa ajudar no diagnóstico clínico, tratamento e acompanhamento das doenças.

Na rotina clínica existem várias técnicas não invasivas de imageamento cardíaco que permitem diagnosticar e avaliar a função cardiovascular, tais como o ecocardiograma, a tomografia computadorizada por emissão de fóton único (SPECT, do inglês *Single Photon Emission Computed Tomography*), a tomografia computadorizada (CT, do inglês *Computed Tomography*) e a ressonância magnética cardíaca (CMRI, do inglês *Cardiac Magnetic Resonance Imaging*) (WU; LIMA, 2003). A CMRI é uma das técnicas de diagnóstico preferidas devido a que fornece um alto contraste entre os diferentes tecidos moles, não utiliza radiação ionizante e possui alta resolução espacial. Por outro lado, além das informações estruturais tais como a anatomia e a caracterização dos tecidos, ela fornece uma análise dinâmica como medidas de perfusão, metabolismo e função em um único exame (WU; LIMA, 2003). Esta técnica será analisada com maior detalhe na Secção 3.1 bem como os planos de captura.

As técnicas mencionadas anteriormente fornecem informação multidimensional, isto é, nos eixos espaciais (largura, altura e profundidade) e no eixo do tempo, gerando dados 2D, 3D ou 4D com alta resolução espacial e temporal. Portanto, um único exame

pode resultar numa grande quantidade de dados para analisar, fato pelo qual existe uma necessidade de automatizar a extração de parâmetros clínicos relevantes.

Por outro lado, a avaliação da função cardíaca requer o cálculo de diferentes indicadores que podem ser classificados em duas categorias: globais e locais. Os globais avaliam o desempenho dos ventrículos em sua capacidade de ejetar sangue, eles incluem a fração de ejeção (EF, do inglês *Ejection Fraction*), a massa ventricular esquerda (LVM, do inglês *Left Ventricle Mass*), o volume ventricular esquerdo (LVV, do inglês *Left Ventricle Volume*), o *Stroke Volume* e a saída cardíaca (FRANGI; NIESSEN; VIERGEVER, 2001). Já os indicadores locais ajudam a avaliar disfunções regionais no coração determinando o estado do tecido do miocárdio, detectando possíveis áreas danificadas por causa de um fluxo sanguíneo reduzido. Alguns indicadores locais são: movimentação parietal regional, espessamento da parede regional e deformação miocárdica regional (PENG et al., 2016).

Todos os parâmetros mencionados anteriormente dependem da segmentação dos contornos do endocárdio e do epicárdio do ventrículo esquerdo (VE) nas sequências de imagens adquiridas a partir da CMRI. A segmentação manual desses contornos em todo o conjunto de dados (ou seja, todos os quadros de tempo por todas as fatias) exige muito tempo e esforço dos especialistas. Portanto, a segmentação automática é muito importante e pode ser considerada como uma tarefa desafiante.

As imagens do eixo curto adquiridas pelo método de ressonância magnética cinemática (cine MR, do inglês *cinematic Magnetic Ressonance*) são muito comuns nos estudos do coração. Na Figura 1 é mostrado um exemplo delas e das suas estruturas de interesse. O ventrículo esquerdo e o ventrículo direito (VD) são duas câmaras inferiores do coração que bombeiam o sangue recebido das câmaras superiores para as artérias pela contração das paredes da câmara. A cavidade ventricular esquerda está preenchida

por um volume de sangue conhecido como *blood pool*. O miocárdio é o músculo que reveste as cavidades ventriculares, sendo limitado internamente pelo endocárdio e externamente pelo epicárdio. Os músculos papilares são estruturas que atravessam a cavidade ventricular.

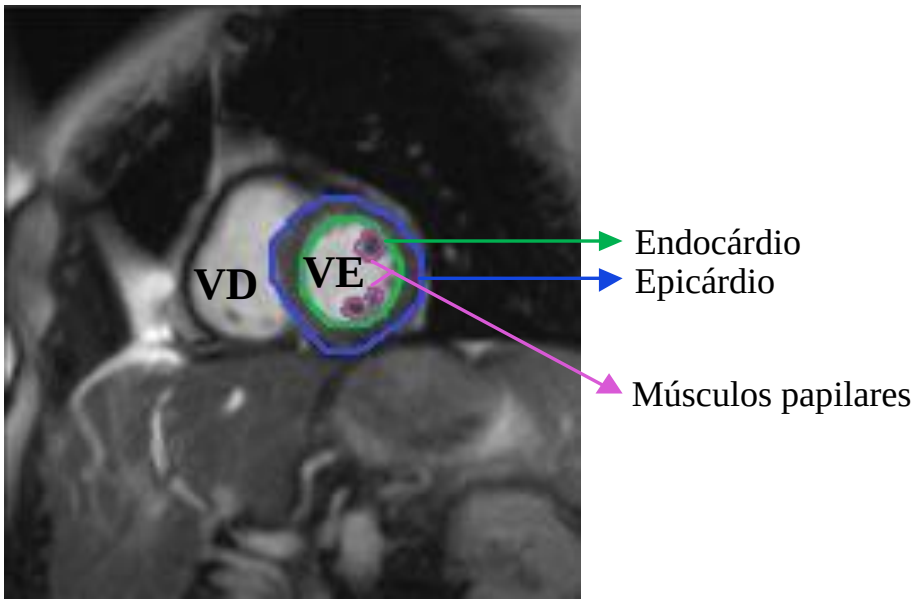


Figura 1. Imagem do eixo menor adquirida pela técnica *cine MR* com as estruturas do coração. Fonte: Adaptada de (SHI, 2012)

Na segmentação destas imagens existem alguns desafios técnicos que precisam ser tratados. Na Figura 2 pode ser observada a grande variabilidade das imagens de CMRI, tanto em termos de aparência quanto de forma. A aparência pode variar devido à utilização dos diferentes *scanners* e protocolos de aquisição. Em termos de forma, o ventrículo pode variar significativamente entre indivíduos, entre diferentes patologias (dilatação ou hipotrofia) e ao longo do tempo.

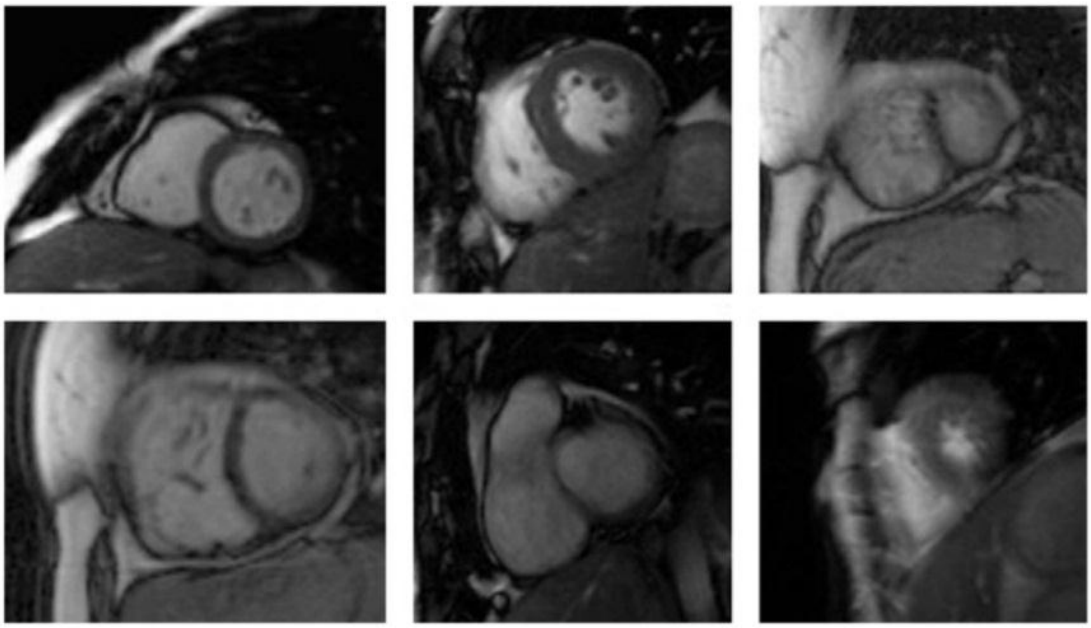


Figura 2. Variabilidade de aparência e de forma em imagens de ressonância magnética cardíaca. Fonte: (SHI, 2012)

A parede do epicárdio se forma na borda entre o miocárdio e os tecidos circundantes, os quais apresentam baixo contraste em relação ao miocárdio. Contudo o contraste entre o miocárdio e a *blood pool* é bom. Isto significa que a parede do endocárdio é mais fácil de ser identificada. No entanto, as intensidades não homogêneas causadas por artefatos do fluxo de sangue podem fazer a segmentação desafiadora. Além disso, os músculos papilares e trabecular no interior das câmaras do coração têm a mesma intensidade do que o miocárdio, o que dificulta a detecção. Na maioria dos protocolos de segmentação manual dos especialistas, os músculos papilares e trabecular não são tomados em consideração para a delimitação da parede do endocárdio.

Conforme o exposto até aqui, o desenvolvimento de um método de segmentação dos contornos do endocárdio e o epicárdio constitui uma contribuição significativa, pois possibilita o posterior cálculo dos indicadores da função cardíaca. Nas últimas duas décadas foram propostos vários métodos baseados em diferentes técnicas de processamento de imagem e visão computacional. No próximo capitulo será

apresentada uma visão geral deste tema, bem como uma análise dos métodos publicados nos últimos anos. É importante mencionar que já existem alguns sistemas comerciais que facilitam o trabalho clínico dos médicos. Por exemplo, o pacote de software Medis que contém a ferramenta "QMass®" (MEDIS MEDICAL IMAGING SYSTEM, 2015) para a análise quantitativa de imagens de ressonância magnética cardíaca; o "Argus 4D Ventricular Function" (EKINCI, 2009) que permite visualizar e avaliar a função ventricular, bem como calcular vários indicadores e outros como o "CMR 42®", da empresa canadense Circle CVI e o "Heart Navigator" da empresa Philips.

Como pode ser observado, existe uma ampla variedade de métodos para segmentação de imagens de ressonância magnética cardíaca. Mais recentemente tem sido explorada uma nova técnica baseada em redes neurais convolutivas (CNN, do inglês *Convolutional Neural Networks*), da qual existem poucos trabalhos até o momento no que diz respeito à aplicação em imagens cardíacas. Estas redes, segundo sua concepção e modo de operação, pertencem à área do aprendizado de máquina profundo, a qual nos últimos anos ganhou popularidade pelos seus excelentes resultados em visão computacional. De forma geral, as redes neurais estão baseadas na estrutura do sistema nervoso do cérebro humano e tentam imitar o seu comportamento. Elas têm a capacidade de aprender a realizar tarefas após uma etapa inicial de treinamento e de generalizar de casos anteriores a novos casos.

O presente trabalho pretende contribuir com o estudo destas novas arquiteturas de aprendizagem profundo e aplicá-las à solução do problema de segmentação de imagens de ressonância magnética cardíaca.

1.1 Objetivo geral

Segmentar de forma automática o miocárdio em imagens de ressonância magnética cardíaca.

1.2 Objetivos específicos

- Contribuir para o estado da arte no tema de segmentação automática do miocárdio em imagens CMRI através da avaliação de uma arquitetura de rede neural convolutiva treinada com seis métodos de otimização.
- Avaliar o desempenho do método proposto através do uso de uma base de dados robusta e pública, segundo métricas de qualidade bem estabelecidas, de modo a viabilizar o *benchmark* com as técnicas presentes na literatura.

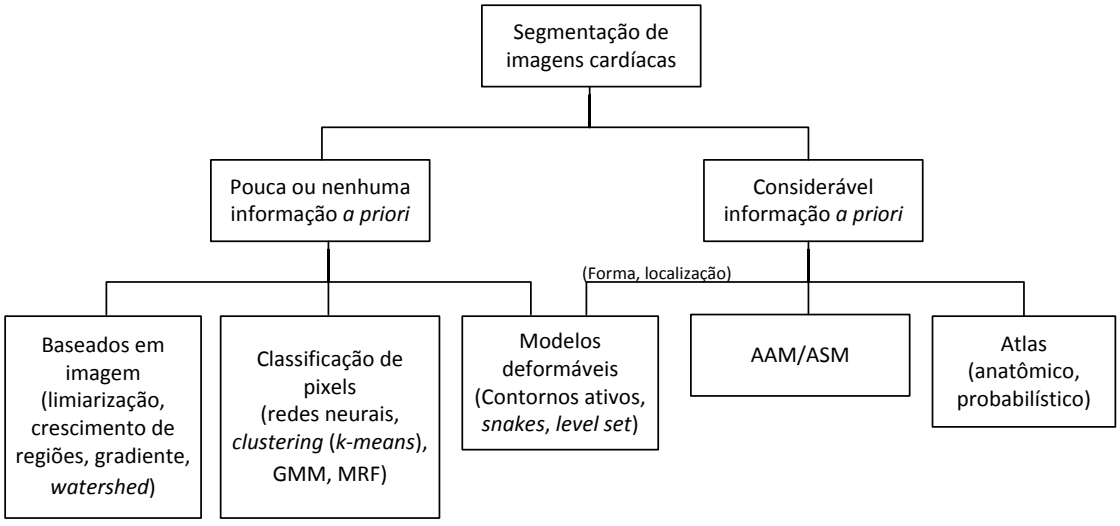
2 REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta uma revisão bibliográfica de algumas técnicas de segmentação do miocárdio. Faz-se uma abordagem das metodologias propostas e dos resultados publicados em vários artigos. As bases de dados *IEEE Explore*, *Web of Science* e *Google Scholar* foram utilizadas para a busca dos trabalhos mais relevantes dos últimos anos.

Segundo (PETITJEAN; DACHER, 2011) os métodos de segmentação podem ser classificados em duas categorias considerando o nível de informação externa fornecida (vide Figura 3):

1) segmentação com pouca ou nenhuma informação *a priori* (inclui métodos baseados em imagens, classificação de pixels e modelos deformáveis).

2) segmentação com considerável informação *a priori* (inclui métodos baseados em modelos deformáveis com conhecimento prévio de forma, modelos de aparência ativos (AAM, do inglês *Active Appearance Models*), modelos de forma ativos (ASM, do inglês *Active Shape Models*) e atlas.



(GMM: *Gaussian Mixture Model*, MRF: *Markov Random Field*)

Figura 3. Classificação dos métodos de segmentação de imagens cardíacas

Fleagle e colaboradores (1991) foram pioneiros na identificação das bordas do endocárdio e do epicárdio em imagens de ressonância magnética. Os autores trabalharam com uma série de imagens do eixo curto de 13 corações excisados de animais, além de imagens de 11 voluntários humanos. Eles aplicaram um método baseado na busca de grafos nas imagens "*ex vivo*" e "*in vivo*". As bordas obtidas foram comparadas com as bordas da segmentação manual feitas por um observador. A correlação para os corações excisos foi de 0,97 e 0,99 para o endocárdio e epicárdio, respectivamente. No caso dos corações humanos foi de 0,92 e 0,90 para o endocárdio e epicárdio, respectivamente.

Já mais recente, Wang e colaboradores (2009) fizeram algumas modificações no método de Campo Aleatório de Markov (MRF, do inglês *Markov Random Field*) para melhorar o desempenho do referido método na segmentação das bordas do endocárdio e do epicárdio. O MRF é um processo estocástico no qual as relações espaciais da imagem são incluídas no processo de etiquetagem através da dependência estatística entre pixels vizinhos. Neste método, assume-se que a imagem contém L classes diferentes e cada pixel pertence apenas a uma classe. O problema de segmentação é abordado como a atribuição de cada pixel a uma classe.

O MRF não utiliza a informação das bordas, fato pelo qual em imagens de MRI cardíaca com bordas de pouco contraste ou fracas, os resultados da segmentação não são satisfatórios. Uma das contribuições dos autores foi um modelo de segmentação baseado em MRF melhorado que integra região, conhecimento *a priori* e informação das bordas.

Este método também utiliza o algoritmo de otimização de Recozimento Simulado (SA, do inglês *Simulated Annealing*) para obter o critério máximo *a posteriori*. O SA é um algoritmo de otimização global que tem uma convergência lenta.

Os autores introduziram o algoritmo de Recozimento Simulado Caótico (CSA, do inglês *Chaotic Simulated Annealing*) no modelo MRF para melhorar a taxa de convergência. O algoritmo CSA tem um mecanismo de ergodicidade de caos, ele se beneficia da ergodicidade global do Algoritmo do Caos (COA, do inglês *Chaos Algorithm*) e das regras heurísticas do algoritmo SA no processo de busca. O algoritmo CSA melhora a velocidade da otimização global.

Os autores utilizaram imagens adquiridas no "*Second Affiliated Hospital of China Medical University*". Dois especialistas realizaram a segmentação manual das imagens para a avaliação do método. Eles obtiveram uma precisão de 97%, uma *True Positive Fraction* (TPF) = 99 %, *False Negative Fraction* (FNF) = 1 %, *False Positive Fraction* (FPF) = 2 % e *True Negative Fraction* (TNF) = 98 %, em um tempo de 201 segundos.

No trabalho de (DAKUA; SAHAMBI, 2011) é proposta uma modificação à função de peso no método do Caminho Aleatório (RW, do inglês *Random Walk*) para melhorar os resultados da segmentação, particularmente em imagens isquêmicas.

O método do RW está baseado na teoria de grafos. Um grafo $G = (V, E)$ é formado por um conjunto de nós ou vértices V que correspondem aos pixels da imagem mais dois nós adicionais S (fonte) e T (dreno). As arestas E correspondem a uniões não direcionadas que conectam nós vizinhos entre si e os dois nós especiais (S e T) com outros nós. Tradicionalmente, é usada uma função de ponderação gaussiana para calcular os pesos das arestas:

$$\omega_{ij} = e^{-\beta(H_u - H_v)^2} \quad (1)$$

em que, $\omega_{i,j}$ é o peso da aresta na posição (i, j); H_u e H_v são as intensidades da imagem no pixel u e v, respectivamente; β é um parâmetro de livre escolha, o qual introduz variabilidade no desempenho. Um valor inadequado de β causa uma segmentação deficiente.

Os autores modificaram totalmente a função de ponderação utilizada para o RW. Eles fizeram todo um desenvolvimento matemático até obter a derivada da expressão (1) para achar a função de ponderação derivada da gaussiana (ω_{DroG}), mostrada na equação 2, que foi a que aplicaram ao método RW:

$$\omega_{DroG}(x) = \frac{-x}{2\sqrt{2\pi\sigma^3}} e^{-\frac{x^2}{2\sigma^2}} \quad (2)$$

em que, σ é o desvio padrão da função gaussiana original e x pertence a uma região de interesse (ROI, do inglês *Region of Interest*) determinada.

Nesse trabalho os autores mostraram através de algumas imagens que o método RW teve melhor desempenho usando a função de ponderação proposta que com a função gaussiana tradicional, especificamente em pacientes isquêmicos. Entretanto, a avaliação do método foi apenas qualitativa, comparando visualmente as imagens obtidas aplicando o RW com cada uma das funções de ponderação.

Constantinides e colaboradores (2012) apresentaram uma metodologia de segmentação baseada em filtragem morfológica e um *snake* conduzido pelo fluxo do vetor gradiente (GVF, do inglês *Gradient Vector Flow*). Inicialmente foi determinada uma ROI na imagem de entrada para garantir uma maior eficiência da metodologia. Primeiro foi detectada a região cardíaca assumindo que o coração é a única estrutura com movimento durante um ciclo cardíaco. Em seguida, foi estimada a menor região

que inclui o ventrículo esquerdo. Para isto, foi aplicada a Transformada de Hough para detectar círculos concêntricos. Foram aplicadas duas regras: a primeira foi calcular a projeção de intensidade máxima (PIM) dos círculos detectados e aqueles com os centros mais próximos à máxima PIM da imagem foram selecionados. A segunda regra usou o fato de que os níveis de cinza dentro da cavidade ventricular esquerda devem ser altos (alto valor de média, m) e homogêneos (baixo desvio padrão, σ). Para cada círculo foi calculada a relação $h = m/\sigma$ e o de maior valor de h foi selecionado. Dentro do quadro que contém o círculo foi executado o algoritmo *fuzzy k-means*, para agrupar os pixels em duas classes: miocárdio ou cavidade do ventrículo esquerdo. Do conjunto de pixels conectados pertencentes à segunda classe foi calculado o centro de massa P_0.

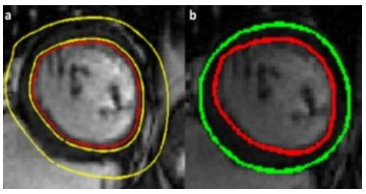
Para achar o contorno do endocárdio foi aplicado um filtro morfológico seguido de um método de contornos ativos. Um pequeno círculo de 3 mm de raio centrado em P_0 foi usado como *snake* inicial C. Sua evolução foi conduzida pelas energias interna e externa (vide equação 3).

$$\int_0^1 \left[\alpha \left| \frac{\delta C}{\delta s} \right|^2 + \beta \left| \frac{\delta C}{\delta s} \right|^2 + \kappa_p N(s) + \kappa V_\mu(s) \right] ds \qquad (3)$$

em que, C é o *snake*, α é elasticidade, β é a rigidez, κp é o parâmetro de pressão, N é a normal do *snake*, k é o peso associado ao GVF e $V\mu$ é o GVF.

A energia interna é controlada por dois parâmetros, α e β, ponderados segundo a elasticidade e rigidez, respectivamente. A energia externa é uma função do parâmetro de pressão κp aplicado à normal do *snake* N e o peso κ, associado ao GVF $V\mu$, o qual foi estimado a partir do gradiente da imagem usando um parâmetro de regularização μ. Os valores utilizados foram: $\alpha = 1$, $\beta = 40$, $\kappa p = 0{,}8$, $\kappa = 1{,}7$ e $\mu = 0{,}2$.

No caso do epicárdio a metodologia foi a mesma. O *snake* foi inicializado no contorno detectado anteriormente com os seguintes parâmetros: α = 1, β = 40, κp = 0,35, κ = 1,7 e μ = 0,2. A região de busca para ajustar o *snake* foi restrita com uma máscara em forma de anel. A borda interna do anel foi definida a uma distância de 2,5 mm do endocárdio e a externa a uma distância de 20 mm. Na Figura 4 é mostrada a máscara (cor amarelo) e o endocárdio previamente detectado (cor vermelha).



(a) (b)

Figura 4. (a) Segmentação do endocárdio (cor vermelha) e definição da máscara (cor amarela) (b) Segmentação do endocárdio (cor vermelha) e epicárdio (cor verde).
Fonte: (CONSTANTINIDES et al., 2012)

O método foi avaliado segundo a percentagem de "bons contornos", através das métricas: distância perpendicular média (APD, do inglês *Average Perpendicular Distance*) e a métrica Dice (DICE, 1945). A percentagem de "bons contornos" foi de 80 ± 16 % e 71 ± 26 %, a APD foi de 2,44 ± 0,56 mm e 2,80 ± 0,71 mm e a métrica Dice foi de 0,86 ± 0,05 e 0,91 ± 0,03 para o endocárdio e epicárdio, respectivamente.

No trabalho de Hadhoud e colaboradores (2012) inicialmente foi determinada a posição do coração na imagem. Eles extraem uma ROI calculando o desvio padrão de cada uma das imagens para formar um mapa a partir deste parâmetro. A imagem resultante desse mapa é binarizada. O limiar usado nessa operação é calculado depois de plotar a função de densidade de probabilidade dos valores de desvio padrão e tomar o valor que obtém o 80% da área em baixo da curva.

Para segmentar o contorno do endocárdio, a ROI foi dividida em subimagens (*patches*) de tamanho 7 × 7, e foram determinadas as seguintes 6 características para cada pixel: (1) magnitude do gradiente, (2) maior valor próprio, (3) valor após a aplicação de um filtro de mediana 3 × 3, (4) 5 × 5, (5) 7 × 7 e (6) valor de intensidade de nível de cinza. Em seguida, foi usada a técnica de Análise de Componentes Principais (PCA, do inglês *Principal Component Analysis*) para reduzir a dimensionalidade dos dados de entrada ao classificador kNN (*k-Nearest Neighbors*), *k* = 1. A saída do classificador é pós-processada para determinar as componentes conectadas com maior circularidade usando equação 4:

$$C = \frac{4\pi A}{P^2} \tag{4}$$

em que, *A* é a área e *P* o perímetro do objeto.

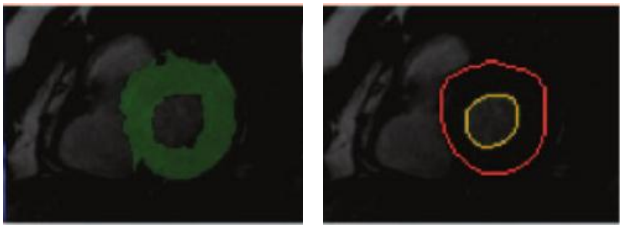
Para segmentar o epicárdio, os autores convertem de coordenadas cartesianas a polares uma subimagem (extraída da ROI) centrada na cavidade do VE e com um raio de 30 pixels. Nessa imagem foi aplicado o detector de bordas de Canny. Além disso, eles convertem a mesma subimagem da imagem binária da cavidade do VE a coordenadas polares com igual centro e raio, para usa-la como máscara para eliminar as bordas não desejadas. Por fim, foram selecionadas as menores componentes conectadas, a imagem foi levada a coordenadas polares e, a partir dos pontos resultantes, foi calculado o fecho convexo (c*onvex hull*) para determinar o contorno do epicárdio.

O método apresentou uma sensibilidade de 0,9561 ± 0,0625 e 0,9332 ± 0,0766 (média ± desvio padrão), uma especificidade de 0,9890 ± 0,0089 e 0,9849 ± 0,0092, valor da métrica Dice de 0,8937 ± 0,0747 e 0,9161 ± 0,0473 para o endocárdio e para o

epicárdio, respectivamente. Além desses resultados, os autores calcularam a EF a partir das segmentações manual e automática, obtendo uma correlação entre elas de 0,88.

Uzunbaş e colaboradores (2012) desenvolveram um método semiautomático usando cortes de grafos e modelos deformáveis. O método de corte de grafos está baseado em otimização de energia.

O método proposto aplica inicialmente a técnica de cortes de grafos para segmentar a *blood pool (bp)*. A imagem inicial foi capturada na fase de ES (t_o). Nela os pixels pertencentes ao fundo e ao objeto de interesse foram marcados de forma manual, como inicialização. A *blood pool* foi extraída aplicando o algoritmo *min-cut* que minimiza uma função de custo (BOYKOV; JOLLY, 2000)(BOYKOV; FUNKA-LEA, 2006). O centroide da região segmentada serve de inicialização para a próxima imagem. Esta mesma técnica é usada em um segundo passo para segmentar de forma aproximada o miocárdio. Os pixels do fundo e do objeto foram inicializados usando operações morfológicas, especificamente um *convex hull* dos pontos da região da *blood pool* seguido de uma dilatação com elemento estruturante em forma de disco de raio 7. A seguir, foi aplicado um ajuste de um *level set*. Para dirigir a deformação da curva foi usada a técnica de Deformação de Forma Livre (FFD, do inglês *Free Form Deformation*) (SEDERBERG et al., 1986). A ideia principal é envolver um objeto com um volume parametrizado para ligá-lo com pontos de controle do dito volume. A segmentação do miocárdio obtida com os cortes de grafos foi usada como inicialização de uma região FFD. Na Figura 5 são mostrados os contornos do endocárdio e epicárdio obtidos. O método obteve uma métrica Dice de 0,82 ± 0,06 e 0,91 ± 0,03 e uma APD (mm) de 2,98 ± 0,88 e 1,78 ± 0,35 para o endocárdio e epicárdio, respectivamente.



(a) **(b)**

Figura 5. (a) Segmentação do miocárdio com os cortes de grafos (b) Resultado do ajuste de *level set* inicializado com (a)
Fonte: (UZUNBAS et al., 2012)

Dreijer e colaboradores (2013) utilizaram os Campos Aleatórios Condicionais (CRF, do inglês *Conditional Random Field*), os quais são modelos probabilísticos geralmente aplicados em reconhecimento de padrões e aprendizado de máquinas para predição estruturada. O CRF é um método de aprendizado supervisionado que classifica cada amostra levando em conta a sua vizinhança.

Os autores modelaram o endocárdio e o epicárdio como duas séries de raios desde o centro até as bordas, que estão inter-relacionados usando características que descrevem forma e movimento. As características da imagem foram tomadas a partir da informação de bordas de imagens anotadas por humanos. Elas são combinadas dentro de um CRF treinado discriminativamente.

O método *Loopy Belief Propagation*, o qual é uma abordagem de programação dinâmica que soluciona questões de probabilidade condicional em um modelo gráfico, foi usado para estimar segmentações nas imagens de uma sequência de vídeo.

O método de Powell é uma técnica de busca do mínimo local de uma função. Ele foi usado pelos autores para encontrar parâmetros para o CRF, minimizando a diferença entre a segmentação *ground truth* e a estimada através do *Loopy Belief Propagation*.

O método foi avaliado utilizando a base de dados Sunnybrook obtendo um valor da métrica Dice de 0,91 ± 0,02 e de 0,93 ± 0,02 para os contornos do endocárdio e do epicárdio, respectivamente.

Este método apresentou problemas em imagens de pacientes com hipertrofia donde a *blood pool* desaparece visualmente no final da sístole. Um inconveniente é que o centro do ventrículo esquerdo deve ser marcado manualmente ao início do algoritmo, pelo qual não é um método totalmente automático. Segundo (AVENDI; KHERADVAR; JAFARKHANI, 2016) os métodos baseados em CRF são difíceis computacionalmente devido à complexidade da estimação dos parâmetros, e sua convergência não é garantida.

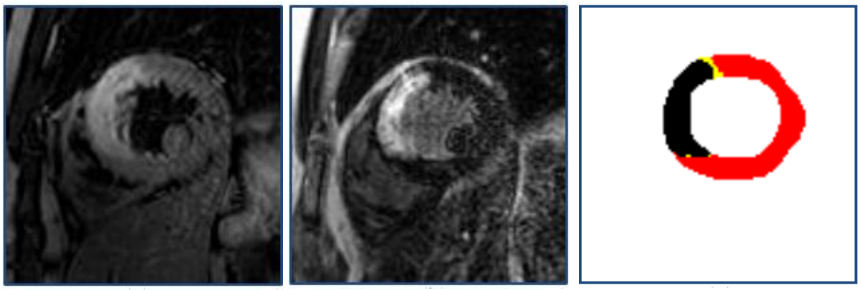
O trabalho de Marino e colaboradores (2014) começa identificando a cavidade do VE assumindo que o coração é a única estrutura com movimento da sequência de imagens de CMRI. Por tanto, através da diferencia entre imagens podem ser detectadas as posições dos ventrículos direito e esquerdo. Foi aplicada a Transformada de Hough para detectar o VE considerando sua forma circular. Desta forma, foi determinada uma ROI, cujo centro foi usado como inicialização na etapa de segmentação.

Para a segmentação do contorno do endocárdio foi aplicado o método de *level set* baseado em um modelo estatístico explorando a informação relacionada à distribuição do ruído na imagem. Este método conduz a evolução da curva particionando à imagem em regiões maximamente homogêneas, as quais têm diferentes padrões locais de ruído. Devido aos músculos papilares presentes no VE, o resultado do contorno não foi adequado. Por isso, os autores aplicaram uma evolução adicional numa região limitada por uma máscara. A referida máscara foi criada considerando a diferença entre a curva previamente detectada e a curva obtida da filtragem da imagem original com o filtro anisotrópico do Perona-Malik (1990) para homogeneizar os níveis de cinza na cavidade do VE.

As bordas do epicárdio foram obtidas aplicando um *level set* baseado em curvatura e advecção, usando a máscara anterior como uma matriz de pesos. A inicialização foi feita a partir do contorno do endocárdio.

O método foi avaliado com métricas de similaridade como a Dice, o índice de Jaccard, e as distâncias de Hausdorff em pixels e em milímetros para o final da diástole (ED, do inglês *End of Diastole*) e da sístole (ES, do inglês *End of Systole*). No caso da ED, os autores obtiveram uma métrica Dice de 0,93 ± 0,06 e 0,95 ± 0,03, um índice de Jaccard de 0,88 ± 0,09 e 0,90 ± 0,06, uma distância de Hausdorff (em pixels) de 3,42 ± 0,46 e 3,72 ± 0,51 e uma distância de Hausdorff em mm de 1,71 ± 0,32 e 1,86 ± 0,39 para o endocárdio e miocárdio, respectivamente. No caso da ES obtiveram uma métrica Dice de 0,91 ± 0,04 e 0,89 ± 0,11, um índice de Jaccard de 0,83 ± 0,07 e 0,82 ± 0,16, uma distância de Hausdorff (em pixels) de 3,43 ± 0,51 e 3,84 ± 0,55 e uma distância de Hausdorff em mm de 1,72 ± 0,35 e 1,94 ± 0,42 para o endocárdio e miocárdio, respectivamente.

Liu e colaboradores (2014) propuseram um método direcionado à segmentação de imagens de ressonância magnética de Realce Tardio (DE, do inglês *Delayed Enhanced*) (Figura 6 (a)), as quais permitem visualizar os infartos, e de imagens T2 (Figura 6 (b)), as quais fornecem informações das regiões isquêmicas.



(a) **(b)** **(c)**

Figura 6. Imagens MRI cardíaca: (a) T2 (b) realce tardio (c) resultado da classificação do miocárdio.
Fonte: (LIU et al., 2014)

Os autores combinaram as informações das duas modalidades e segmentaram simultaneamente dentro de um *framework* unificado. O método, baseado em Modelo de Mistura Gaussiana, classifica os pixels da imagem em quatro classes: (1) miocárdio, (2) ventrículo esquerdo, (3) ventrículo direito e (4) fundo (vide Figura 6 (c)).

Dada uma variável aleatória X de dimensão d e uma mistura com K componentes, a função de probabilidade de mistura de gaussianas pode ser definida pela equação 5:

$$\Phi(X|\Theta_k) = \sum_{i=1}^{K} \pi_i \phi(X|\theta_i) \tag{5}$$

em que, cada θ_i corresponde ao conjunto de parâmetros definidos pela i-ésima componente da mistura; $\pi_i \in [0, 1]$ com $i \in (1, 2, ..., K)$ e $\sum_{i=1}^{K} \pi_i = 1$; o vetor $\Theta_k = (\pi_1, \ldots \pi_k, \theta_1, \ldots, \theta_k)$ é o conjunto dos parâmetros da mistura.

Cada componente $\phi(X \mid \Theta_i)$ da mistura é uma função de densidade de probabilidade gaussiana definida por:

$$\phi(X = x|\theta_i) = \frac{1}{(2\pi^{\frac{d}{2}})|\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i)} \tag{6}$$

em que, μ_i é a média; Σ_i é a matriz de covariância e $\theta_i = (\mu_i, \Sigma_i)$ representa os parâmetros de uma Gaussiana.

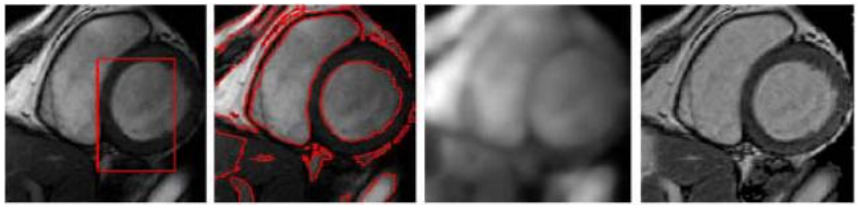
No método, primeiro são determinadas as probabilidades das 4 classes para um pixel da imagem através da propagação de atlas probabilísticos, construídos a partir de indivíduos saudáveis utilizando registro de imagem. Depois, os valores de intensidades

de ambas as imagens registradas foram modelados em conjunto com um vetor aleatório bivariado. Os autores propuseram um modelo de Mistura Gaussiana multicomponentes, que leva em conta as heterogeneidades de intensidades entre as classes. Os parâmetros do modelo e da segmentação forma estimados maximizando a função de verossimilhança logarítmica usando o algoritmo *Expectation Maximization*.

Foram adquiridos registros nas modalidades DE e T2 de seis pacientes que tinham sofrido um infarto do miocárdio. As imagens foram segmentadas manualmente por um clínico usando o software "ITK-Snap". A comparação entre as segmentações manual e a estimada pelo método proposto resultou em uma Dice de 0,623 ± 0,129 e de 0,643 ± 0,084 nas imagens T2 e DE, respectivamente.

O trabalho apresentado por Wang e colaboradores (2015) está baseado na aplicação de modelos deformáveis (*level set*) e o algoritmo de agrupamento *fuzzy C-means*. Inicialmente foi determinada uma ROI a partir da observação de que em todas as imagens da base de dados usada, o VE aparece entre os pixels 76 e 185.

Geralmente as imagens MRI são afetadas por intensidades não homogêneas. A evolução de um *level set* (inicializado com um retângulo) permitiu a estimação do campo de polarização (*bias field*) através da minimização de energia, como é mostrado na Figura 7.



(a) (b) (c) (d)

Figura 7. Ilustração do *level set*: (a) Retângulo inicial (b) Evolução do *level set* (c) *Bias field* estimado (d) Imagem corrigida
Fonte: (WANG et al., 2015)

Subtraindo o *bias field* da imagem original foi obtida a imagem corrigida, a qual posteriormente foi binarizada. Das componentes conectadas resultantes foi selecionada a de maior circularidade (correspondente à *blood pool*) e nela foi ajustado o *level set*, obtendo-se o contorno do endocárdio.

Em seguida, foi aplicado o algoritmo *fuzzy C-means* para agrupar os pixels da imagem corrigida em três classes: *blood pool*, miocárdio e fundo. A imagem resultante possui três níveis de cinza, o valor intermédio corresponde aos pixels do miocárdio.

Combinando o miocárdio com a segmentação prévia da *blood pool* foi obtida a imagem mostrada na Figura 8 (a). Nela foi aplicada morfologia matemática para obter o epicárdio com o conhecimento *a priori* de que o VE é similar a um círculo.

(a) (b) (c)

Figura 8. (a) Resultado da combinação dos pixels do miocárdio com os da *blood pool* (b) Resultado da morfologia (c) Segmentação do epicárdio
Fonte: (WANG et al., 2015)

Os autores avaliaram o seu método somente para os contornos do endocárdio. A percentagem de "bons contornos" foi de 100 %, a APD foi de 1,2312 mm e a métrica Dice foi de 0,9429.

No trabalho de Tran (2016) foi implementada e treinada uma rede neural totalmente convolutiva para segmentar os contornos do endocárdio e epicárdio nos ventrículos esquerdo e direito. A validação do desempenho foi feita utilizando as bases de dados Sunnybrook, *Left Ventricle Segmentation Challenge* (LVSC) e *Right Ventricle Segmentation Challenge* (RVSC).

Na etapa inicial de preparação dos dados, o autor aplicou uma técnica multi-resolução para obter regiões de interesse de vários tamanhos baseados em que a cavidade do coração está localizada aproximadamente no centro da imagem. Também aumentou os dados do conjunto de treinamento fazendo transformações como rotação, giro horizontal e vertical.

A rede proposta está formada por camadas de convolução, de unidades lineares retificadoras e de normalização de média e variância intercaladas com camadas de sub-amostragem (*pooling*). No final da rede encontram-se uma camada de sobreamostragem e uma *Softmax*. O treinamento foi feito em 10 épocas (passes sobre todo o conjunto de treinamento) utilizando o algoritmo do gradiente descendente com momento de 0,9. A taxa de aprendizado foi atualizada com um decaimento polinomial durante o treinamento segundo a fórmula:

$$base\_lr \times (1 - \frac{iter}{max\_iter})^{power} \quad (7)$$

em que, *base_lr* = 0,01 é a taxa de aprendizado inicial, *iter* é a iteração atual, *max_iter* é o número máximo de iterações (depende do tamanho do conjunto de dados) e *power* = 0,5 controla a taxa de decaimento.

Neste trabalho foi aplicada uma estratégia de transferência de pesos da seguinte forma: (1) treinamento de um modelo de rede totalmente convolutiva utilizando a base de dados LVSC e os parâmetros mencionados anteriormente; (2) os pesos resultantes foram copiados, ou seja, transferidos para outro modelo a ser utilizado com as restantes duas bases de dados. Neste caso a taxa de aprendizado inicial foi de 0,001 com o intuito

de refinar a atualização dos pesos durante o algoritmo de *backpropagation*. Este processo melhorou ligeiramente os resultados.

Em cada base de dados os autores aplicaram diferentes métricas. Na Sunnybrook a Dice obtida foi de 0,92 ± 0,03 e 0,96 ± 0,01, a APD (mm) de 1,73 ± 0,35 e 1,65 ± 0,31 e a percentagem de "bons contornos" foi de 98,48 ± 4,06 e 99,17 ± 2,20, para o endocárdio e epicárdio, respectivamente.

Na base de dados LVSC, para a área segmentada do miocárdio, o índice de Jaccard foi de 0,74 ± 0,13, a sensibilidade foi de 0,83 ± 0,12, a especificidade foi de 0,96 ± 0,03, o valor preditivo positivo (PPV, do inglês *Positive Predictive Value*) foi de 0,86 ± 0,10 e o valor preditivo negativo (NPV, do inglês *Negative Predictive Value*) foi de 0,95 ± 0,03.

Na base de dados RVSC a métrica Dice foi de 0,84 ± 0,21 e 0,86 ± 0,20 e a distância de Hausdorff (mm) foi de 8,86 ± 11,27 e 9,33 ± 10,79, para o endocárdio e epicárdio, respectivamente.

2.1 Considerações finais

Na Tabela 1 é apresentado um sumário dos artigos analisados com respeito aos métodos e as bases de dados utilizados e os resultados reportados. Como pode ser observado, alguns autores avaliam os seus resultados usando conjuntos de dados próprios com diferentes casuísticas. A menor casuística foi utilizada por Liu e colaboradores (2014) que avaliaram seu trabalho em imagens de apenas seis pacientes, sendo este o menor número se comparado com os outros trabalhos. O trabalho de Wang e colaboradores (2009) também utiliza um conjunto de dados próprio, sem especificar o número de imagens e pacientes. A desvantagem do uso de imagens proprietárias é que

não viabiliza o *benchmark* com outros métodos. Entretanto, existem trabalhos que utilizam conjuntos de imagens públicos. No ano 2009, o Centro de Ciências da Saúde Sunnybrook (*Sunnybrook Health Sciences Centre*), no Canada, disponibilizou uma base de dados com imagens de ressonância magnética cardíaca que inclui o *ground truth* feito por dois especialistas. Essa base se tornou uma plataforma popular para a avaliação de algoritmos de segmentação. Existem outras bases de dados públicas como a criada na Universidade de York (ANDREOPOULOS; TSOTSOS, 2008), e as pertencentes aos desafios *Left Ventricle Segmentation Challenge* (SUINESIAPUTRA et al., 2014) e *Right Ventricle Segmentation Challenge* (PETITJEAN et al., 2015).

Podemos dizer que existe uma ampla variedade de métodos para abordar o problema de segmentação de imagens cardíacas. Dentre os trabalhos analisados, quatro deles estão baseados ou pelo menos utilizam, parcialmente, modelos deformáveis sejam em forma de *snakes* ou *level set*. Os valores da métrica Dice para eles estão na faixa entre 0,82 ± 0,06 / 0,95 ± 0,03 para o endocárdio e epicárdio, respectivamente. Algumas limitações destes métodos são o problema da inicialização, o custo computacional e a velocidade lenta devido aos cálculos iterativos.

Outros trabalhos incluem os grafos, por exemplo, Fleagle (1991) propôs um método baseado na busca de grafos. Ele obteve uma correspondência entre as segmentações automática e manual de 0,92 e 0,90, para o endocárdio e epicárdio, respectivamente. Estes resultados são razoáveis, mas o inconveniente é que a avaliação foi feita em imagens de apenas 11 pacientes.

O trabalho de Dreijer e colaboradores (2013), baseado em campos aleatórios condicionais é avaliado com bases de dados públicas, obteve resultados razoavelmente bons. No entanto, a maior desvantagem dos CRF é a alta complexidade computacional. Também o trabalho de Tran (2016) obteve resultados relevantes e foi o primeiro a

implementar e aplicar uma rede totalmente convolutiva à segmentação de imagens de ressonância magnética cardíaca.

As métricas utilizadas para a avaliação quantitativa dos métodos são diversas, dentre elas a Dice é a mais utilizada. Utilizando a base de dados Sunnybrook, a faixa de valores da referida métrica para o endocárdio está entre 0,82, obtido pelo Uzunbas e colaboradores (2012), e 0,94, obtido pelo Wang e colaboradores (2015); para o epicárdio está entre 0,91, obtido pelo Uzunbas e colaboradores (2012), e 0,96, obtido pelo Tran (2016).

De forma geral, a debilidade destes métodos consiste na limitação dos parâmetros a um determinado conjunto de dados. Nesse sentido, o uso das redes neurais apresenta uma grande vantagem que consiste na possibilidade de aprender a realizar uma tarefa e de generalizar de casos anteriores a novos casos, após uma etapa inicial de treinamento. Desta forma, essa técnica pode se adaptar com maior facilidade a dados provenientes de diferentes sistemas de aquisição ou *scanners*.

O surgimento de uma arquitetura de rede neural totalmente convolutiva (LONG; SHELHAMER; DARRELL, 2015) viabilizou a aplicação das redes neurais convolutivas, usadas até esse momento apenas para classificação, em tarefas de segmentação semântica. Uma prova disto é o trabalho proposto pelo Tran (2016) que mostra como um mesmo modelo poder ser usado para segmentar diferentes conjuntos de dados. Nessa mesma linha está direcionado o presente trabalho, o qual pretende continuar o estudo das redes neurais totalmente convolutivas aplicadas à segmentação de imagens de ressonância magnética cardíaca.

Tabela 1 Sumário da revisão da literatura sobre o tema "métodos de segmentação do miocárdio em imagens de MRI cardíaca"

Ano	Autor (es)	Título	Bases de dados usadas	Técnica usada	Resultados (média ± desvio padrão)
1991	S. R. Fleagle, D. R. Thedens, J. C. Ehrhardt, T. D. Scholz, D. J. Skorton	*Automated identification of left ventricular borders from spin-echo magnetic resonance images. Experimental and clinical feasibility studies.*	Imagens do eixo curto de 13 corações excisados de animais e de 11 voluntários humanos	Método baseado na busca de grafos.	Endocárdio animal: r = 0,97 Epicárdio animal: r = 0,99 Endocárdio humano: r = 0,92 Epicárdio humano: r = 0,90 r (correspondência entre segmentação automática e manual nas suas imagens)
2009	G. Wang, Y. Guo, S. Zhang , Y. Ma	*Novel Segmentation Method for Left Ventricular from Cardiac MR Images Based on Improved Markov Random Field Model*	Imagens do "*Second Affiliated Hospital of China Medical University*"	Melhora do MRF incorporando informação de regiões e das bordas. Otimização da função objetivo usando Recozimento Simulado Caótico.	Precisão = 97 % *TPF* = 99 % *FNF* = 1 % *FPF* = 2 % *TNF* = 98 % Tempo: 201 segundos
2011	S. P. Dakua J. S. Sahambi	*Weighting Function in Random Walk Based Left Ventricle Segmentation*	Não especificada	Método do Caminho Aleatório baseado em grafos. Modificação da função de ponderação aplicada ao Caminho Aleatório.	Sem resultados numéricos (Avaliação qualitativa das imagens)
2012	M. M. A. Hadhoud, M. I. Eladawy, A. Farag, F. M. Montevecchi, U. Morbiducci	*Left Ventricle Segmentation in Cardiac MRI Images*.	York Database	Endocárdio: Extração de 6 características e seleção (PCA) Classificador KNN Epicárdio: Detector de Canny em coordenadas polares. Convex Hull (cartesianas)	Endocárdio/ Epicárdio: S: 0.9561 ± 0.0625 / 0.9332 ± 0.0766 E: 0.9890 ± 0.0089 / 0.9849 ± 0.0092 Dice = 0.8937 ± 0.0747 / 0.9161 ± 0.0473
2012	C. Constantinidès, E. Roullot, M. Lefort, F. Frouin	*Fully Automated Segmentation of the Left Ventricle applied to cine MR Images: Description and Results on a Database of 45 Subjects*	Sunnybrook Database (45 *datasets*)	Filtro morfológico + *Snakes*	Endocárdio / Epicárdio: Dice: 0,86 ± 0,05 / 0,91 ± 0,03 APD (mm): 2,44 ± 0,56 / 2,80 ± 0,71 Bons contornos (%): 80 ± 16 / 71 ± 26

2012	M. G. Uzunbas, S. Zhang, K. M. Pohl, D. Metaxas, L. Axel	*Segmentation of Myocardium using Deformable Regions and Graphs Cuts*	Sunnybrook Database (15 *datasets*)	Cortes de grafos (*Ghaphs cuts*) *Level set*	Endocárdio / Epicárdio: Dice: 0,82 ± 0,06 / 0,91 ± 0,03 APD (mm): 2,98 ± 0,88 / 1,78 ± 0,35
2013	J. F. Dreijer B. M. Herbst J. A du Preez	*Left ventricular segmentation from MRI datasets with edge modelling conditional random fields*	York Database Sunnybrook Database (15 *datasets*)	Campo Aleatório Condicional	Endocárdio / Epicárdio: York Dice: 0,87 / 0,92 APD (mm): 2,70 / 2,23 Sunnybrook Dice: 0,91 / 0,93 APD (mm): 1,84 / 1,95
2014	M. Marino, F. Veronesi, C. Corsi	*Fully Automated Assessment of Left Ventricular Volumes and Mass from Cardiac Magnetic Resonance Images*	Imagens geradas usando um *scanner* Intera Achieva 1.5T (10 pacientes)	*Level set*	(ED) Endocárdio / Epicárdio: Dice: 0,93 ± 0,06 / 0,95 ± 0,03 Jaccard: 0,88 ± 0,09 / 0,90 ± 0,06 Hausdorff (pixels): 3,42 ± 0,46 / 3,72 ± 0,51 Hausdorff (mm): 1,71 ± 0,32 / 1,86 ± 0,39 (ES) Endocárdio / Epicárdio: Dice: 0,91 ± 0,04 / 0,89 ± 0,11 Jaccard: 0,83 ± 0,07 / 0,82 ± 0,16 Hausdorff (pixels): 3,43 ± 0,51 / 3,84 ± 0,55 Hausdorff (mm): 1,72 ± 0,35 / 1,94 ± 0,42
2014	J. Liu, X. Zhuang, J. Liu, S. Zhang, G. Wang, L. Wu, J. Xu, L. Gu	*Myocardium Segmentation Combining T2 and DE MRI using Multi-component Bivariate Gaussian Mixture Model*	Sequências de imagens DE e T2 de 6 pacientes depois do infarto do miocárdio.	Modelo de Mistura Gaussiana multicomponentes	Miocárdio (imagens DE) Dice: 0,643 ± 0,084 Miocárdio (imagens T2) Dice: 0,623 ± 0,129
2015	L. Wang, Y. Ma, K. Zhan, Y. Ma	*Automatic Left Ventricle Segmentation in Cardiac MRI via Level Set and Fuzzy C-Means*	Sunnybrook Database (7 *datasets*)	*Level set* Agrupamento *Fuzzy C-means*	Endocárdio: Dice: 0,9429 APD (mm): 1,2312 Bons contornos (%): 100

2016	P. V. Tran	*A Fully Convolutional Neural Network for Cardiac Segmentation in Short – Axis MRI*	Sunnybrook Database (30 *datasets*) LVSC Database RVSC Database	Técnica multi-resolução para extração de ROIs de vários tamanhos. Aumento dos dados de treinamento através de transformações afins. Treinamento de uma rede neural totalmente convolutiva com o SGD + momento. Refinamento (*finetuning*).	Endocárdio / Epicárdio: (30 *datasets* de Sunnybrook) Dice: 0,92 ± 0,03 / 0,96 ± 0,01 APD (mm): 1,73 ± 0,35 / 1,65 ± 0,31 Bons contornos (%): 98,48 ± 4,06 / 99,17 ± 2,20 Miocárdio (LVSC): Jaccard: 0,74 ± 0,13 S: 0,83 ± 0,12 E: 0,96 ± 0,03 PPV: 0,86 ± 0,10 NPV: 0,95 ± 0,03 Endocárdio / Epicárdio do ventrículo direito (RVSC): Dice: 0,84 ± 0,21 / 0,86 ± 0,20 Hausdorff (mm): 8,86 ± 11,27 / 9,33 ± 10,79

3 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos necessários para um melhor entendimento desta dissertação e está dividido em duas secções. Na primeira, serão abordados os princípios básicos da técnica de imageamento por ressonância magnética, bem como os planos de captura. As imagens utilizadas neste trabalho são provenientes da técnica *cinematic* MR, fato pelo qual é realizada uma breve descrição do processo de aquisição das imagens cardíacas a partir deste método. Estes temas são bastante extensos, sendo assim discutidos aqui apenas os principais aspectos.

Na segunda secção, serão introduzidas as redes neurais convolutivas, as quais são um ramo importante das técnicas de aprendizagem profunda (*deep learning*). Serão abordadas as características e funcionamento das principais camadas, a estratégia para evitar o sobreajuste da rede, bem como os diferentes algoritmos de otimização empregados nos treinamentos.

3.1 Imagens de Ressonância Magnética

3.1.1 Princípios básicos

A ressonância magnética está baseada no campo magnético direcional, ou momento, associado com partículas carregadas em movimento. O elemento mais abundante no corpo humano é o hidrogênio. Seu núcleo é composto por um próton, que possui spin (rotação em torno de seu próprio eixo) e, portanto, gera um momento magnético de dipolo. Normalmente os momentos magnéticos nucleares têm direção aleatória conforme ilustrado na Figura 9 (a). Devido a essa aleatoriedade, macroscopicamente não existe campo magnético. Quando o corpo do paciente é

colocado sob a ação de um forte campo magnético, que geralmente está entre 1,5 e 3 Tesla, os prótons de hidrogênio irão se orientar de acordo com a direção do campo magnético aplicado B_0 como se fossem pequenas bússolas, apontando tanto paralelamente quanto antiparalelamente ao campo, conforme ilustrado na Figura 9 (b). Um número maior de prótons se alinha paralelamente, entre 0,3 a 5 por milhão (BLINK, 2004).

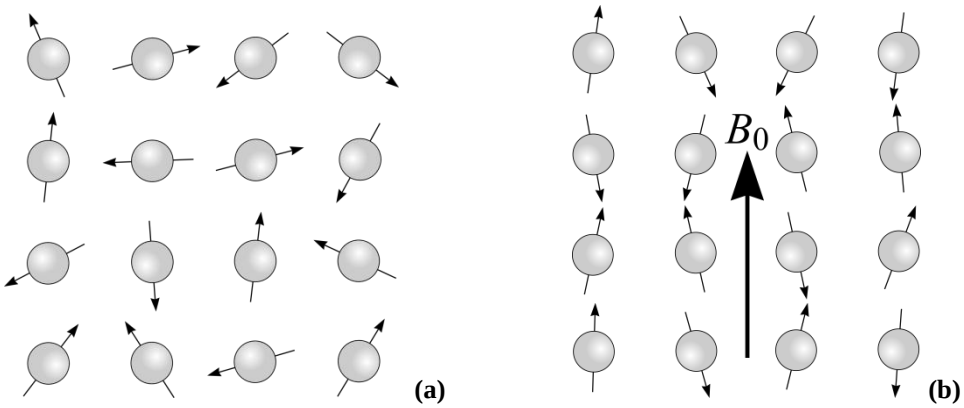


Figura 9. Orientações dos prótons (a) Na ausência de um campo magnético aplicado exteriormente, os momentos magnéticos possuem orientações aleatórias (b) Prótons de hidrogênio sob a ação do campo magnético externo B_0 alinhados paralela e antiparalelamente a ele.
Fonte: (PUDDEPHAT, 2002)

Como pode ser observado na Figura 10 (a), o eixo do momento magnético ou spin não se encontra exatamente alinhado com B_0, ele forma um ângulo Θ. Quando os prótons estão sob o efeito do campo magnético, os spins em rotação giram ao redor do eixo de B_0 realizando um movimento de cone conhecido como movimento de precessão, mostrado na Figura 10 (b). A frequência de precessão, chamada de frequência de Larmor, é proporcional à intensidade do campo magnético aplicado segundo a seguinte equação:

$$\omega_0 = \gamma B_0 \qquad (7)$$

em que, ω_0 é a frequência de Larmor, γ é a razão giromagnética e B_0 é a intensidade do campo magnético aplicado. A razão giromagnética é uma constante especifica para os núcleos. Para o hidrogênio, $\gamma \approx 42{,}6$ MHz/Tesla.

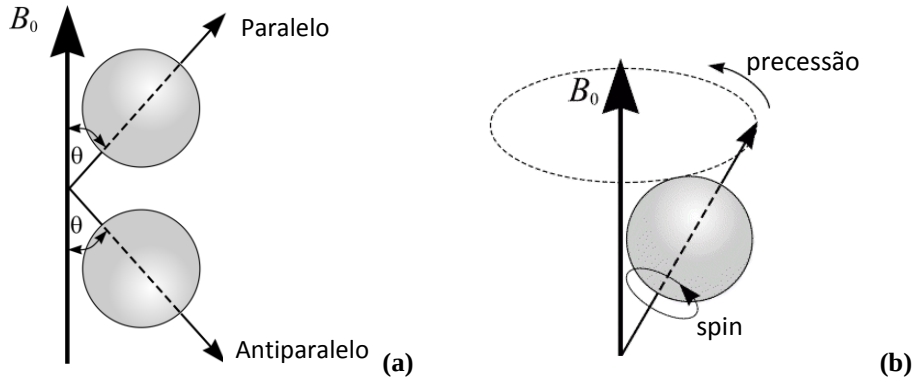


Figura 10. (a) Duas possíveis orientações dos prótons sob a ação de um campo magnético externo. (b) Spins em rotação giram ao redor do eixo de B_0 realizando um movimento de cone chamado precessão.
Fonte: (PUDDEPHAT, 2002)

Na Figura 11 e mostrada a representação vetorial dos momentos magnéticos de um conjunto de prótons. Cada vetor pode ser descrito pelas suas componentes perpendiculares e paralelas a B_0. Para um número suficientemente grande de spins distribuídos na superfície do cone, as componentes individuais perpendiculares a B_0 são canceladas, deixando apenas componentes na direção paralela a B_0. Como a maioria dos spins adota a orientação paralela, sua soma resultará no vetor M denominado como magnetização efetiva. Esse vetor é constante no tempo e assume direção e sentido iguais aos de B_0, uma vez que as componentes perpendiculares ao campo se anulam entre si como foi explicado já. A magnetização efetiva fornece os sinais para análise da ressonância magnética.

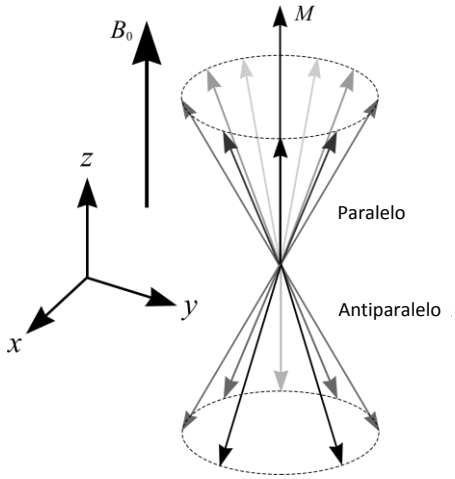


Figura 11. Representação vetorial dos momentos magnéticos. M é o vetor de magnetização resultante.
Fonte: (PUDDEPHAT, 2002)

Uma energia de radiofrequência (RF) à frequência de Larmor irradia-se perpendicularmente a B_0. Ela corresponde justamente à energia que o próton precisa absorver para que o momento magnético passe do estado paralelo para o antiparalelo. Isso tem um efeito sob o vetor de magnetização efetiva M que provoca sua inclinação até o plano x-y, conforme mostrado na Figura 12. O campo magnético associado à energia de RF é representado como B_1.

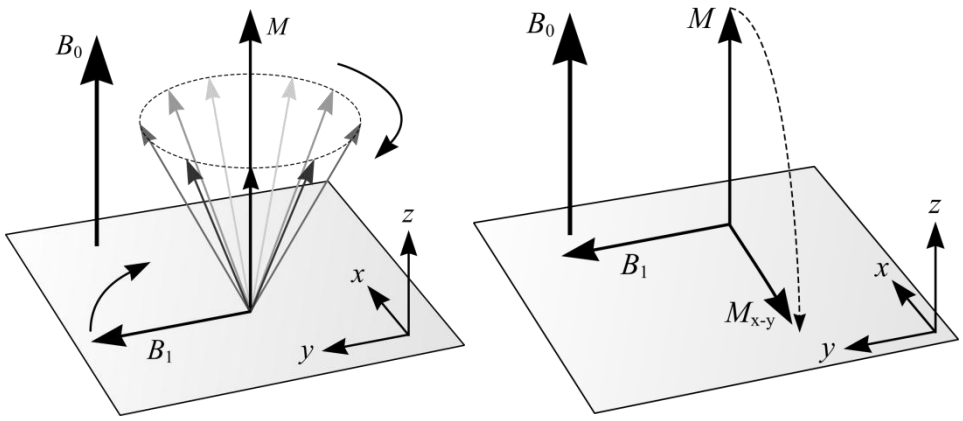


Figura 12. O efeito da radiação de RF na magnetização efetiva e produzir uma componente de M, $M_{x\text{-}y}$, ortogonal a B_0. B_1 é o campo magnético associado à energia de RF. O vetor M é inclinado da sua orientação original longitudinal ao eixo z, paralela ao campo magnético externo B_0, até o plano transversal x-y.
Fonte: (PUDDEPHAT, 2002)

Após a aplicação do pulso de RF, é possível medir o sinal $M_{x\text{-}y}$ utilizando uma bobina em quadratura como a mostrada na Figura 13 (a), que apresentará uma corrente induzida cuja amplitude realizará um sinal que decai com o tempo, conhecido como sinal de decaimento livre (FID, do inglês *Free Induction Decay*), mostrado na Figura 13 (b). O processo de decaimento é conhecido como relaxação. Isso ocorre, pois, ao fim do pulso de RF, os prótons retornam gradualmente para o seu estado inicial e de menor energia. Para efeitos de imagem em ressonância magnética, a informação obtida é ponderada pelo tempo que leva essa relaxação, acentuando o contraste de acordo com a interação (ligação) do átomo em relação à molécula e ao meio.

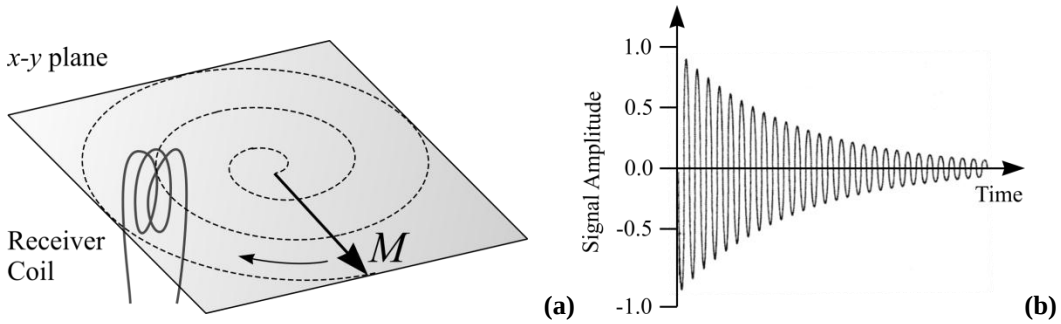


Figura 13. (a) Depois da aplicação de um pulso de RF de 90°, M se inclina para o plano x-y e precessa ao redor do eixo z. A componente $M_{x\text{-}y}$ decai com o tempo e pode ser captada com uma bobina situada perpendicular a B_0. (b) Amplitude do sinal de decaimento livre induzida na bobina. Fonte: (PUDDEPHAT, 2002)

Há dois tempos de relaxamento que devem ser considerados. O primeiro é o T_1, que se refere à relaxação spin-rede ou relaxação longitudinal, na qual o próton libera a energia e decai para o estado paralelo. T_1 é o intervalo de tempo entre o fim do pulso de RF e o aumento da componente paralela M até 63% do seu valor inicial. O segundo tempo é o T_2, que se refere à relaxação spin-spin ou relaxação transversal, na qual os prótons que estavam precessionando em fase retornam à distribuição uniforme. T_2 é o

intervalo de tempo entre o fim do pulso de RF e a diminuição da componente perpendicular M_{x-y} até 37% do valor inicial.

Os tecidos do corpo apresentam diferentes tempos de relaxação T_1 e T_2. Na Tabela 2 são apresentados os valores aproximados para alguns tecidos do corpo humano a 1,5 T. Estas diferenças são utilizadas para gerar contraste entre os tecidos nas imagens de ressonância magnética, o qual é uma vantagem de esta técnica de imageamento sobre as outras existentes. Em resumo, a intensidade do sinal de ressonância magnética depende de diversos parâmetros, incluindo a densidade de prótons, e tempos de relaxação T_1 e T_2.

Tabela 2 Tempos de relaxação T_1 e T_2 aproximados para diversos tecidos do corpo humano a 1,5 T.

Tecido	T1(ms)	T2(ms)
Substância branca	790	90
Substância cinzenta	920	100
Líquido cefalorraquidiano	4000	2000
Sangue (arterial)	1200	50
Miocárdio	870	50
Lipídios (gordura)	260	80

3.1.2 Ressonância magnética cardiovascular. Planos de captura.

A ressonância magnética cardiovascular oferece uma avaliação abrangente dos pacientes com insuficiência cardíaca e atualmente é a técnica de imagem “padrão ouro” para avaliar a anatomia do miocárdio, as funções regional e global e a viabilidade miocárdica. A informação derivada da CMRI muitas vezes revela a etiologia subjacente da insuficiência cardíaca e sua alta precisão de medição faz com que seja uma técnica

ideal para monitorar a progressão da doença e os efeitos do tratamento (KARAMITSOS et al., 2009).

Os dois sistemas de coordenadas principais usados para CMRI incluem os planos do corpo e os cardíacos. Os planos anatômicos estão orientados ortogonalmente ao eixo longo do corpo e são: axial, sagital e coronal, como se ilustra na Figura 14. Eles são utilizados para obter as imagens de exploração que fornecem uma descrição qualitativa da morfologia cardíaca (GINAT et al., 2011).

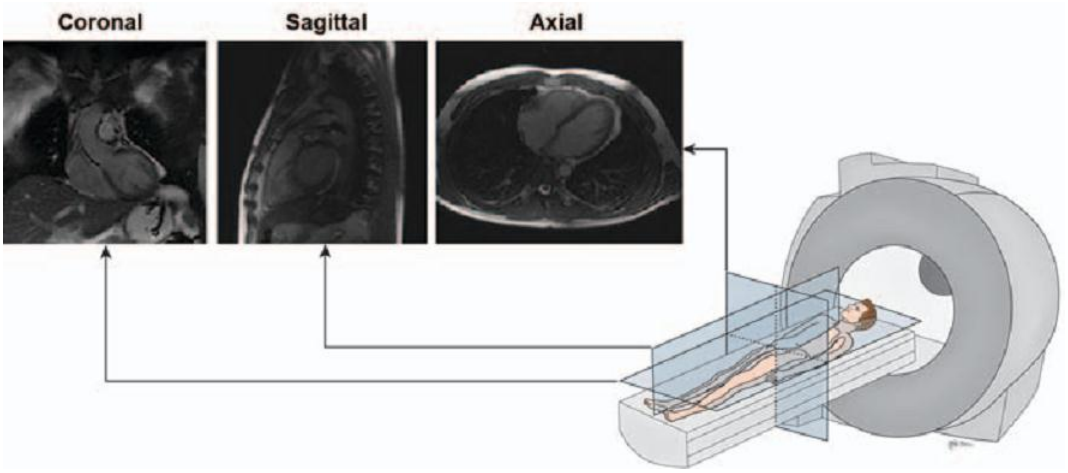


Figura 14. Orientação dos planos do corpo em relação ao paciente e suas correspondentes aparências em sequencias de imagem de sangue brilhante.
Fonte: (GINAT et al., 2011)

Através do plano axial são capturadas as quatro câmaras do coração e o pericárdio simultaneamente; o sagital mostra os grandes vasos que surgem na continuidade dos ventrículos e o coronal pode ser usado para avaliar a saída ventricular, o átrio esquerdo e as veias pulmonares. No entanto, a obliquidade (≈ 45°) destes planos às paredes do coração impede a caracterização anatômica e funcional com precisão. Por isso, estas informações devem ser obtidas a partir de planos cardíacos especializados.

Os planos cardíacos são definidos ao longo de uma linha que se estende desde o ápice cardíaco até o centro da válvula mitral (eixo longo do coração) usando as imagens do plano axial do corpo. Como pode ser observado na Figura 15 eles são: eixo curto (*short axis*), eixo longo horizontal (*horizontal long axis*) (vista de quatro câmaras) e eixo longo vertical (*vertical long axis*) (vista de duas câmaras).

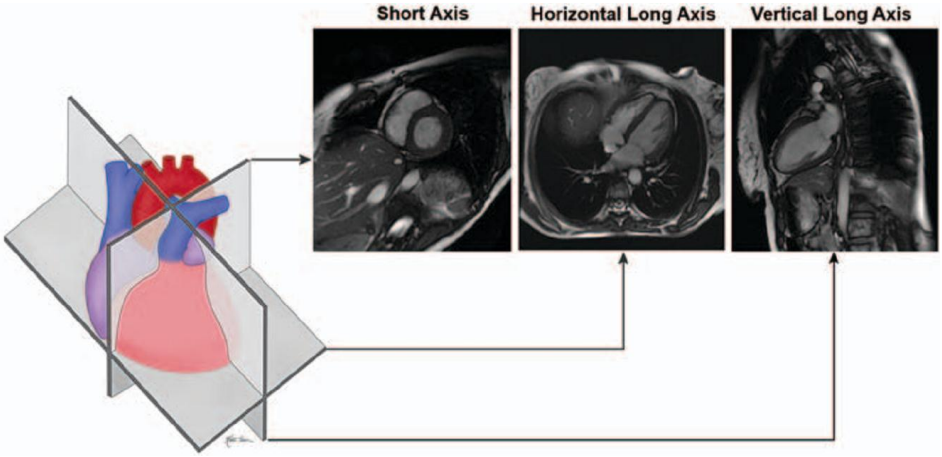


Figura 15. Orientação dos planos cardíacos: eixo curto, eixo longo horizontal e eixo longo vertical com respeito ao coração e suas aparências correspondentes.
Fonte: (GINAT et al., 2011)

O plano do eixo curto é obtido perpendicularmente ao eixo longo do coração, de forma que podem ser adquiridas imagens desde a base até o ápice. Isso permite avaliar o miocárdio em suas porções basal, medial e apical, o tamanho do ventrículo esquerdo, quantificar a função cardíaca e a contractilidade do ventrículo (NACIF et al., 2010).

O eixo longo horizontal é gerado selecionando o plano horizontal que é perpendicular ao eixo curto. Ele atravessa as quatro câmaras cardíacas, o que permite a avaliação do tamanho delas e da posição valvular. Também permite avaliar as válvulas mitral e tricúspide simultaneamente, entre outras estruturas anatômicas.

O eixo longo vertical está definido ao longo do plano vertical ortogonal ao eixo curto. Ele é paralelo ao plano sagital orientado através do eixo do coração. Pode ser

adquirido em duas direções, para analisar o trato de entrada do ventrículo esquerdo ou o trato de saída (GINAT et al., 2011).

Neste trabalho são utilizadas imagens do eixo curto, nas quais se quer segmentar o miocárdio no ventrículo esquerdo, sendo limitado pelos contornos do endocárdio e do epicárdio.

3.1.3 Método *cinematic* – MR

As imagens utilizadas nesta dissertação são provenientes de um método de imageamento denominado ressonância magnética cinemática no qual o coração é observado dinamicamente. Para cada posição de corte múltiplas imagens são adquiridas ao longo do intervalo de um batimento cardíaco. Desta forma, o funcionamento do coração pode ser visualizado través do espaço (correspondendo a fatias desde o ápice até a base) e do tempo (correspondendo a fases do ciclo cardíaco).

A aquisição das imagens é feita em diferentes pausas respiratórias do paciente e é sincronizada ao sinal de eletrocardiograma do paciente, tendo-se normalmente o pico da onda R como ponto de referência para a obtenção de dados.

3.2 Aprendizagem profunda: redes neurais convolutivas

Até há pouco tempo, a grande maioria das técnicas de aprendizagem de máquina e processamento de sinais exploravam arquiteturas de estruturas rasas (*shallow-structured architectures*). Essas arquiteturas contêm tipicamente uma única camada de transformações não lineares de características. Alguns exemplos delas são os convencionais Modelos Escondidos de Markov (HMM, do inglês *Hidden Markov*

Models), sistemas dinâmicos lineares ou não lineares, campos aleatórios condicionais, modelos de entropía máxima, Máquinas de Vetores de Suporte (SVMs, do inglês *Support Vector Machines*), regressão logística e redes neurais com apenas uma camada escondida. A propriedade comum delas é que a arquitetura consiste de uma única camada responsável por transformar os sinais de entradas em um problema de espaço de características, que pode ser inobservável. Elas têm demonstrado ser eficazes na resolução de muitos problemas simples ou com restrições, mas a sua modelagem e poder de representação limitada podem causar dificuldades ao lidar com aplicações mais complexas, por exemplo: sinais de fala humana, sons naturais, imagens e cenas visuais. Os mecanismos de processamento de informação humana (visão e fala) sugerem, porém, a necessidade de arquiteturas profundas para extrair estruturas complexas e construir representações internas a partir das entradas sensoriais (DENG, 2011).

Os sistemas de processamento de sinais com arquiteturas profundas são compostos por várias camadas de transformações não lineares, onde a saída de cada camada alimenta a entrada da camada imediata posterior. Assim, a aprendizagem profunda, também conhecida como aprendizagem hierárquica tem emergido como uma nova área na pesquisa de aprendizado de máquina (HINTON; SALAKHUTDINOV, 2006). É formada por um conjunto de algoritmos que tentam modelar abstrações de alto nível nos dados, utilizando uma arquitetura como a mencionada anteriormente. A aprendizagem profunda encontra-se na interseção entre as áreas de pesquisa de redes neurais, modelagem gráfica, otimização, reconhecimento de padrões e processamento de sinais. Ela exige o uso de recursos computacionais consideráveis. A redução significativa do custo do *hardware* dos computadores e o incrível aumento das capacidades de processamento dos *chips*, por exemplo, das unidades de processamento

gráfico (GPU, do inglês *Graphics Processing Units*) são duas importantes razões para a popularidade atual da aprendizagem profunda (DENG, 2011).

Várias arquiteturas de aprendizagem profunda tais como as redes neurais profundas, redes neurais convolucionais, redes de crenças profundas (DBN, do inglês *Deep Belief Networks*) e redes neurais recorrentes têm sido aplicadas em áreas como a visão computacional (CIREŞAN; MEIER; SCHMIDHUBER, 2012), reconhecimento automático da voz (FERNÁNDEZ; GRAVES; SCHMIDHUBER, 2007), processamento da linguagem natural (MESNIL et al., 2015), recuperação da informação (WAN et al., 2014) e bioinformática (CHICCO; SADOWSKI; BALDI, 2014) onde demonstraram produzir resultados do estado da arte em várias tarefas.

3.2.1 Características principais das redes neurais convolutivas

As CNNs são variantes do bem conhecido perceptron multicamadas (MLP, do inglês *Multi Layer Perceptron*) e estão inspiradas biologicamente segundo a arquitetura multiestado de Hubel-Wiesel (HUBEL; WIESEL, 1962). No trabalho citado anteriormente, os autores demostraram que os gatos contêm arranjos complexos de células no seu córtex visual. Essas células são sensíveis a pequenas sub-regiões do campo visual chamadas campo receptivo, e atuam como filtros locais que aproveitam a correlação espacial presente nas imagens naturais. Existem dois tipos de células: as simples, que têm uma resposta máxima ante bordas de orientação seletiva dentro do seu campo receptivo, similar aos filtros convolucionais; e as complexas, que têm grandes campos receptivos e são localmente invariantes à posição exata do padrão.

Estas redes têm suas raízes no *Neocognitron* (FUKUSHIMA; MIYAKE, 1982) que usava uma arquitetura similar, mas não tinha um algoritmo de aprendizado supervisionado de extremo a extremo (*end-to-end*). Este modelo foi melhorado depois por Yann LeCun e colaboradores (Y. LECUN et al., 1998) ao introduzir um método de aprendizado baseado no *Backpropagation.* No ano 2011 foram refinadas por Ciresan e colaboradores (2011) e implementadas em uma GPU obtendo resultados impressionantes.

As CNNs combinam três ideias principais, que além de ser a base do seu funcionamento, garantem certo grau de imunidade à translação, à rotação e conferem robustez. Elas são: (1) filtros locais receptivos, (2) pesos compartilhados e (3) sub-amostragem (NIELSEN, 2016).

- Filtros locais receptivos

Nas CNNs, os neurônios de uma camada estão conectados somente a uma região local do volume de entrada. A extensão espacial dessa conectividade em termos de largura e comprimento é um hiperparâmetro do neurônio chamado campo receptivo; e ao longo do eixo de profundidade é sempre igual à profundidade do volume de entrada. Por exemplo, na Figura 16 se ilustra um volume de entrada de 32 × 32 × 3, isto é, uma imagem com três canais RGB. Se for suposto um campo receptivo (ou tamanho do filtro) de 5 × 5, então cada neurônio na camada convolutiva teria 5 × 5 × 3 = 75 pesos mais um valor de polarização (*bias*). Além disso, se o passo de deslocamento do filtro (daqui para frente mencionado com *stride*) for 1, vão existir 28 × 28 neurônios na camada. Isto é porque só podemos mover o campo receptivo 27 vezes antes de colidir com o lado direito (ou com a parte inferior) da imagem de entrada. Mais à frente serão mostradas as equações para chegar a esse valor.

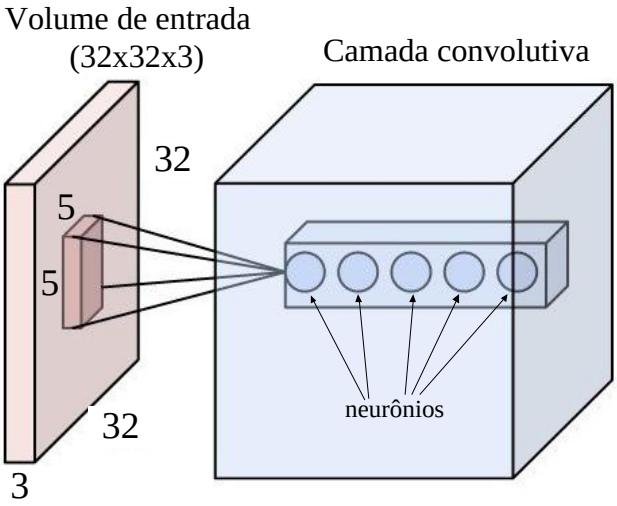


Figura 16. O volume azul representa uma camada convolutiva com 5 planos de neurônios onde cada um deles tem um campo receptivo de 5x5 em relação ao volume de entrada de tamanho 32x32x3. Cada neurônio vai ter 5x5x3 pesos mais um valor de polarização.
Fonte: (NG et al., 2015)

Com este esquema, cada unidade é insensível a variações fora do seu campo receptivo. A arquitetura garante, assim, que os filtros aprendidos produzam uma resposta forte a um padrão de entrada espacialmente local. No entanto, empilhar muitas camadas conduz a filtros não lineares que se tornam cada vez mais "globais", isto é que respondem a uma região maior de espaço de pixels.

- Pesos e polarizações compartilhados

Continuando com o exemplo da figura anterior, esses 28 × 28 neurônios da camada utilizarão os mesmos pesos e polarizações e vão formar um "mapa de características". Em outras palavras, para o *j*, *k*-ésimo neurônio a saída, *y*, é:

$$y = \sigma\left(b + \sum_{l=0}^{4}\sum_{m=0}^{4} \boldsymbol{W}_{l,m} a_{j+l,k+m}\right) \tag{8}$$

em que, σ é a função de ativação, b é o valor de polarização compartilhado, $\boldsymbol{W}_{l,m}$ é uma matriz 5 × 5 de pesos compartilhados e $a_{j+l,k+m}$ denota a ativação de entrada na posição $(j + l)$, $(k + m)$.

Na Figura 17 são mostradas as ligações entre neurônios de duas camadas diferentes. Os pesos da mesma cor são denominados "pesos compartilhados" por serem idênticos.

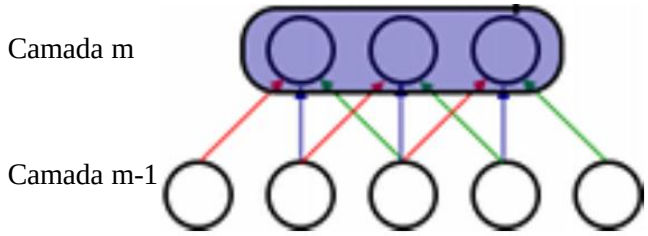


Figura 17. Ilustração de duas camadas de neurônios. Os pesos da mesma cor que ligam os neurônios da camada "m-1" com os da camada "m" são chamados "pesos compartilhados" por serem iguais.
Fonte: (NG et al., 2015)

A partilha de pesos e polarizações implica que os neurônios de uma determinada camada vão detectar exatamente a mesma característica (um tipo de padrão que causa sua ativação, por exemplo, uma borda na imagem) só que em diferentes localizações na imagem de entrada, independentemente da sua posição. Esta propriedade confere às CNNs a capacidade de ser invariantes à translação de imagens. Além disso, se reduz o número de parâmetros livres a ser aprendidos provocando um incremento da eficiência do aprendizado. Geralmente são usados vários mapas de características de forma que a rede pode detectar diferentes tipos de padrões.

- Sub-amostragem

As CNNs contêm camadas de sub-amostragem (*pooling layers*) que serão analisadas nas próximas secções. Elas geralmente são colocadas depois das camadas de convolução para simplificar a informação de saída, reduzindo o tamanho espacial da representação e a quantidade de parâmetros e cálculos na rede. Também ajudam a controlar o sobreajuste da rede.

3.2.2 Camadas das redes neurais convolutivas

A seguir é analisado o funcionamento e os parâmetros das principais camadas das CNNs.

Camada de Convolução

As camadas de convolução são o núcleo de uma CNN e fazem a maior parte do trabalho computacional. Seus parâmetros são um conjunto de filtros treináveis. Cada filtro é sensível a uma característica particular que pode ser, por exemplo, uma borda vertical ou horizontal, entre outras. A saída de cada filtro é conhecida como mapa de característica. Uma camada convolutiva tem vários mapas de características, produzidos por planos de neurônios, cada plano implementa um filtro diferente. Geralmente, os filtros são pequenos espacialmente (ao longo da largura e altura), mas estende-se através de toda a profundidade do volume de entrada. Na Figura 18 são mostrados os mapas de características gerados pelos 96 filtros aprendidos na primeira camada convolutiva da arquitetura proposta por Krizhevsky, Sutskever e Hinton (2012). A referida arquitetura de classificação de imagens é composta por cinco camadas convolutivas e três de

conexão total (conhecidas em inglês como *fully connected*). Conforme mostra a referida imagem, os filtros são seletivos a orientação, frequência e *blobs* coloridos.

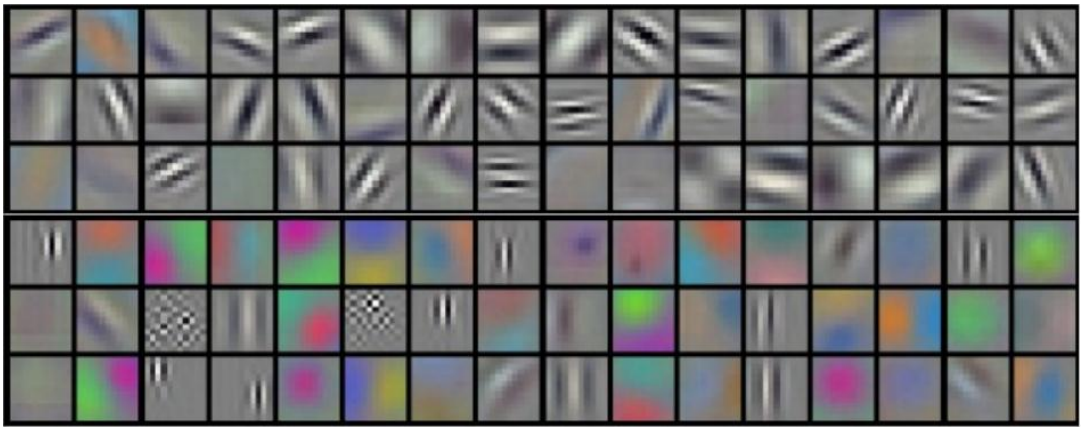


Figura 18. Exemplos de mapas de características seletivos a orientação, frequência e cor, gerados por 96 filtros de uma camada convolutiva.
Fonte: (KRIZHEVSKY, SUTSKEVER, HINTON, 2012)

A camada de convolução funciona deslizando cada filtro em toda a largura e altura do volume de entrada, produzindo um mapa de ativação bidimensional desse filtro. A medida que o filtro se desloca, através da entrada, é calculado o produto entre os pesos do filtro e os pixels correspondentes da entrada, como é expressado na equação 9:

$$g(x,y) = \sum_{l=0}^{n}\sum_{m=0}^{n} \boldsymbol{W}(l,m) \cdot f(x+l, y+m) \tag{9}$$

Assim, o valor de cada pixel com posição (*x*, *y*) da nova imagem *g* é obtido mediante a somatória do produto dos pesos do filtro $\boldsymbol{W}$ (de largura e altura *n*) pelos valores dos pixels correspondentes à posição do filtro na imagem de entrada *f*. Esse processo é conhecido como convolução espacial.

Através das iterações do treinamento, a rede vai aprender os pesos dos filtros, que vão se ativar quando virem algum tipo específico de característica em alguma

posição espacial na entrada. O volume de saída é formado empilhando os mapas de ativações de todos os filtros ao longo da dimensão de profundidade. Cada valor no volume de saída pode, portanto, também ser interpretado como uma saída de um neurônio que "olha" apenas em uma pequena região da entrada e compartilha parâmetros com outros neurônios no mesmo mapa de ativação, como foi explicado na subsecção anterior.

O volume de saída é calculado em função: do volume de entrada, do número de filtros (*K*), cada um aprendendo para detectar alguma característica na entrada; do tamanho do campo receptivo dos neurônios (*F*), do *stride* (*S*) que representa o quanto vai se deslocar os filtros, e da quantidade de zeros usados para preenchimento nas bordas, referido de aqui para frente como *zero padding* (*P*). Às vezes é conveniente usar o *zero padding* com o intuito de preservar o tamanho do volume de entrada na saída, de forma que tenham a mesma largura e altura. Em termos gerais, a partir de um volume de entrada, expressado em termos de largura ($W_{entrada}$), altura ($H_{entrada}$) e profundidade ($D_{entrada}$), é produzido um volume de saída $W_{saída} \times H_{saída} \times D_{saída}$ em que:

$$W_{saída} = \frac{W_{entrada} - F + 2P}{S} + 1\,, \tag{10}$$

$$H_{saída} = \frac{H_{entrada} - F + 2P}{S} + 1\,, \tag{11}$$

$$D_{\text{saída}} = K \tag{12}$$

Na Figura 19 são mostrados dois exemplos gráficos da convolução em apenas uma dimensão (eixo *x*). Nos dois casos o tamanho de entrada $W = 7$, o campo receptivo

F = 3 e o *zero padding* P = 0, mas em (a) o *stride* S = 1 e em (b) S = 2. Os pesos do filtro são [1, 0, -1] e o *bias* é 0.

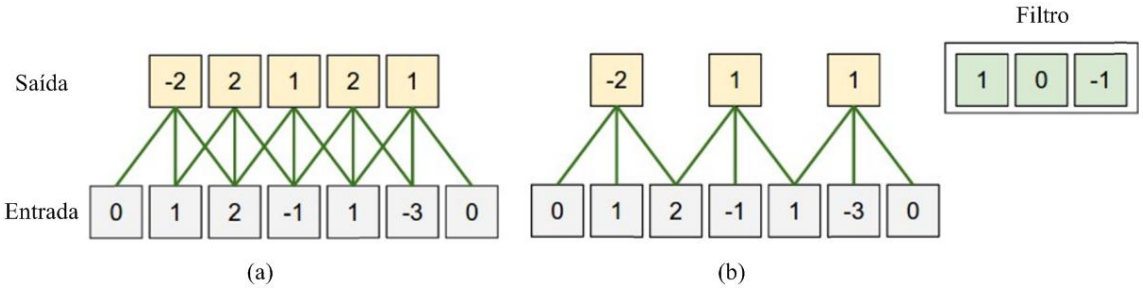


Figura 19. Exemplos de saídas obtidas usando um neurônio com F = 3, tamanho de entrada W = 7, P = 0 e (a) S = 1 (b) S = 2.
Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016)

Na Figura 19 (a) o filtro é deslocado através da entrada com S = 1, resultando em uma saída de tamanho (7 - 3 + 0)/1+1 = 5. Em (b) o *stride* é 2, portanto, a saída é de tamanho (7 - 3 + 0)/2+1 = 3. Um *stride* igual a 3 não poderia ser utilizado, uma vez que o filtro não se encaixa perfeitamente em todo o volume. Em termos da equação, isto pode ser determinado devido a que (7 - 3 + 0) = 4 não é divisível por 3. Essa limitação deve ser levada em conta na hora de projetar a rede.

Em resumo:

- A camada convolutiva requer quatro hiperparâmetros:

  (1) K: Número de filtros, que dá a quantidade de mapas de características.

  (2) F: Campo receptivo, isto é a extensão espacial dos filtros.

  (3) S: *Stride*

  (4) P: *Zero padding*

- Compartilhando os parâmetros dos neurônios em um plano, são introduzidos $F \times F \times D_{\text{entrada}}$ pesos por filtro, totalizando $(F \times F \times D_{\text{entrada}}) \times K$ pesos e K *biases*.

- No volume de saída, o *d*-ésimo plano de profundidade (de tamanho $W_{saída}$ × $H_{saída}$) é o resultado da execução da convolução do *d*-ésimo filtro sobre o volume de entrada com um *stride S*, mais a soma do *d*-ésimo *bias*.

Inicialização dos pesos

No trabalho com redes neurais profundas, a inicialização da rede com pesos adequados pode fazer a diferença entre a convergência da rede numa quantidade de tempo razoável e a não convergência, quando a função de perda não diminui e a rede realiza infinitas iterações sem obter resultado.

Se os pesos são muito pequenos, a variância do sinal de entrada começa a diminuir assim que passam através de cada camada na rede. A entrada eventualmente irá caindo a valores realmente baixos e não poderá ser mais útil. Se os pesos são muito grandes, então a variância dos dados de entrada tende a aumentar rapidamente com cada passagem pelas camadas e, eventualmente, tornam-se inúteis.

Para garantir que a rede vai funcionar corretamente, faz-se necessário certificar-se de que os pesos estejam em um intervalo razoável, antes de começar o treinamento. Este processo é aleatório e em muitas ocasiões é feito segundo uma distribuição gaussiana com média zero e uma determinada variância. Consideremos um neurônio linear:

$$y = w_1 x_1 + w_2 x_2 + \ldots + w_k x_k \tag{13}$$

em que, $\boldsymbol{x_k}$ é o vetor de entradas ao neurônio, $\boldsymbol{w_k}$ é o vetor de pesos, *b* é o *bias* e *y* é a saída estimada.

Em cada passagem pelas camadas, é desejável que permaneça a mesma variância nos pesos. Isso vai ajudar a evitar que os pesos se tornem muito grandes ou muito pequenos. Um processo de inicialização com estas características é conhecido como inicialização de Xavier (GLOROT; BENGIO, 2010)**.**

Calculando a variância na equação 13, temos que:

$$var(y) = var(w_1 x_1 + w_2 x_2 + \ldots + w_k x_k + b) \quad (14)$$

Calculando a variância dos termos dentro dos parênteses no lado direito da equação acima temos:

$$var(w_i x_i) = E(x_i)^2 var(w_i) + E(w_i)^2 var(x_i) + var(w_i) var(x_i) \quad (15)$$

Como *b* é uma constante, tem variância zero. *E*(·) representa o valor esperado de uma variável dada, ou seja, o valor médio. Assumindo que as entradas e pesos possuem uma distribuição gaussiana de média zero, o termo *E*(·) pode ser eliminado:

$$var(w_i x_i) = var(w_i) var(x_i) \quad (16)$$

Então a equação 16 pode ser escrita como:

$$var(y) = var(w_1) var(x_1) + \cdots + var(w_k) var(x_k) \quad (17)$$

Uma vez que os termos têm distribuições iguais, podemos escrever:

$$var(y) = Nvar(w_i)var(x_i) \tag{18}$$

Então se quisermos que a variância de *y* seja a mesma que *x*, então o termo (*N* var (w_i)) deve ser igual a 1. Assim:

$$Nvar(w_i) = 1 \quad \Rightarrow \quad var(w_i) = \frac{1}{N} \tag{19}$$

em que, *N* é o número de neurônios de entrada.

A equação 19 é a formula da implementação prática da inicialização de Xavier, que consiste em escolher os pesos a partir de uma distribuição gaussiana com média zero e variância $\frac{1}{N}$.

Camada de *Pooling*

Nas arquiteturas das CNNs é comum a inserção de uma camada de sub-amostragem (*pooling*) entre as camadas consecutivas de convolução. A sua função é reduzir progressivamente o tamanho espacial da representação e consequentemente reduzir a quantidade de parâmetros e cálculos na rede, além de controlar o sobreajuste, conhecido também com o termo em inglês de *overfitting*, conceito que será analisado proximamente.

A camada de *Pooling* opera de forma independente sobre cada fatia através de toda a profundidade. Conforme é ilustrado na parte superior da Figura 20, ela dá um novo tamanho ao volume de entrada, mantendo a dimensão de profundidade (*D*) sem mudanças, usando uma operação determinada.

As operações mais comuns são MAX (toma o máximo dos valores contidos numa janela) e MÉDIA (calcula a média dos valores contidos numa janela), embora a primeira seja a mais utilizada nas aplicações.

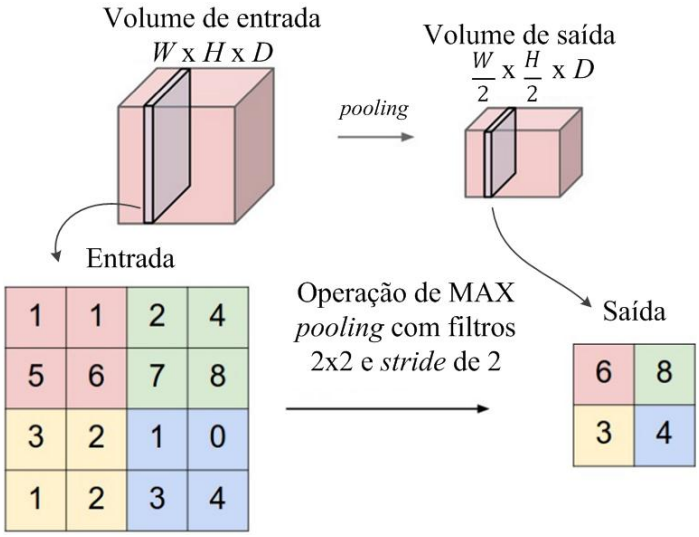


Figura 20. Um filtro 2x2 desloca-se na imagem com stride de 2, de forma que dos 4 elementos da janela é escolhido o de maior valor.
Fonte: (NG et al., 2015)

Geralmente, são utilizadas janelas de tamanho 2 × 2 que são deslocadas com um *stride* de 2. No caso da operação MAX, toma-se o máximo entre 4 números. Desta forma faz-se uma sub-amostragem de 2 ao longo da largura e do comprimento de cada fatia da entrada, eliminando 75 % das ativações. Este processo é mostrado na parte inferior da Figura 20.

A camada de *Pooling* requer dois hiperparâmetros:

(1) *F*: extensão espacial da janela

(2) *S*: *Stride*

Em termos gerais, a partir de um volume de entrada $W_{entrada} \times H_{entrada} \times D_{entrada}$, é produzido um volume de saída em que:

$$W_{saída} = \frac{W_{entrada} - F}{S} + 1 \quad (20)$$

$$H_{saída} = \frac{H_{entrada} - F}{S} + 1 \quad (21)$$

$$D_{saída} = D_{entrada} \quad (22)$$

O uso do *zero padding* não é comum. Esta camada não aprende durante o treinamento, ela realiza operações pré-definidas sobre os elementos.

Vale a pena mencionar que existem apenas duas variações da camada MAX *Pooling* encontradas na prática: (1) camada com $F = 3$, $S = 2$ (também chamada *Pooling* de sobreposição), e mais comumente (2) $F = 2$, $S = 2$. Tamanhos de F maiores podem ser destrutivos e não aportar nenhum benefício ao desempenho da rede (KARPATHY, 2016).

Camada de Unidades Lineares Retificadoras

A forma padrão para modelar a saída f de um neurônio em função das suas entradas x é através da função tangente hiperbólica $f(x) = \tanh(x)$, mostrada na Figura 21 ou da função sigmoide $f(x) = (1+e^{-x})^{-1}$, mostrada na Figura 22. No entanto, em termos de tempo de treinamento com o algoritmo do gradiente descendente estocástico, essas não linearidades de saturação são muito mais lentas do que a linearidade não saturante $f(x) = \max(0, x)$, mostrada na Figura 23 (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

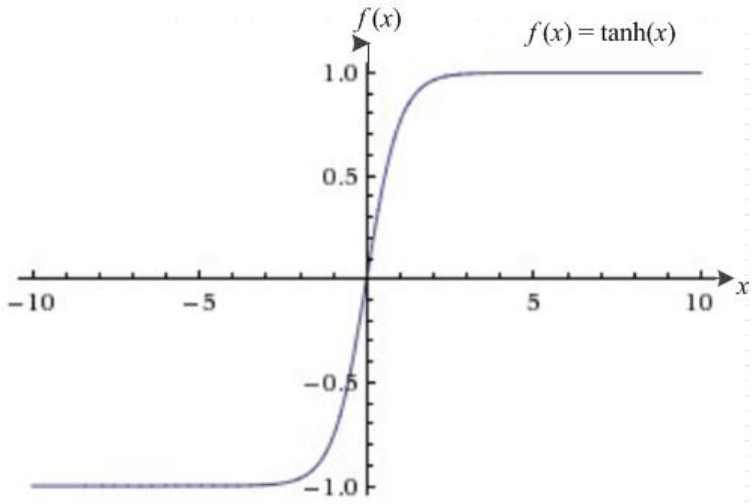


Figura 21. Função de ativação tangente hiperbólica
Fonte: (NG et al., 2015)

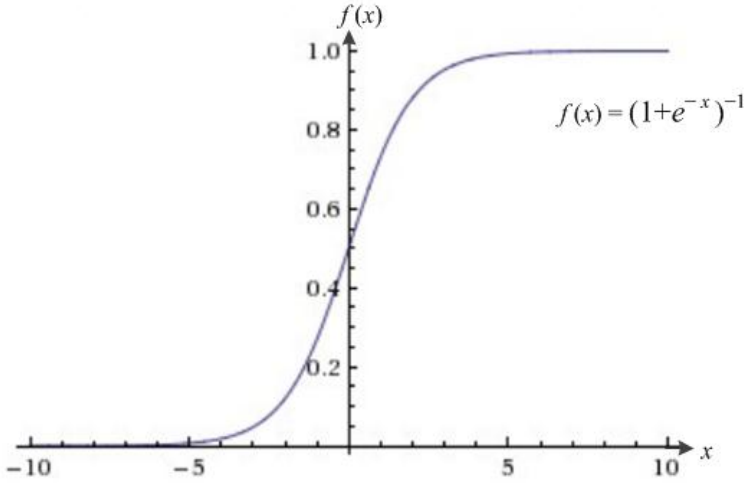


Figura 22. Função de ativação sigmoide
Fonte: (NG et al., 2015)

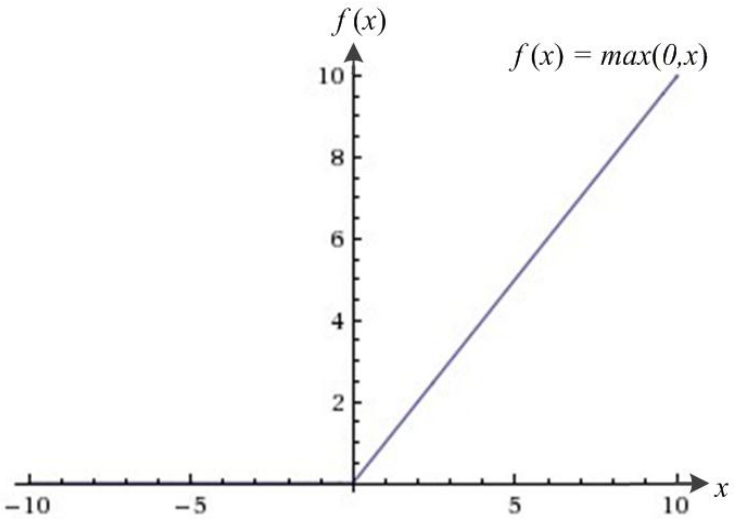


Figura 23. Função de ativação linear não saturada ReLU
Fonte: (NG et al., 2015)

Segundo Nair e Hinton (2010), os neurônios que aplicam a função $f(x) = \max(0, x)$, são referidos como Unidades Lineares Retificadoras (ReLUs, do inglês *Rectified Linear Units*).

As CNN que utilizam as camadas de ReLUs treinam várias vezes mais rápido do que seus equivalentes com unidades *tanh* ou sigmoide devido a que estas unidades aceleram a convergência da rede. Na Figura 24 é mostrado o número de iterações necessárias para atingir 25 % de erro no treinamento com o conjunto de dados CIFAR-10 para uma rede convolutiva de quatro camadas. Pode ser observado que com as funções ReLUs (linha sólida) esse valor é atingido seis vezes mais rapidamente do que com uma rede equivalente usando a função *tanh* (linha tracejada), ou seja, é acelerada a convergência.

A magnitude do efeito mostrado aqui varia com a arquitetura de rede, mas com redes que usam ReLUs a aprendizagem é várias vezes mais rápida do que com neurônios de saturação. Também seu uso evita o problema do desvanecimento do gradiente quando existem muitas camadas. Isto se deve ao fato de $f(x) = \max(0, x)$ ser uma função linear e não ter saturação no sentido positivo do seu domínio. Outra

vantagem da função da ativação em questão é que pode ser implementada de forma mais simples que a sigmoide e a *tanh*, bastando limiarizar uma matriz de ativações com respeito ao zero. A camada ReLU não aprende durante o treinamento, ela apenas aplica uma função fixa.

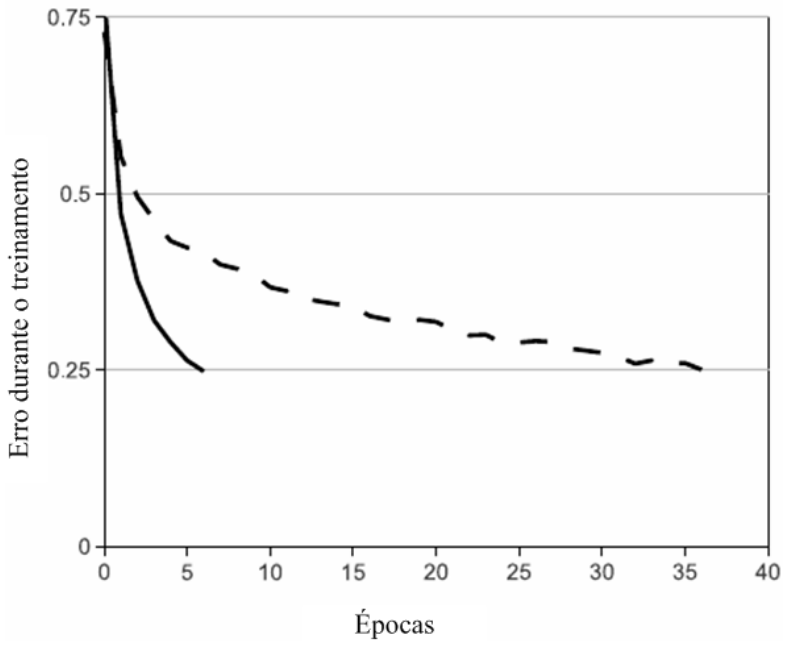


Figura 24. Erros durante o treinamento em uma CNN usando funções de ativação ReLU (linha sólida) e funções tangente hiperbólica (linha tracejada).
Fonte: (KRIZHEVSKY; SUTSKEVER; HINTON, 2012)

Uma desvantagem das unidades ReLUs é que podem ser frágeis durante o treinamento e "morrer". Se o gradiente move os pesos de forma que os neurônios não se ativem para nenhum exemplo de treinamento, então o gradiente sempre será zero para esse neurônio e nunca vai se ativar. Se isso acontecer, então o gradiente que flui através do neurônio será sempre zero a partir desse ponto. Ou seja, as unidades ReLUs podem "morrer" irreversivelmente durante o treinamento. Às vezes podem "morrer" até 40 % dos neurônios se a taxa de aprendizado é muito alta. No entanto, quando a taxa de aprendizado é adequada este problema é pouco frequente.

Camada de *Dropout*

As redes neurais profundas contêm múltiplas camadas escondidas não lineares, as quais as tornam modelos expressivos que podem aprender relações muito complicadas entre suas entradas e saídas. No entanto, é comum que os dados apresentem desvios causados por erros de medição ou fatores aleatórios, de modo que existirão no conjunto de treino, mas não em dados de teste reais (SRIVASTAVA et al., 2014). Além disso, se o conjunto de dados é pequeno e limitado, a rede pode se-ajustar demasiado, de modo que só irá a reconhecer os padrões com os quais foi treinada e não vai conseguir uma boa generalização. Os dois casos mencionados conduzem ao sobreajuste da rede, mais conhecido com o termo em inglês *overfitting*.

Um modelo sobreajustado apresenta alta precisão quando testado com seu conjunto de dados, porém tal modelo não é uma boa representação da realidade e por isso deve ser evitado. Na Figura 25 se ilustra um exemplo gráfico que ajuda a entender este fenômeno. Os pontos vermelhos pertencem a uma classe e os azuis a outra. Se para classificar, a rede segue a linha verde, vai ficar tão adaptada a esses padrões que ante novos dados vai apresentar grandes erros. No casso da classificação seguindo a linha preta a rede vai conseguir generalizar mais.

Uma técnica para contornar este problema é a regularização, que adiciona à função de custo um termo de penalidade, visando obter um modelo mais representativo da realidade. Através da validação cruzada, na qual o modelo é testado em relação a uma parte reservada do conjunto de dados que não foi utilizada no treinamento, é possível se ter uma ideia de se o modelo sofre de sobreajuste ou não.

Um método de regularização simples, mas efetivo, é introduzir penalidades, que podem ser de tipo L1 ou L2, aos pesos da rede (SRIVASTAVA et al., 2014). Outras

formas comuns incluem a parada antecipada, a regularização bayesiana, a eliminação de pesos e o aumento dos dados (WU; GU, 2015).

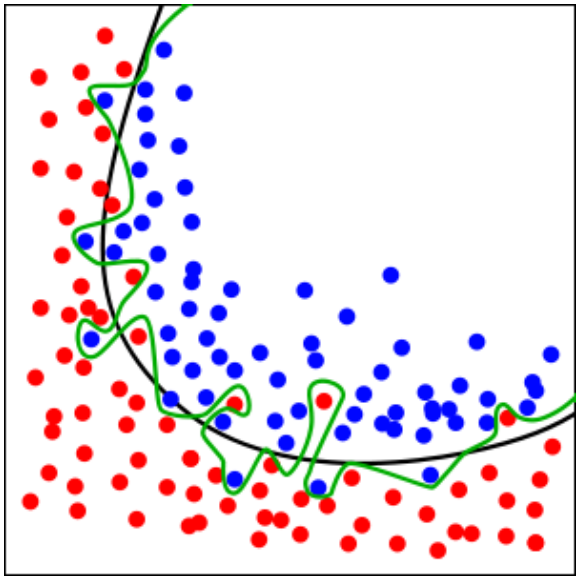


Figura 25. Sistema de predição de duas classes em que a linha verde representa um modelo sobreajustado e a linha preta um modelo regularizado.
Fonte: (SRIVASTAVA et al., 2014)

O *dropout* (HINTON et al., 2012) é um método de regularização para enfrentar o *overfitting* que tem sido empregado recentemente na aprendizagem profunda. Ele evita que se produzam co-adaptações com os dados de treinamento. O termo "*dropout*" refere-se à omissão das unidades (escondidas e visíveis) numa rede neural. Na Figura 26 é mostrado este conceito: em cada apresentação de cada um dos casos do treinamento, uma unidade pode ser removida temporariamente da rede juntamente com todas as suas conexões de entrada e saída. A escolha de qual unidade será removida é aleatória. No caso mais simples, cada unidade é retida com uma probabilidade fixa *p* independente das outras unidades, em que o valor de *p* pode ser selecionado usando um conjunto de validação ou simplesmente pode ser estabelecido como sendo 0,5, o que parece ser um valor ótimo para uma ampla gama de redes e tarefas. No entanto, para as unidades de

entrada, a probabilidade ótima de retenção é usualmente mais perto de 1 que de 0,5 (SRIVASTAVA et al., 2014).

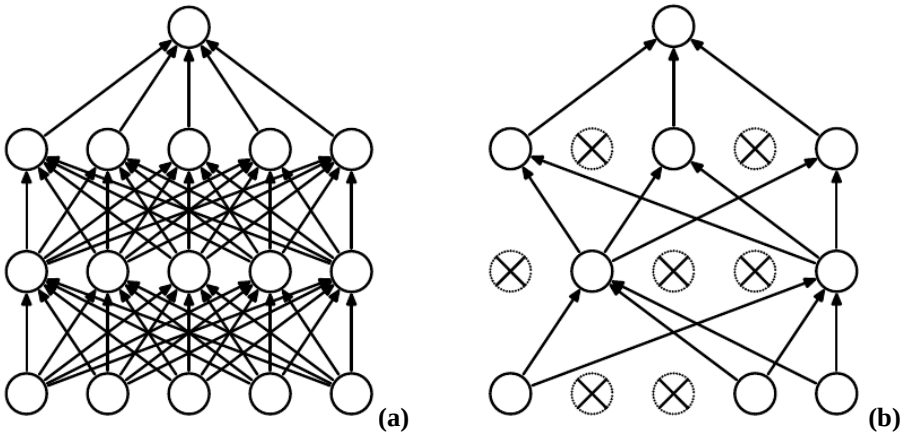


Figura 26. Modelo *Dropout* nas redes neurais. (a) Rede neural padrão com 2 camadas escondidas (b) Aplicação de *Dropout* à rede em (a). As unidades cruzadas foram retiradas. Fonte: (SRIVASTAVA et al., 2014)

Uma rede neural com n unidades pode ser vista como uma coleção de 2^n possíveis redes neurais simplificadas com *dropout*. Essas redes compartilham todos os pesos de forma que o número total de parâmetros é $O(n^2)$, ou menor. Como foi dito, para cada apresentação de cada caso de treinamento, é treinada uma rede sob o efeito do *dropout*. Assim, o treinamento de uma rede utilizando este método pode ser visto como treinar uma coleção de 2^n redes simplificadas com partilha de pesos, em que cada rede com *dropout* foi treinada, pelo menos uma vez.

Na hora do teste, não é viável calcular a média das predições a partir de cada um dos modelos simplificados. No entanto, existe um método de média aproximada muito simples, que funciona bem na prática. A ideia é usar uma única rede neural sem *dropout* na hora do teste. Os pesos desta rede são versões de escala reduzida dos pesos treinados. Se uma unidade é retida, com uma probabilidade p durante o treino, os pesos de saída

dessa unidade são multiplicados por *p* no momento do teste, conforme mostrado na Figura 27.

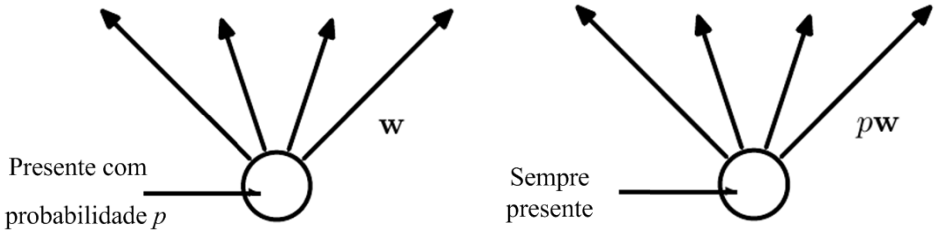


Figura 27. Forma de aplicação do *Dropout* na etapa de (a) treinamento (b) teste
Fonte: (SRIVASTAVA et al., 2014)

Na Figura 27 (a) é mostrada uma unidade que está presente no treinamento com probabilidade *p* e está conectada a outras unidades na próxima camada com pesos ***W***. Na Figura 27 (b) é mostrado que, na etapa de teste, a mesma unidade está sempre e que os pesos são multiplicados por *p*. A saída no teste é a mesma que a esperada no treinamento. Isso garante que, para qualquer unidade escondida, o resultado esperado (sob a distribuição utilizada para unidades com *dropout* no treinamento) é o mesmo que a saída real no teste. Ao fazer este dimensionamento, 2^n redes podem ser combinadas numa única rede neural a ser utilizada no momento do teste.

O treinamento de uma rede com *dropout* e o uso deste método de média aproximada no momento do teste, conduzem a uma maior generalização em uma grande variedade de problemas de classificação, comparado com outros métodos de regularização (SRIVASTAVA et al., 2014).

Camada *Softmax*

Geralmente, para problemas de classificação utilizando técnicas de aprendizagem profunda, é colocada uma camada *Softmax* no final da arquitetura. No

caso das redes neurais totalmente convolutivas (apresentadas na próxima seção) ela calcula a probabilidade de cada pixel pertencer a uma determinada classe.

Dadas *n* possíveis classes, a camada *Softmax* terá *n* nós denotados por p_i, em que $i = 1, ..., n$; p_i especifica uma distribuição de probabilidade discreta, portanto, $\sum_{i=1}^{n} p_i = 1$.

Seja $\boldsymbol{h}$ a ativação dos nós da penúltima camada, $\boldsymbol{W}$ é o vetor de pesos que conectam a penúltima camada à camada Softmax. A entrada total à referida camada, denotada por *a*, é descrita através da equação 23 (TANG, 2014):

$$a_i = \sum_k \boldsymbol{h}_k \boldsymbol{W}_{ki} \tag{23}$$

Então temos que, a probabilidade de uma entrada *a* pertencer à classe *i*, é determinada pela equação 24:

$$p_i = \frac{e^{a_i}}{\sum_j^n e^{a_j}} \tag{24}$$

A classe estimada $\hat{\imath}$ será aquela de maior valor de probabilidade:

$$\begin{aligned} \hat{\imath} &= \arg\max_i p_i \\ &= \arg\max_i a_i \end{aligned} \tag{25}$$

Geralmente, é usada uma camada de tipo *Softmax* com perda, chamada "*SoftmaxWithLoss*", que calcula a perda logística multinomial da *Softmax* de suas

entradas. Ela é conceptualmente idêntica à camada *Softmax* seguida por uma camada de perda logística multinomial, mas fornece gradientes mais estáveis numericamente.

3.2.3 Redes neurais totalmente convolutivas

Os sucessos recentes de arquiteturas profundas tais como: AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), VGGNet (SIMONYAN; ZISSERMAN, 2015), GoogLeNet (IOFFE; SZEGEDY, 2015), e ResNet (HE et al., 2015) tem convertido as CNNs no padrão *de facto* para a classificação de imagens. Além disso, para aproveitar o seu bom desempenho, elas tem sido adaptadas para outras tarefas de reconhecimento visual, por exemplo: a detecção de objetos através da delimitação com um retângulo (GIRSHICK et al., 2013), predição de pontos e zonas de interesse e segmentação semântica (LONG; SHELHAMER; DARRELL, 2015). Esta última refere-se à rotulagem de pixels, a qual é uma tarefa de maior complexidade em comparação com o reconhecimento de imagem e a detecção de objetos, devido a que a escala de percepção muda de uma imagem inteira a elementos de tipo pixel.

Uma arquitetura padrão de CNN tipicamente consiste de camadas de convolução, funções de ativação, camadas de *pooling* e camadas totalmente conectadas como blocos básicos. Long, Shelhamer e Darrell (2015) adaptaram e estenderam estas arquiteturas profundas de classificação para que pudessem executar a segmentação semântica de imagens. Eles eliminaram as camadas totalmente conectadas e introduziram camadas de convolução fracionárias, denominadas como deconvolução, para aprender a predizer o rotulo de cada pixel, a partir de uma imagem inteira na

entrada e uma imagem *ground truth* na saída. Este novo modelo foi denominado Rede Neural totalmente Convolutiva (FCN, do inglês *Fully Convolutional Neural Network*).

Na Figura 28 é mostrada uma arquitetura de FCN onde o objetivo é realizar uma segmentação semântica, isto é, classificar cada pixel da imagem de entrada de acordo à classe que pertence. No caso apresentado as classes são: gato, cachorro, sofá, janela e fundo. A primeira parte da rede consta de várias camadas de convolução que produzem mapas de características de diferentes profundidades: 96, 256, 384, 384, 256, 4096, 4096 e por fim 21. A seguir encontra-se a camada de convolução fracionária. Durante o processo de treinamento, os pesos dos seus filtros serão ajustados de forma que possam sobreamostrar (*upsampling*) a matriz de entrada até as dimensões da imagem original fazendo uma predição pixel a pixel, isto é, atribuindo cada pixel a uma classe. No caso da Figura 28, o total de classes do banco de dados utilizado é 21.

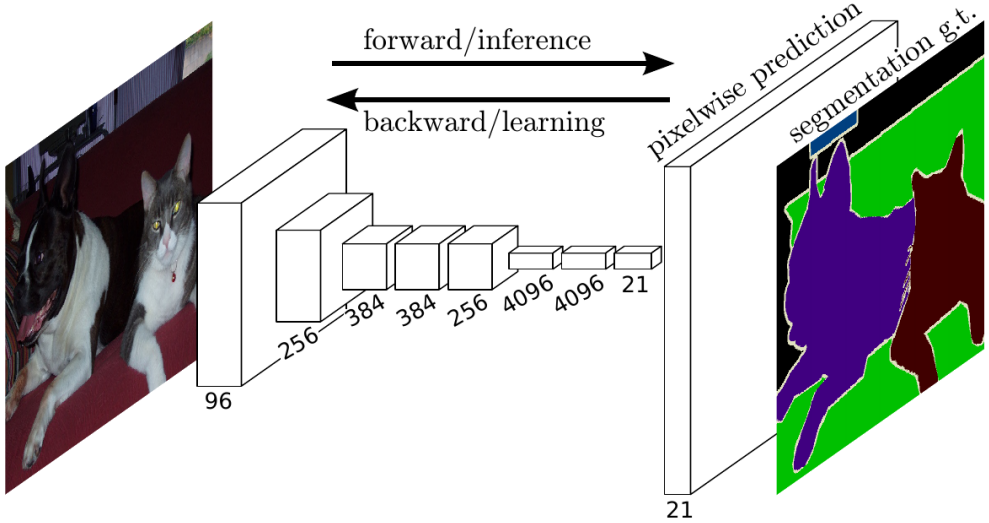


Figura 28. Modelo de rede totalmente convolutiva em que várias camadas de convolução extraem características (96, 256, 384, 384, 256, 4096, 4096, 21) da imagem. No final, uma camada de Deconvolução realiza uma predição pixel a pixel visando segmentar o cachorro, o gato, o sofá, a janela e o fundo. Os pesos da rede são ajustados iterativamente através da inferência e o aprendizado no treinamento.
Fonte: (LONG; SHELHAMER; DARRELL, 2015)

3.2.4 Métodos de otimização

Os algoritmos de aprendizagem profunda envolvem a otimização em muitos contextos, por exemplo, a realização da inferência e retropropagação nas CNN implica a resolução de um problema de otimização. O treinamento da rede é uma questão importante e de alto custo computacional, pelo qual um conjunto especializado de técnicas de otimização foi desenvolvido para resolvê-lo. Esta subseção apresenta as técnicas de otimização para o treinamento da rede neural que serão utilizadas neste trabalho.

3.2.4.1 Gradiente descendente estocástico

No treinamento de uma CNN, o conjunto de pesos $\boldsymbol{W} = (\boldsymbol{W}_1, ..., \boldsymbol{W}_L)$ devem ser aprendidos de forma que a função geral $\boldsymbol{Z} = f(\boldsymbol{X}; \boldsymbol{W})$ atinja o objetivo desejado. Consideraremos que temos N exemplos de relações entrada – saída desejadas $(\boldsymbol{X}_1, \boldsymbol{Z}_1)$, …, $(\boldsymbol{X}_N, \boldsymbol{Z}_N)$, em que $\boldsymbol{X}_i$ são as entradas de dados na rede e $\boldsymbol{Z}_i$ os valores de saídas correspondentes, e um termo de perda $\ell(\mathbf{Z}, \hat{\boldsymbol{Z}})$ que expressa a penalidade por predizer $\hat{\boldsymbol{Z}}$ em vez de $\boldsymbol{Z}$. A perda empírica da CNN, $L(\boldsymbol{W})$, é a perda média sobre todos os dados do conjunto de treinamento e pode ser expressada como:

$$L(\boldsymbol{W}) = \frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{Z}_i, f(\boldsymbol{X}_i; \boldsymbol{W})) \tag{26}$$

A forma mais fácil para minimizar a função de perda L e, de fato, a mais utilizada na prática é o algoritmo do gradiente descendente (CAUCHY, 1847). A ideia é simples: calcular o gradiente da função objetivo L em uma solução atual $\boldsymbol{W}_t$ e depois atualizar de forma iterativa as soluções ao longo do sentido da descida mais rápida de L. As seguintes equações expressam formalmente o cálculo do valor atualizado $\boldsymbol{V}_{t+1}$ e os pesos atualizados $\boldsymbol{W}_{t+1}$ na iteração $t+1$, dada a atualização de peso anterior $\boldsymbol{V}_t$ e os pesos atuais $\boldsymbol{W}_t$ (BERKELEY VISION AND LEARNING CENTER, 2014):

$$\boldsymbol{V}_{t+1} = \boldsymbol{V}_t - \alpha \nabla L(\boldsymbol{W}_t)$$
$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t + \boldsymbol{V}_{t+1} \quad (27)$$

em que: $\alpha \in \mathrm{R}_+$ é a taxa de aprendizado.

Em cada iteração do treino, a equação 28 é calculada sobre todo o conjunto de treinamento. Isto é referido na literatura como aprendizado em lote (*batch learning*) porque são considerados todos os dados antes de atualizar os pesos. Outra alternativa é o aprendizado estocástico (*stochastic learning*), onde em cada iteração os pesos são atualizados levando em conta apenas um exemplo de treinamento ou um número relativamente baixo em comparação com o conjunto de dados, conhecido como *mini-batch*. O aprendizado estocástico é a forma preferida porque geralmente é mais rápida do que o aprendizado em lote, oferece melhores soluções e pode ser usada para controlar as mudanças (LECUN et al., 1998).

Na equação 28 a taxa de aprendizado α é um parâmetro que determina o quão rápido ou lento a solução atual se move para o valor ótimo. Se α for muito grande se pode pular a solução ótima. Se α for muito pequeno se precisará de muitas iterações

para convergir aos melhores valores. A escolha de um bom valor de taxa de aprendizado é crucial.

Uma boa estratégia para a aprendizagem profunda com o algoritmo do gradiente descendente estocástico (SGD, do inglês *Stochastic Gradient Descend*) é inicializar a taxa de aprendizado α com um valor em torno de $\alpha \approx 0.01 = 10^{-2}$, diminuir α durante o treinamento usando um fator constante (por exemplo 10) quando a perda começa a chegar a um "*plateau*" aparente e repetir esse processo várias vezes (BERKELEY VISION AND LEARNING CENTER, 2014). Esta foi a estratégia utilizada por Krizhevsky, Sutskever e Hinton (2012) no treinamento da sua CNN na competição ILSVRC-2012, onde foram os vencedores.

Enquanto o SGD permanece como uma estratégia de otimização muito popular, o aprendizado com ele as vezes pode ser lento. O método do momento (POLYAK, 1964) é uma técnica para acelerar o gradiente descendente, especialmente em funções com curvaturas elevadas em que o SGD tende a oscilar. Ele acumula um vetor de velocidade nas direções de redução persistente da função objetivo através das iterações. Acrescentando o momento $\mu \in [0, 1)$ na equação 27 temos:

$$\boldsymbol{V}_{t+1} = \mu \boldsymbol{V}_t - \alpha \nabla L(\boldsymbol{W}_t)$$
$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t + \boldsymbol{V}_{t+1} \tag{28}$$

Em uma rede neural, o padrão de entrada $\boldsymbol{X}$ fornece uma informação inicial que se propaga pelas unidades escondidas em cada camada, e por fim produz uma saída $\hat{\boldsymbol{Y}}$. Isso é chamado propagação para frente (*forward propagation*). Durante o treinamento, a propagação para frente continua até obter um valor de custo $J(\theta)$. Depois se executa

uma retropropagação na rede utilizando o algoritmo *Backpropagation*, analisado a seguir.

Algoritmo *Backpropagation* nas CNN

O algoritmo *backpropagation* (RUMELHART; HINTON; WILLIAMS, 1986) permite à informação do custo fluir para atrás através da rede, a fim de calcular o gradiente. O termo *backpropagation* é muitas vezes mal interpretado como que ele é o único algoritmo responsável pela aprendizagem nas redes neurais multicamadas. Na verdade, ele somente se refere ao método para calcular o gradiente, enquanto um outro algoritmo, tal como o SGD, é usado para realizar a aprendizagem utilizando este gradiente.

Chamaremos $\delta^{(l+1)}$ ao termo de erro para a (l+1)-ésima camada com uma função de custo $J(\boldsymbol{W}, \boldsymbol{b}; x, y)$ em que ($\boldsymbol{W}$, $\boldsymbol{b}$) são parâmetros da rede e (x, y) são os pares de dados de treinamento e *labels*. Se a l-ésima camada está densamente conectada à (l+1)-ésima camada, então o erro para a l-ésima camada é calculado como segue (Ng et al., 2015):

$$\delta^{(l)} = \left(\left(\boldsymbol{W}^{(l)}\right)^T \delta^{(l+1)}\right) \bullet f'(z^{(l)}) \tag{29}$$

E os gradientes são:

$$\nabla_{W^{(l)}} J(\boldsymbol{W}, \boldsymbol{b}; x, y) = \delta^{(l+1)}\left(a^{(l)}\right)^T,$$

$$\nabla_{b^{(l)}} J(\boldsymbol{W}, \boldsymbol{b}; x, y) = \delta^{(l+1)}. \tag{30}$$

Se a camada *l*-ésima é convolucional ou de *pooling* então o erro é propagado através de

$$\delta_k^{(l)} = upsample((\boldsymbol{W}_k^{(l)})^T \delta_k^{(l+1)} \bullet f'(z_k^{(l)})) \tag{31}$$

em que *k* indica o número do filtro e $f'(z_k^{(l)})$ é a derivada da função de ativação. A operação *upsample* tem que propagar o erro através da camada de *pooling* calculando este com respeito a cada unidade entrante à dita camada. Por exemplo, se temos um "*mean pooling*" então a *upsample* simplesmente distribui uniformemente o erro para uma única unidade de *pooling* entre as unidades que entram na camada anterior. No "*max pooling*", a unidade que foi escolhida como a de valor máximo, recebe todo o erro devido a que mudanças muito pequenas não poderiam perturbar o resultado apenas através dessa unidade.

Finalmente, o cálculo do gradiente no que diz respeito aos mapas de características dos filtros se baseia nas bordas resultantes da operação de convolução, e a matriz de erro $\delta_k^{(l)}$ é rotacionada da mesma forma que acontece com os filtros na camada convolutiva.

$$\nabla_{W_k^{(l)}} J(\boldsymbol{W}, \boldsymbol{b}; x, y) = \sum_{i=1}^{m} (a_i^{(l)} * rot90\left(\delta_k^{(l+1)}, 2\right),$$
$$\nabla_{b_k^{(l)}} J(\boldsymbol{W}, \boldsymbol{b}; x, y) = \sum_{a,b} (\delta_k^{(l+1)})_{a,b} \,. \tag{32}$$

em que $a^{(l)}$ é a entrada à *l*-ésima camada e $a^{(1)}$ é a imagem de entrada. A operação $(a_i^{(l)}) * \delta_k^{(l+1)}$ é a convolução "válida" entre a *i*-ésima entrada na *l*-ésima camada e o erro respeito ao *k*-ésimo filtro.

3.2.4.2 Gradiente acelerado de Nesterov

O gradiente acelerado de Nesterov (NAG, do inglês *Nesterov Accelerated Gradient*) foi proposto por Nesterov (1983) para resolver problemas de otimização convexa. Ele pode ser muito efetivo para otimizar certos tipos de arquiteturas de aprendizagem profunda, por exemplo, os *autoencoders* (SUTSKEVER et al., 2013).

A fórmula de atualização dos pesos é similar à do SGD, conforme é mostrado na equação 33. A diferença é que o gradiente da função de perda (∇L) é calculado nos pesos mais o momento $\nabla L(\boldsymbol{W}_t + \mu \boldsymbol{V}_t)$.

$$\boldsymbol{V}_{t+1} = \mu \boldsymbol{V}_t - \alpha \nabla L(\boldsymbol{W}_t + \mu \boldsymbol{V}_t)$$
$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t + \boldsymbol{V}_{t+1} \quad (33)$$

em que, α é a taxa de aprendizado e μ é o momento.

3.2.4.3 AdaGrad

Este algoritmo de otimização, proposto por Duchi, Hazan e Singer (2011), adapta individualmente as taxas de aprendizado de todos os parâmetros do modelo escalando-os de forma inversamente proporcional à raiz quadrada da soma de todos os seus valores quadrados históricos. Os parâmetros com maior derivada parcial da perda têm uma rápida diminuição em suas taxas de aprendizado, enquanto aqueles com derivadas parciais pequenas têm uma diminuição relativamente pequena das suas taxas

de aprendizado. O efeito na rede é um maior progresso nas direções mais levemente inclinadas do espaço de parâmetros (GOODFELLOW; BENGIO; COURVILLE, 2016).

Dada a informação de atualização de todas as iterações anteriores $(\nabla L(W))_{t'}$ para $t' \in \{1, 2, ..., t\}$, a fórmula de atualização, especificada para cada componente i dos pesos $\boldsymbol{W}$:

$$(\boldsymbol{W}_{t+1})_i = (\boldsymbol{W}_t)_i - \alpha \frac{(\nabla L(\boldsymbol{W}_t))_i}{\sqrt{\sum_{t'=1}^{t} (\nabla L(\boldsymbol{W}_{t'}))_i^2}} \quad (34)$$

em que α é a taxa de aprendizado, $\boldsymbol{W}_{t+1}$ é o vetor de pesos atualizados na iteração $t+1$, $\boldsymbol{W}_t$ é o vetor de pesos atuais, $(\nabla L(\boldsymbol{W}_t))$ é o gradiente da função de erro avaliada em $\boldsymbol{W}_t$.

No contexto da otimização convexa, o algoritmo AdaGrad possui algumas propriedades teóricas desejáveis, no entanto, tem sido demostrado de forma empírica que, para o treinamento de modelos de redes profundas, a acumulação de gradientes ao quadrado desde o início do treinamento pode resultar numa diminuição prematura e excessiva da taxa de aprendizado. O AdaGrad funciona bem para alguns, mas não para todos os modelos de aprendizagem profunda (GOODFELLOW; BENGIO; COURVILLE, 2016).

3.2.4.4 RMSprop

O algoritmo RMSProp, proposto por Tieleman e Hinton (2012), modifica ao AdaGrad para atingir um melhor desempenho no ambiente não-convexo, mudando a acumulação do gradiente dentro de uma média móvel ponderada exponencialmente. Ele

mostrou ser um algoritmo de otimização eficaz e prático para redes neurais profundas (GOODFELLOW; BENGIO; COURVILLE, 2016).

As equações para a atualização dos pesos são:

$$MS((\boldsymbol{W}_t)_i) = \delta\, MS((\boldsymbol{W}_{t-1})_i) + (1-\delta)\big(\nabla L(\boldsymbol{W}_t)\big)_i^{\,2} \tag{35}$$

$$(\boldsymbol{W}_{t+1})_i = (\boldsymbol{W}_t)_i - \alpha \frac{\big(\nabla L(\boldsymbol{W}_t)\big)_i}{\sqrt{MS((\boldsymbol{W}_t)_i))}} \tag{36}$$

em que, $MS(\cdot)$ (*Mean Square*) é o valor quadrático médio, α é a taxa de aprendizado, $\boldsymbol{W}_t$ é o vetor de pesos atuais, $\boldsymbol{W}_{t+1}$ é o vetor de pesos atualizados na iteração t+1, $(\nabla L(\boldsymbol{W}_t))$ é o gradiente da função de perda avaliada em $\boldsymbol{W}_t$, e δ controla a escala de comprimento da média móvel.

3.2.4.5 Adam

O algoritmo Adam, proposto por Kingma e Ba (2015), é um método de otimização de funções objetivos estocásticas a partir de gradientes de primeira ordem, baseado na estimação adaptativa de momentos de ordem inferior. O seu nome se deriva de "*adaptive moment estimation*", isto é, estimação de momento adaptativo. O método combina as vantagens do AdaGrad e o RMSprop, analisados anteriormente. Ele calcula as taxas de aprendizado adaptativas individuais para diferentes parâmetros a partir da estimativa dos momentos de primeira e segunda ordem dos gradientes.

A seguir, são apresentadas as equações de atualização:

$$(m_t)_i = \beta_1 (m_{t-1})_i + (1 - \beta_1)\big(\nabla L(\boldsymbol{W}_t)\big)_i \tag{37}$$

$$(v_t)_i = \beta_2 (v_{t-1})_i + (1 - \beta_2)\big(\nabla L(\boldsymbol{W}_t)\big)_i^2 \tag{38}$$

e

$$(\boldsymbol{W}_{t+1})_i = (\boldsymbol{W}_t)_i - \alpha \frac{\sqrt{1 - (\beta_2)_i^t}}{1 - (\beta_1)_i^t} \frac{(m_t)_i}{\sqrt{(v_t)_i} + \varepsilon} \tag{39}$$

em que, α é a taxa de aprendizado, $\beta_1{}^t$ e $\beta_2{}^t$ denotam β_1 e β_2 à potência t, $\nabla L(\boldsymbol{W}_t)$ é o gradiente da função de erro avaliada no vetor de pesos $\boldsymbol{W}_t$. O algoritmo atualiza as médias moveis exponenciais do gradiente (m_t) e o gradiente quadrado (v_t), donde os hiperparâmetros β_1, $\beta_2 \in [0, 1)$ controlam as taxas de decaimento exponencial de essas médias móveis. As próprias médias móveis são estimativas do primeiro momento (a média) e do segundo momento raso (a variância não centrada) do gradiente. No entanto, essas médias móveis são inicializados como vetores de zeros, levando a estimativas do momento que estão polarizadas para zero, especialmente durante os passos de tempo iniciais, e especialmente quando as taxas de decaimento são pequenas, isto é, as βs estão próximas a 1). Segundo os criadores do método, α = 0.001, β_1 = 0,9, β_2 = 0,999 e $\varepsilon = 10^{-8}$, são boas configurações padrões para os problemas de aprendizado de máquina.

3.2.4.6 AdaDelta

Este método, proposto por Zeiler (2012), se deriva do AdaGrad melhorando algumas das suas desvantagens. As equações de atualização são:

$$(v_t)_i = \frac{RMS\left(v_{t-1_i}\right)}{RMS\left(\nabla L(\boldsymbol{W}_t)\right)_i}(\nabla L(\boldsymbol{W}_{t\prime}))_i \tag{40}$$

$$(\boldsymbol{W}_{t+1})_i = (\boldsymbol{W}_t)_i - \alpha(v_t)_i \tag{41}$$

em que, o RMS (*Root Mean Square*) da função de perda é: $RMS(\nabla L(\boldsymbol{W}_t))_i = \sqrt{E[g^2] + \varepsilon}$ e o valor esperado do gradiente ao quadrado é: $E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho)g_t^2$, ρ é a taxa de decaimento, ε é um valor constante e α é a taxa de aprendizado.

4 MATERIAIS E MÉTODOS

Nesta seção serão descritos os materiais utilizados para este trabalho, o que compreende as características da base de dados utilizada, bem como o ambiente de desenvolvimento em termos de *hardware* e *software*.

Também serão explicados os passos metodológicos do presente trabalho, que visa segmentar os contornos do endocárdio e do epicárdio em imagens de ressonância magnética e aplicar métricas para avaliar o desempenho. Serão descritos os experimentos feitos com os diferentes algoritmos de otimização.

4.1 Materiais

4.1.1 Base de dados Sunnybrook

A base de dados Sunnybrook (RADAU et al., 2009) é formada por 45 conjuntos de dados de imagens de ressonância magnética cardíaca em diversos planos, obtidas com um equipamento 1,5 Tesla GE Signa. Esses 45 estudos estão divididos em três pastas: treinamento, validação e prova *online*, com 15 conjuntos cada uma.

Todas as imagens foram adquiridas durante 10 – 15 segundos de pausa respiratória, com uma resolução temporal de 20 fases cardíacas ao longo do ciclo cardíaco e digitalizadas a partir do final da diástole. De 6 a 12 imagens de 256 x 256 pixels foram obtidas do anel atrioventricular até o ápice (espessura = 8 mm, *gap* = 8 mm, FOV = 320 mm × 320 mm). Os dados estão anonimizados em formato DICOM (*Digital Imaging and Communication in Medicine*) e não possuem nenhum tipo de pré-processamento.

O conjunto de dados adquiridos dos pacientes foi classificado em quatros grupos que representam diversas morfologias, baseado nos seguintes critérios clínicos: (1) insuficiência cardíaca com infarto (HF-I), grupo que tinha EF < 40% e evidência de realce tardio de Gadolínio (Gd); (2) insuficiência cardíaca sem infarto (HF-NI), grupo que tinha EF < 40% sem realce tardio de Gd; (3) hipertrofia do ventrículo esquerdo (HYP), grupo que tinha uma EF normal (> 55%) e uma razão de massa ventricular sobre área de superfície corporal > 83 g/m^2; (4) saudável (N), grupo que tinha EF > 55% sem hipertrofia.

Os contornos do endocárdio e do epicárdio foram traçados por um cardiologista com experiência nas fatias correspondentes ao final da diástole e da sístole, incluindo os músculos papilares e as trabeculações do endocárdio na cavidade ventricular. Todos os contornos foram confirmados por outro cardiologista. As segmentações manuais são usadas como *ground truth* para propósitos de avaliação.

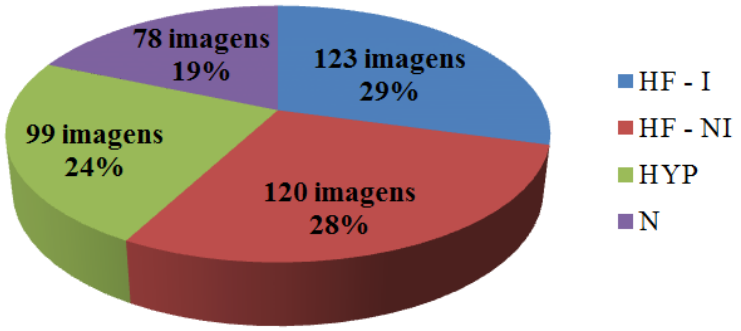


Figura 29. Distribuição das 420 imagens utilizadas da base de dados Sunnybrook de acordo às diferentes patologias: insuficiência cardíaca com infarto (HF-I), insuficiência cardíaca sem infarto (HF-NI), hipertrofia (HYP) e saudáveis (N).

Um total de 420 imagens possuem contornos tanto para o endocárdio quanto para o epicárdio. Delas 123 são de pacientes com insuficiência cardíaca que tiveram infarto, 120 são de pacientes com insuficiência cardíaca sem infarto, 99 são de pacientes

com hipertrofia do ventrículo esquerdo e 78 são de indivíduos saudáveis, o que corresponde a 29 %, 28 %, 24 % e 19 % do total, respectivamente, conforme é mostrado na Figura 29.

4.1.2 *Framework* para desenvolvimento da aprendizagem profunda

Caffe é um *framework* de código aberto que permite realizar experimentos e simulações com arquiteturas de aprendizagem profunda. Foi desenvolvido pelo Centro de Visão e Aprendizado de Berkeley (BVLC, do inglês *Berkeley Vision and Learning Center*) e atualmente é mantido por uma comunidade de contribuidores. Primeiramente foi projetado apenas para visão computacional, mas foi estendido para aplicações de reconhecimento da fala, robótica, neurociências, entre outras.

O seu código foi escrito em C ++ com o uso de CUDA (*Compute Unified Device Architecture*), a qual é uma arquitetura de cálculo paralelo de NVIDIA que aproveita a grande potência da GPU, oferecendo um incremento extraordinário do rendimento do sistema. Por exemplo, podem-se atingir velocidades de processamento de mais de 40 milhões de imagens por dia com uma simples GPU Titan ou K40 (≈ 2,5 ms por imagem). Embora tenha sido escrito em C ++, o Caffe tem extensões para as linguagens Python/Numpy e Matlab (JIA et al., 2014).

Caffe é aderente às melhores práticas de engenharia de *software*, fornecendo testes unitários para a correção, rigor experimental e velocidade na implantação. Os modelos podem ser executados tanto em modo CPU (*Central Processing Unit*) quanto na GPU sobre uma variedade de *hardware* (JIA et al., 2014). Suporta arquiteturas de rede na forma de grafos acíclicos dirigidos de forma arbitrária.

Algumas das vantagens deste *framework* são:

- Modularidade: O *software* foi projetado desde o início para ser tão modular quanto possível, permitindo fácil extensão para novos formatos de dados, camadas de rede e funções de perda.

- Separação entre representação e implementação: As definições de modelos em Caffe são escritas como arquivos de configuração usando a linguagem de *buffer* de protocolo “prototxt” (*Protocol Buffer*). Isso permite separar a representação do modelo da implementação.

- Cobertura de teste: Cada módulo em Caffe tem um teste, nenhum código novo é aceito no projeto sem os testes correspondentes. Isso permite melhorias rápidas e refatoração do código base.

- Modelos de referência: Caffe fornece modelos pré-treinados para tarefas visuais, por exemplo, a arquitetura de referência “AlexNet” (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) e outros modelos. Múltiplas camadas e funções de perda já estão implementadas.

4.1.3 Características do computador

O computador utilizado nos experimentos possui uma placa mãe de marca DELL, modelo Precision-Tower-5810, 128 Gb de memória RAM. O processador é Intel(R) Xeon(R) CPU E5-1650 v3@ a 3,50 GHz com 12 núcleos. A unidade de processamento gráfica utilizada é marca Nvidia, modelo Quadro K4200 com 4 Gb de RAM, 1440 núcleos CUDA e foi usada a versão 352.79 do *driver*. O computador tem instalado o sistema operacional Ubuntu 14.04 sob *kernel* Linux 4.2.0-36-generic.

4.2 Métodos

Na Figura 30 é mostrado o diagrama de blocos que descreve a metodologia adotada desde o início da pesquisa. Cada bloco será explicado nas próximas subsecções visando fornecer os detalhes da pesquisa.

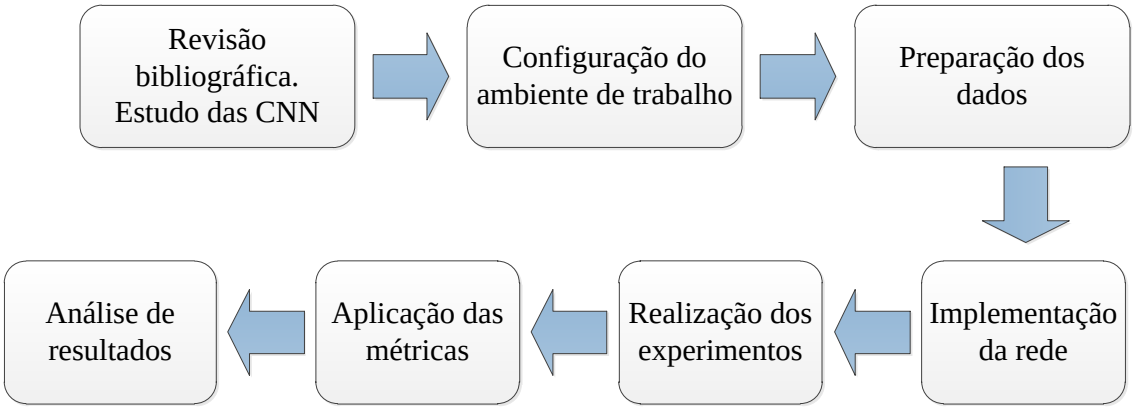


Figura 30. Metodologia seguida na pesquisa

4.2.1 Revisão bibliográfica. Estudo das CNN.

Conforme apresentado na Figura 30, no início da pesquisa foi realizada uma revisão bibliográfica onde foram estudadas várias alternativas de métodos de segmentação, entre os quais se podem mencionar os contornos ativos (*level set*, *sanekes*), os *snakes*, a técnica *watershed* (baseada em operações morfológicas), bem como a otimização de algoritmos de agrupamento utilizando algoritmos genéticos e enxame de partículas. Também foram analisadas várias publicações com propostas de metodologias para segmentação do miocárdio em imagens de ressonância magnética cardíaca, como foi elencado no Capítulo 2.

Devido aos desafios presentes nas imagens de ressonância magnética cardíaca, abordados na introdução do presente trabalho, a aplicação das técnicas anteriores não viabilizou a obtenção de um novo algoritmo generalizável para todo um conjunto de imagens, ou seja, os resultados eram bons apenas para um número reduzido delas. Isto sugeriu a pesquisa de uma técnica robusta, que fosse capaz de aprender as características das imagens e segmentar com "inteligência". Assim, foram abordadas as técnicas de aprendizado profundo, especificamente as redes neurais convolutivas.

No presente trabalho se desenvolveu e treinou um modelo de rede neural totalmente convolutiva para segmentar o miocárdio em imagens de ressonância magnética cardíaca.

4.2.2 Configuração do ambiente de trabalho

Na configuração do ambiente de trabalho foi preciso instalar um conjunto de bibliotecas e pacotes de *software* para garantir que o *framework* de desenvolvimento da aprendizagem profunda se beneficie das vantagens e potencialidades que oferece a placa de vídeo do computador.

O Caffe possui várias versões, dentre as quais foi compilada a versão estável atual. Permite duas modalidades de trabalho: CPU e GPU, esta última brinda maior velocidade e poder computacional. Para utilizá-la foi necessário instalar o *driver* da placa de vídeo Nvidia para o sistema operacional Ubuntu 14.04. Esse *driver* permite utilizar os núcleos CUDA da GPU para o processamento em paralelo com Caffe. Foi instalada a biblioteca cuDNN, que fornece implementação eficientes de rotinas tais como: processos *forward* e *backward* das camadas de convolução, *pooling*, normalização e funções de ativação; e a biblioteca PyCaffe, que permite trabalhar o

Caffe a partir do Python. O ambiente de desenvolvimento integrado (IDE, do inglês *Integrated Development Environment*) usado foi PyCharm.

4.2.3 Preparação dos dados

Em Caffe, os dados podem ser provenientes de base de dados eficientes como LevelDB ou LMDB (*Lightning Memory-mapped Database*), diretamente da memória ou de arquivos no disco rígido em formato HDF5 ou em formatos comuns tais como bmp, jpeg, etc.

LMDB é uma forma de armazenamento em base de dados extraordinariamente rápida e eficiente no que respeita à memória, devido a isso, os arquivos são mapeados nela. Foi desenvolvida pela corporação Symas para o projeto OpenLDAP. Possui o desempenho de leitura de uma base de dados em memória pura, enquanto mantém a persistência das bases de dados padrões baseadas em disco. Quer dizer, que sua execução é várias vezes mais rápida do que outros motores de base de dados. Atendendo a todas essas vantagens foi utilizado este formato.

Na Figura 31 é mostrado um fluxograma dos passos seguidos na preparação dos dados de entrada à rede que foi programado em Python. A base de dados utilizada contém imagens em formato DICOM de diferentes planos cardíacos, dentre os quais foi selecionado o eixo curto. Além disso, as imagens estão organizadas em pastas de acordo a uma determinada patologia. Dentro das pastas as imagens estão nomeadas e organizadas em uma ordem sequencial. Se a rede fosse treinada com essas imagens sequenciais ocorre que ela vai aprender apenas essas morfologias e quando testada com outras, um pouco diferentes, os resultados não serão os melhores. Para contornar esse problema, todas as imagens foram desorganizadas. Desta forma, durante o treinamento a

rede vai aprender de uma variedade de morfologias procedentes de diferentes pacientes e patologias. Igualmente na etapa de teste, a rede vai segmentar imagens, nunca vistas, com várias formas e características que foram aprendidas de algumas similares. Isso viabiliza a generalização no aprendizado.

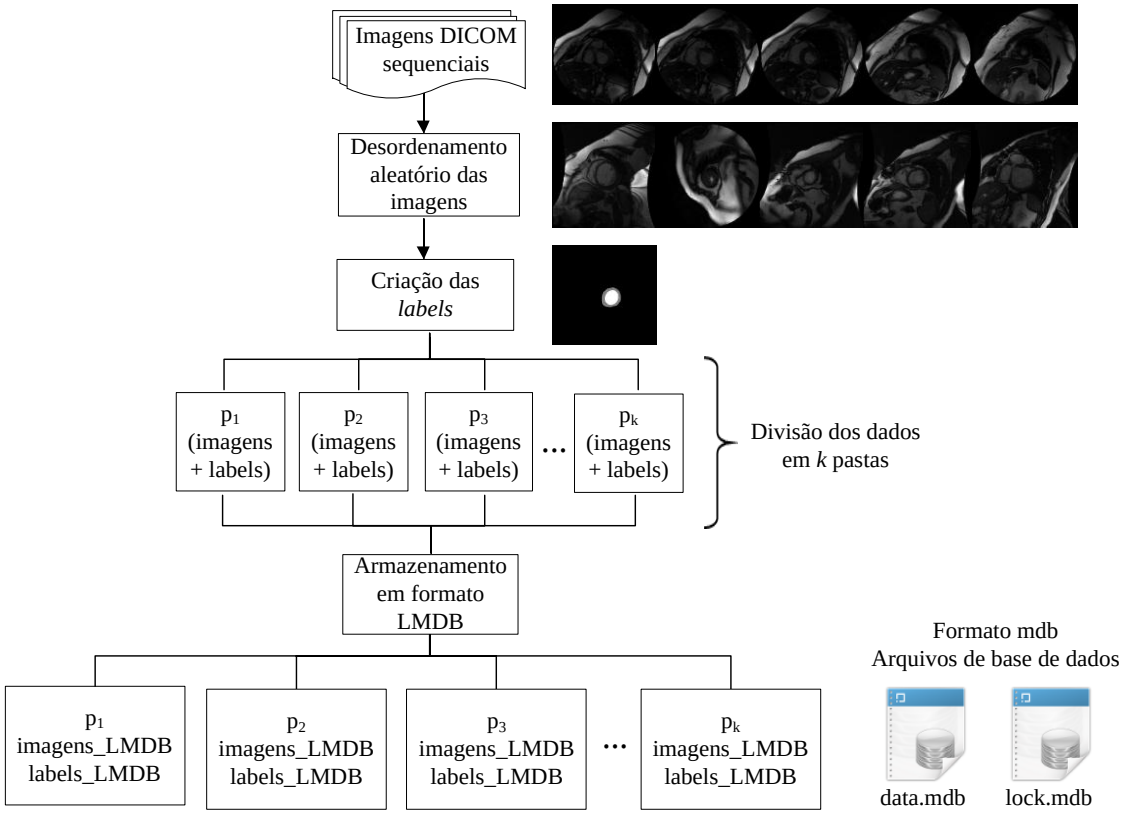
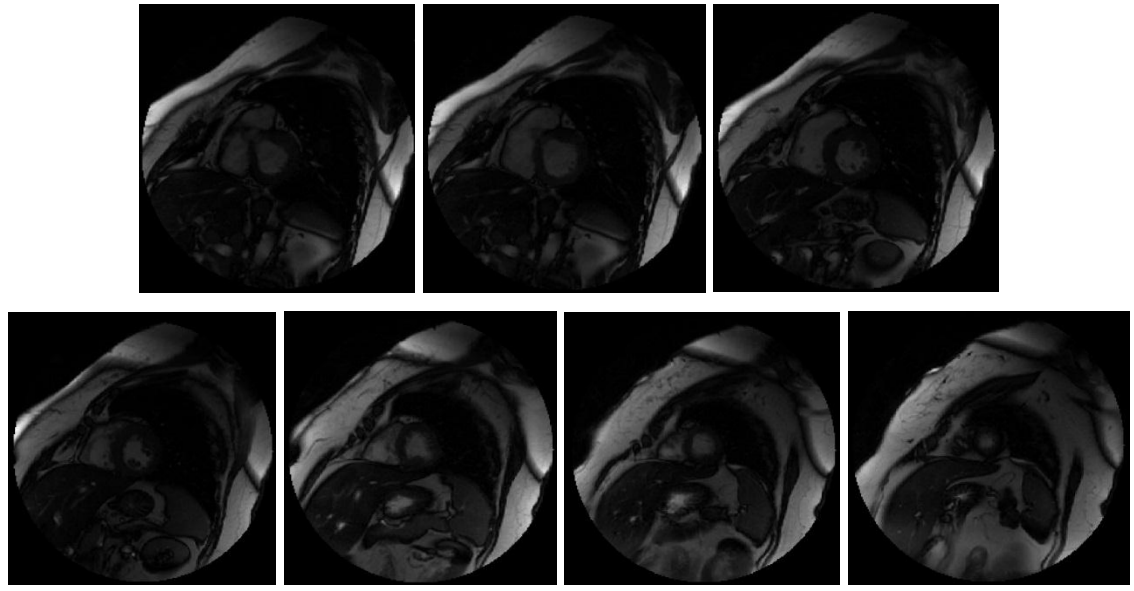


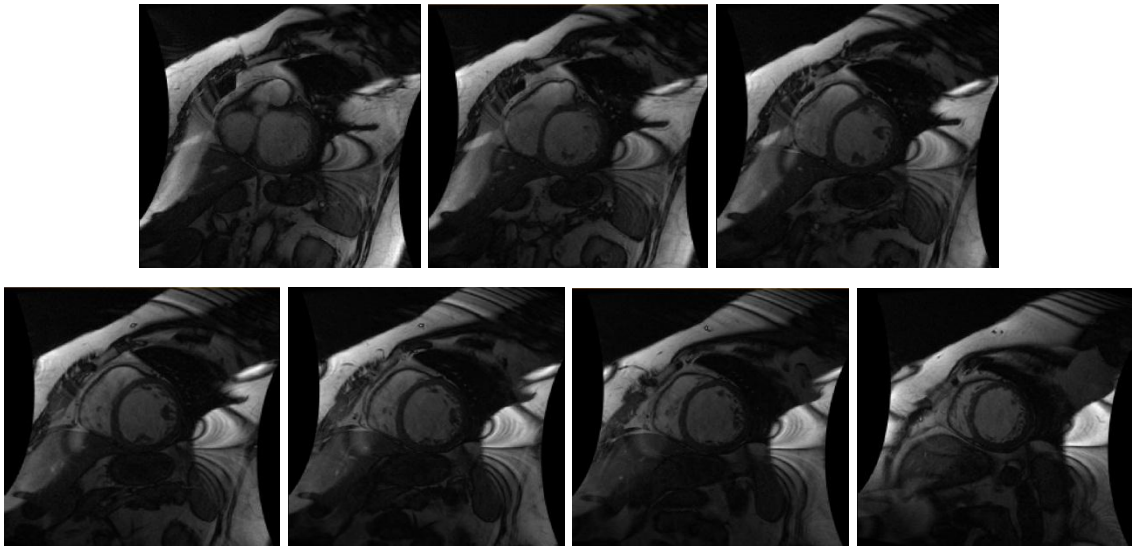
Figura 31. Fluxograma do processo de preparação dos dados.

Para compreender visualmente o que já foi explicado, na Figura 32 (a) são mostrados exemplos de imagens de treinamento da pasta SC-HYP-11, que pertencem a um estudo de pacientes com hipertrofia. Na Figura 32 (b) são mostrados exemplos de imagens para o teste da pasta SC-HF-NI-12, que pertencem a pacientes com insuficiência cardíaca não infartados. Quando a rede é treinada com uma sequencia de imagens muito similares como no caso mostrado na Figura 32 (a), na hora do teste ela

vai ter que segmentar imagens com características que nunca aprendeu, não apenas dos objetos de interesse, mas também do fundo.



(a)

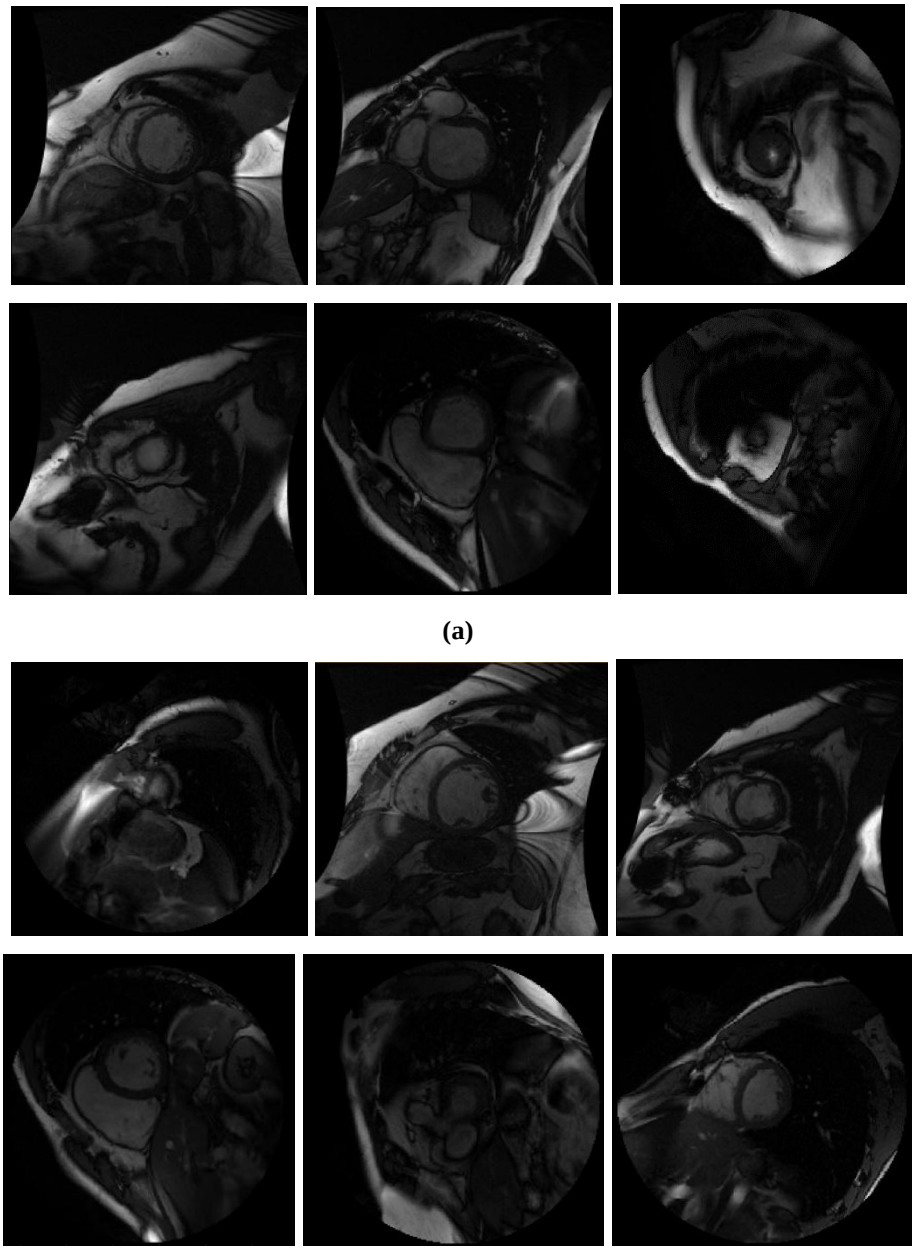


(b)

Figura 32. Imagens sequenciais para (a) treinamento (pasta SC-HYP-11) (b) teste (pasta SC-HF-NI-12).

Alternativamente, na Figura 33 (a) e (b) são mostrados exemplos de um conjunto de imagens de treinamento e teste após o ordenamento aleatório, respectivamente. Nesses conjuntos estão misturadas imagens de todas as pastas (patologias) o que vai

permitir à rede aprender características de uma variedade de imagens. Isso foi feito em Python utilizando a função “random.shuffle()” da biblioteca NumPy.



(a)

(b)

Figura 33. Imagens sequenciais para (a) treinamento e (b) teste de diversas pastas após o ordenamento aleatório.

As imagens com os rótulos de cada pixel (*labels*) para o treinamento supervisionado e para a validação dos resultados foram criadas a partir dos arquivos de

texto que contêm as coordenadas do endocárdio e do epicárdio da forma apresentada a seguir:

1. Criação de uma imagem de 256x256 pixels (mesmo tamanho que as usadas) onde os pixels internos à curva do endocárdio são preenchidos com “1” e o resto com “0”. Ela será chamada I_1.
2. Criação de uma imagem de 256x256 onde os pixels internos à curva do epicárdio são preenchidos com “1” e o resto com “0”. Ela será chamada I_2.
3. Criação de uma nova imagem I_3 resultante da resta: $I_2 - I_1$.
4. Em I_3 os pixels internos à curva do endocárdio são preenchidos com “2”. A efeitos da rede, ela vai classificar pixels como pertencendo a três classes: (0) fundo, (1) miocárdio e (2) ventrículo esquerdo.

A imagem resultante dos passos anteriores é mostrada na Figura 34.

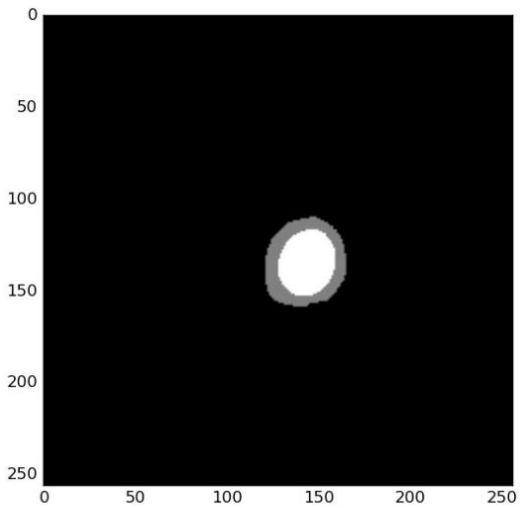


Figura 34. Imagem padrão com os rótulos dos pixels pertencendo a três classes diferentes: ventrículo esquerdo (em branco), miocárdio (em cinza) e fundo (em preto).

Após a criação das *labels* para a totalidade de imagens, estas foram divididas em 10 pastas para realizar uma validação cruzada (*k-fold cross validation*, $k = 10$). Seguidamente foram convertidas ao formato LMDB utilizando funções pertencentes à

biblioteca “lmdb” para Python. Quando as imagens são levadas ao referido formato, são compactadas originando dois arquivos: “data.mdb” e “lock.mdb” que só são entendíveis pela rede.

4.2.4 Implementação da arquitetura da rede

As redes profundas são modelos composicionais que são representados como uma coleção de camadas interconectadas que trabalham em fragmentos de dados. Uma etapa crucial é a definição e construção do modelo de rede a ser implementado. Inicialmente, foram realizados experimentos com várias arquiteturas, obtendo-se altos valores de perdas. Por exemplo, a arquitetura apresentada por Long, Shelhamer e Darrell (2015), a qual possui mais de 30 camadas, foi proposta originalmente para segmentar pixels pertencentes a 21 classes diferentes. Ela foi implementada e treinada em Caffe com nossos dados. Após três dias de treinamento, a imagem de saída era totalmente preta, ou seja, todos os pixels eram 0.

Uma outra arquitetura experimentada foi: (Conv100 - ReLU) – MaxPooling – (Conv200 - ReLU) – MaxPooling – (Conv300 - ReLU) (2x) – Dropout – Conv2 – Deconv – Crop – Softmax. Essa rede era muito mais simples que a anterior, porém ela começou a dar resultados incipientes. Esse modelo foi aprimorando-se gradualmente até chegar à nossa arquitetura atual, mostrada na Figura 35:
(Conv64 - ReLU) (2x) – MaxPooling – (Conv128 - ReLU) (2x) – MaxPooling – (Conv256 - ReLU) (2x) – MaxPooling – (Conv512 – ReLu - Dropout) (2x) – Conv3 - ReLU – Deconv – Crop – Softmax.

A mencionada arquitetura foi implementada através da programação em Python na plataforma de desenvolvimento Caffe. Nesta plataforma, a rede é definida camada

por camada, construindo o modelo de baixo para cima, isto é, começando com uma camada de dados que carrega do disco e terminando com uma de perda que calcula a função objetivo de uma determinada tarefa, como por exemplo, classificação ou reconstrução. O conjunto de camadas está conectado em um grafo acíclico dirigido, isso significa que para cada vértice *v*, não existe um caminho direto que comece e termine em *v*.

Na medida em que os dados e os gradientes fluem através da rede, o Caffe armazena, comunica e manipula as informações como *blobs*, os quais são basicamente objetos de abstração de memória.

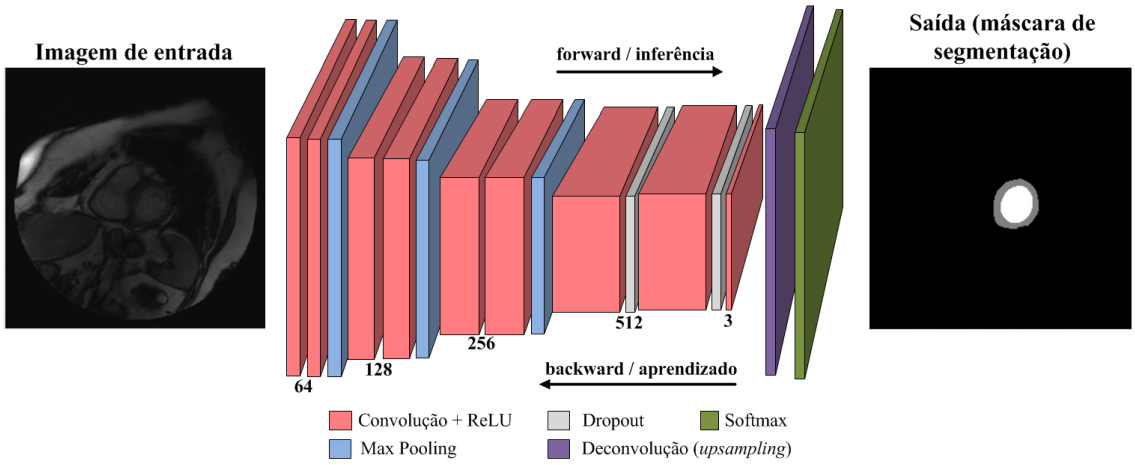


Figura 35. Arquitetura da FCN utilizada para segmentação do miocárdio. A imagem de entrada passa por camadas de convolução que vão extraindo características e por camadas de *pooling* que reduzem suas dimensões de largura e altura. No final uma camada de Deconvolução recupera o tamanho da imagem original e a camada Softmax atribui uma classificação para cada pixel. Os pesos dos neurônios que conformam as camadas são ajustados através dos processos de inferência e retropropagação durante o treinamento.

As camadas têm duas responsabilidades principais para atingir o funcionamento da rede como um todo: uma passagem para frente, conhecida como inferência ou *forward*, que recebe as entradas e produz as saídas; e uma passagem para trás, conhecida como retropropagação ou *backward*, que recebe o gradiente em relação à

saída e calcula os gradientes com relação aos parâmetros e às entradas, que são, por sua vez, retropropagadas para camadas anteriores.

Os dois trechos de código apresentados nos Quadros 1 e 2 ilustram como foi programado o modelo. Primeiro foram importadas as classes ***layers*** e ***params***, que contêm as implementações de várias camadas e seus parâmetros, respectivamente.

A função **NetSpec()** permite definir a rede programaticamente e ir anexando as camadas. Inicialmente é criada uma rede vazia à qual se irá acrescentando camadas. Depois foram definidas duas funções auxiliares para facilitar a chamada das camadas mais utilizadas, ou seja, as mais repetidas na arquitetura da rede, que são: a de convolução, as funções de ativação ReLU e a de *pooling*. Isso simplifica o código de programação da construção da rede e a configuração dos parâmetros de cada camada e algumas constantes no momento da criação da camada. As camadas para a convolução, as funções de ativação ReLU, *pooling*, *dropout*, deconvolução e *softmax* estão pré-definidas no Caffe e pertencem à classe ***Layers***.

A função **conv_relu** recebe as configurações dos parâmetros que precisa a camada de convolução para sua criação e retorna duas camadas: uma de convolução e uma ReLU. Os parâmetros utilizados na criação da camada de convolução são:

- *bottom*: Especifica a camada anterior, ou seja a entrada da camada atual.
- *kernel_size*: Recebe as dimensões do filtro.
- *stride*: Recebe o deslocamento desejado do filtro. Se não é especificado, toma o valor de 1.
- *num_output*: Número de filtros que vai ter essa camada.
- *pad*: Recebe o número de pixels para preencher com zeros a cada lado da entrada. Se não é especificado, toma o valor de 0.

- *group*: Especifica o número de grupos de filtros. Se $g > 1$, a conectividade de cada filtro é restrita a um subconjunto da entrada. Especificamente, os canais de entrada e de saída são separados em g grupos, e o i-ésimo canal de grupo de saída será ligado apenas ao i-ésimo canal de grupo de entrada. Se não é especificado, toma o valor de 1.
- *weight_filler*: Especifica o tipo de inicialização dos pesos.
- *bias_filler*: Especifica o tipo de inicialização das polarizações.

Existem outros dois parâmetros: *lr_mult* e *decay_mult* que são os multiplicadores da taxa de aprendizado e do decaimento de esta, respetivamente. Eles são configurados tanto para os pesos dos filtros quanto para as polarizações através da definição dos chamados “dicionários” em Python.

A criação da camada ReLU é simples, ela apenas precisa da informação da camada anterior, que sempre vai ser a de convolução criada anterior a ela.

A função **max_pool** recebe as configurações dos parâmetros que precisa a camada de *pooling* para sua criação e retorna dita camada com a operação fixa de achar o máximo valor. Os parâmetros utilizados na criação da camada de *pooling* são:

- *bottom*: Especifica a camada anterior, ou seja a entrada da camada atual.
- *pool*: Especifica que predicado será utilizado para subamostrar a imagem, pode ser escolhendo a média, o valor máximo, entre outros.
- *kernel_size*: Recebe as dimensões do filtro.
- *stride*: Recebe o deslocamento desejado do filtro. Se não é especificado, toma o valor de 1.

Quadro 1. Trecho de código com a criação de duas funções auxiliares para as camadas de convolução, ReLU e *pooling*.

```
import caffe, os
from caffe import layers as L
from caffe import params as P

n = caffe.NetSpec()

# funções auxiliares para estruturas comuns
def conv_relu(bottom, ks, nout, mult=1, stride=1, pad=0, group=1):
    conv = L.Convolution(bottom, kernel_size=ks, stride=stride,
                         num_output=nout,pad=pad, group=group,
                         weight_filler=dict(type='xavier'),
                         bias_filler=dict(type='constant', value=0),
                         param=[dict(lr_mult=mult, decay_mult=mult),
                         dict(lr_mult=2*mult, decay_mult=0*mult)])
    return conv, L.ReLU(conv, in_place=True)

def max_pool(bottom, ks, stride=1):
    return L.Pooling(bottom, pool=P.Pooling.MAX, kernel_size=ks,
                     stride=stride)
```

O conjunto de camadas e as ligações entre elas foram programados dentro de uma função chamada **FCN**, a qual recebe como parâmetros as imagens e suas respectivas *labels* em formato LMDB; o tamanho do *batch*, que neste caso é 1 devido a que o aprendizado é estocástico.

Quadro 2. Definição das camadas da rede neural totalmente convolutiva.

```
def FCN(images_lmdb, labels_lmdb, batch_size, include_acc=False):

    # definição da rede
    n.data = L.Data(source=images_lmdb, backend=P.Data.LMDB,
                    batch_size=batch_size, ntop=1,
                    transform_param=dict(crop_size=0, mean_value=[77],
                    mirror=False))

    n.label = L.Data(source=labels_lmdb, backend=P.Data.LMDB,
                     batch_size=batch_size, ntop=1)

    n.conv1, n.relu1 = conv_relu(n.data, ks=3, nout=64, stride=1,
                                 pad=1)
    n.conv2, n.relu2 = conv_relu(n.relu1, ks=3, nout=64, stride=1,
                                 pad=1)
    n.pool1 = max_pool(n.relu2, ks=2, stride=2)

    n.conv3, n.relu3 = conv_relu(n.pool1, ks=3, nout=128, stride=1,
                                 pad=1)
    n.conv4, n.relu4 = conv_relu(n.relu3, ks=3, nout=128, stride=1,
                                 pad=1)
    n.pool2 = max_pool(n.relu4, ks=2, stride=2)
```

```
    n.conv5, n.relu5 = conv_relu(n.pool2, ks=3, nout=256, stride=1,
                                 pad=1)
    n.conv6, n.relu6 = conv_relu(n.relu5, ks=3, nout=256, stride=1,
                                 pad=1)
    n.pool3 = max_pool(n.relu6, ks=2, stride=2)

    n.conv7, n.relu7 = conv_relu(n.pool3, ks=7, nout=512, stride=1,
                                 pad=3)
    n.drop1 = L.Dropout(n.relu7, dropout_ratio=0.5, in_place=True)

    n.conv8, n.relu8 = conv_relu(n.drop1, ks=1, nout=512, stride=1,
                                 pad=0)
    n.drop2 = L.Dropout(n.relu8, dropout_ratio=0.5, in_place=True)

    n.conv9, n.relu9 = conv_relu(n.drop2, ks=1, nout=3, stride=1,
                                 pad=0)

    n.upscore1 = L.Deconvolution(n.conv9,
                                 convolution_param=dict(num_output=3,
                                 kernel_size=16, pad=4, stride=8,
                                 weight_filler=dict(type='bilinear'),
                                 bias_filler=dict(type='constant',
                                 value=0.1), bias_term=True),
                                 param=[dict(lr_mult=1, decay_mult=1),
                                 dict(lr_mult=2, decay_mult=0)])
    n.score1 = L.Crop(n.upscore1, n.data)

    n.loss = L.SoftmaxWithLoss(n.score1, n.label,
                               loss_param=dict(normalize=True))
```

Conforme pode ser observado no código, inicialmente são declaradas duas camadas. A primeira corresponde aos dados em LMDB e a outra se refere às *labels* que só serão usadas na camada final Softmax para calcular a função de perda.

A rede contem nove camadas de convolução. As primeiras seis têm campos receptivos de 3 × 3 pixels, os hiperparâmetros de *padding* e *stride* são ambos iguais a 1. Cada uma das primeiras seis camadas de convolução produz 64, 64, 128, 128, 256 e 256 mapas de características, respectivamente. Assim, à saída delas temos um volume. As duas camadas convolutivas seguintes produzem 512 mapas de características. Elas possuem campos receptivos de 7 × 7 e 1 × 1, *padding* 3 e 0, respectivamente, e *stride* igual a 1.

A última camada de convolução produz três mapas de características devido a que a classificação dos pixels será em três classes: (0) fundo, (1) miocárdio e (2) ventrículo esquerdo. Todos os pesos das camadas de convolução foram inicializados segundo o esquema de Xavier que determina automaticamente a escala de inicialização baseado no número de neurônios de entrada e saída. As polarizações foram inicializadas como constantes, com o valor de 0. Cada camada de convolução é seguida de uma camada ReLU, que aplica uma função de ativação em cada um dos neurônios, como foi analisado na seção anterior.

Como foi abordado, as camadas de *Pooling* reduzem progressivamente o tamanho espacial da entrada a fim de diminuir a quantidade de parâmetros e cálculos na rede. Na arquitetura da Figura 35, todas elas possuem filtros de tamanho 2 × 2 aplicados com um *stride* de 2, desta forma cada fatia do volume de entrada é sub-amostrada por um fator de 2.

As camadas de convolução que produzem 512 mapas de características são seguidas por camadas de *Dropout* que, como foi analisado no capítulo 3, reduzem o *overfitting* impedindo as co-adaptações complexas dos dados de treinamento. O parâmetro de razão de *Dropout*, que controla a probabilidade de que uma determinada unidade seja omitida, foi estabelecido como sendo 0,5.

A camada de deconvolução realiza uma sobreamostragem a fim de obter um mapa preditivo do mesmo tamanho da entrada e faz a predição da classe à que pertence cada pixel. Ela funciona de forma inversa à camada de convolução. Em outras palavras, é uma camada de convolução com as passagens de *forward* e *backward* revertidas. Ela reutiliza os parâmetros da camada de convolução, mas eles tomam o sentido oposto, isto é, o *padding* é removido da saída em vez de adicionado na entrada, e o *stride* resulta em

uma sobre amostragem em vez de uma sub-amostragem. O tamanho do filtro, o *stride* e o *padding* foram fixados a 16, 8 e 4, respectivamente.

A camada *Crop* recorta sua entrada ao tamanho e nas dimensões especificadas. Ela não aprende parâmetros, apenas realiza uma operação fixa. O seu uso é opcional, ela é utilizada em caso que seja preciso adaptar o tamanho de saída da camada de deconvolução ao tamanho original da imagem de entrada.

A última camada *SoftmaxWithLoss* implementa a *softmax* e a perda logística multinomial, isto economiza tempo e melhora a estabilidade numérica. Ela recebe como entrada os dados da camada anterior que contem as predições de cada pixel e as *labels* introduzidas no início da rede. A partir dessas entradas, ela calcula o valor da função de perda, o reporta quando começa a retropropagação e inicializa o gradiente em relação a *score1*, que são as predições.

Com o decorrer do treinamento os pesos das conexões internas dos neurônios vão se ajustando de forma que o valor da função de perda vai minimizando até atingir uma condição de parada determinada.

4.2.5 Experimentos

Conforme é mostrado na Figura 36, as etapas de preparação dos dados e de construção da rede, explicadas em subseções anteriores, foram o prelúdio para a realização dos treinamentos. Posteriormente, com o modelo treinado e as imagens nunca vistas pela rede, foi realizada a etapa de teste.

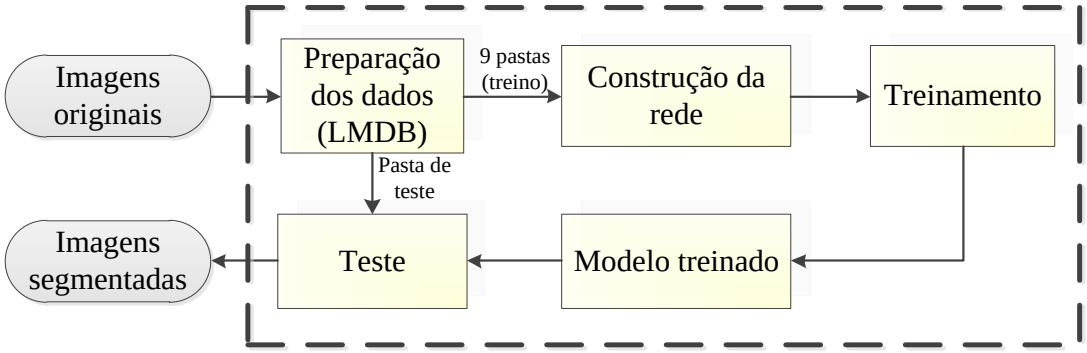


Figura 36. Esquema geral dos processos realizados para a segmentação: o conjunto de imagens é preparado e convertido ao formato de entrada à rede, após a construção dela é realizado o treinamento o qual resulta em um modelo com os pesos dos neurônios otimizados para resolver a tarefa da segmentação. Esse modelo treinado posteriormente é testado com imagens inéditas.

O treinamento envolve a resolução de um problema de otimização que visa minimizar a perda da rede. Para um conjunto de dados *D*, a função objetivo da otimização é a perda média em todas as instâncias de dados |*D*|, sua expressão é apresentada na equação 42:

$$L(W) = \frac{1}{D}\sum_{i}^{|D|} f_W\left(X^i\right) + \lambda r(W) \tag{42}$$

em que, $f_W\left(X^i\right)$ é a perda na instância de dado $X^{(i)}$ e $r(\boldsymbol{W})$ é um termo de regularização com peso λ.

Na prática, em cada iteração é usada uma aproximação estocástica da função objetivo anterior passando um *mini-batch* de *N* instâncias de dados em que $N << |D|$, conforme é mostrado na equação 43:

$$L(W) \approx \frac{1}{N}\sum_{i}^{N} f_W\left(X^i\right) + \lambda r(W) \tag{43}$$

Durante o treinamento, a rede calcula f_W na passagem para a frente e o gradiente ∇f_W na retropropagação. A atualização do vetor de pesos $\boldsymbol{W}$ é calculada a partir do gradiente da perda ∇f_W, o gradiente da regularização $\nabla r(\boldsymbol{W})$ e outros parâmetros particulares a cada método de otimização. Neste trabalho, o *mini-batch* tem um tamanho igual a 1, o que quer dizer que, em cada iteração, a rede ajusta os pesos para uma imagem de treinamento.

O termo de regularização $r(\boldsymbol{W})$ foi ponderado com peso $\lambda = 0$ devido a que já a rede contém duas camadas Dropout que são encarregadas de realizar esta mesma função de regularização. Se, além disso, adiciona-se esse termo, é provocada uma maior penalização nos pesos dos neurônios que, conforme foi testado, resulta em um pior desempenho da rede.

Na etapa de treinamento, foram realizados vários experimentos visando aprofundar no estudo das redes neurais convolutivas e na compreensão do comportamento da convergência dos diferentes métodos de otimização. A rede proposta foi treinada com os seis métodos analisados na seção 3.2.4, com cada um deles foram rodados dez treinamentos, correspondentes às dez pastas de imagens criadas conforme explicado na seção 4.2.3, totalizando 6 experimentos e 60 treinamentos.

As configurações dos parâmetros do treinamento e da otimização são programadas em um arquivo denominado "Solver". No referido arquivo pode ser especificado:

- O número de iterações em que se deseja salvar os pesos e o estado geral da rede em arquivos "*.caffemodel*" e "*.solverstate*", respectivamente.

- Visualização da perda e do valor da taxa de aprendizado ao decorrer das iterações.
- Tipo de método de otimização e seus parâmetros correspondentes em cada caso.
- Número máximo de iterações.
- Uso da GPU ou da CPU para rodar o treinamento.

No presente trabalho tudo foi feito sobre a GPU devido a que o aprendizado profundo envolve uma grande quantidade de operações vetoriais e matriciais. As GPUs foram projetadas para lidar com tais operações em paralelo, ao contrário de um único núcleo CPU que realiza as operações em série, ou seja, processando um elemento de cada vez. Do anterior podemos compreender intuitivamente que, usando GPU, o aprendizado profundo é executado várias vezes mais rápido que usando CPU.

Cada um dos métodos de otimização utilizados conduz o aprendizado da rede segundo suas próprias características e de acordo às equações de atualização dos pesos. Na Tabela 3 é apresentado um resumo da configuração dos parâmetros em cada caso. A condição de parada dos treinamentos foi ao alcançar o número máximo de iterações, que em todos os casos foi 100.000. A escolha desse número foi baseada na quantidade de imagens de treinamento. Além disso, vai permitir analisar a convergência e o comportamento do aprendizado durante um tempo considerável.

Tabela 3 Resumo da configuração dos parâmetros dos métodos de otimização utilizados nos treinamentos

Método	Taxa de aprendizado inicial	Momento	Política de taxa de aprendizado	Número máximo de iterações
SGD	0.01	0.9	“multistep”, stepvalues: [30.000, 60.000], gamma: 0.1	100.000

Nesterov	0.01	0.9	“step”, stepsize: 20.000 gamma: 0.1	100.000
RMSProp	4·10-5	0	“inv”, gamma: 0.0002, power: 0.75, rms_decay: 0.9	100.000
Adam	0.001	m1: 0.90 m2: 0.99	“step”, stepsize: 20.000, gamma: 0.1	100.000
AdaDelta	0.01	0.90	“step”, stepsize: 20.000 gamma: 0.1, delta: 1e-6	100.000
AdaGrad	0.001	0	“poly”, power: 0.001	100.000

A plataforma de aprendizagem profunda usada permite a implementação de várias políticas para controlar a atualização da taxa de aprendizado ao longo do treinamento. Para o SGD foi utilizada a política de múltiplos passos, denominada “multistep”, a qual obteve sucesso no trabalho de classificação de imagens do Krizhevsky e colaboradores (2012). Sua fórmula, apresentada na equação 44, é a mesma que para o caso da política “step” usada para Nesterov, Adam e AdaDelta. O detalhe é que no caso da “multistep”, os valores dos passos são variáveis, não sendo assim para “step” que são passos iguais.

$$lr_{new} = lr_{base} \cdot \gamma^{floor\left(\frac{iter}{step}\right)} \tag{44}$$

em que, lr_{new} é a nova taxa de aprendizado calculada, lr_{base} é a taxa de aprendizado inicial, gamma (γ) é um fator constante elevado ao valor arredondado da divisão da iteração atual (iter) e o valor do passo (step).

Nas Figuras 37, 38, 39 e 40 são apresentados os gráficos das taxas de aprendizado segundo as políticas de atualização usadas no SGD, Nesterov, Adam e AdaDelta, respectivamente.

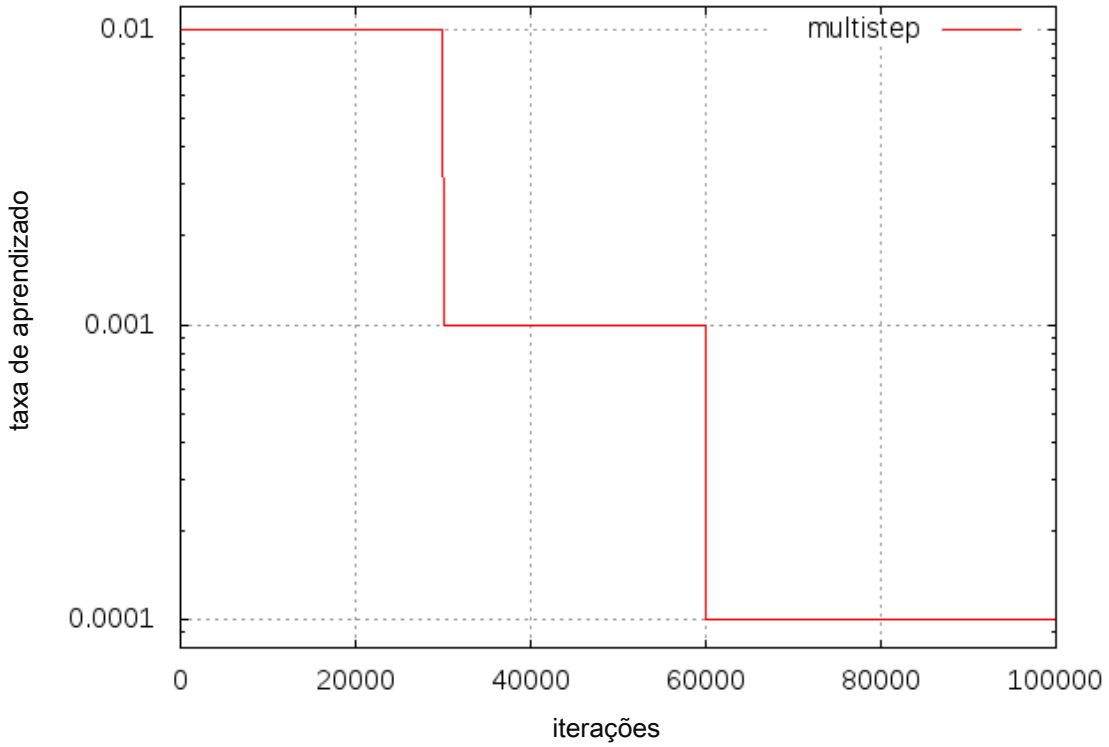


Figura 37. Taxa de aprendizado com a política "*multistep*" durante o treinamento com o SGD.

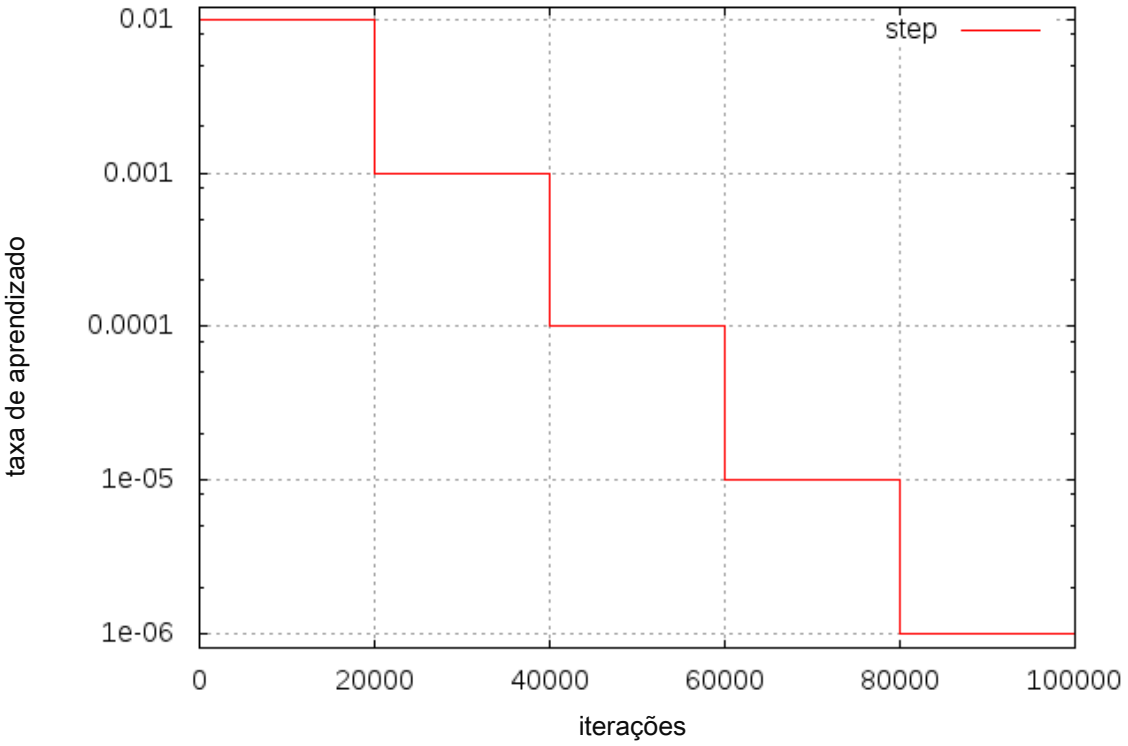


Figura 38. Taxa de aprendizado com a política "*step*" durante o treinamento com Nesterov.

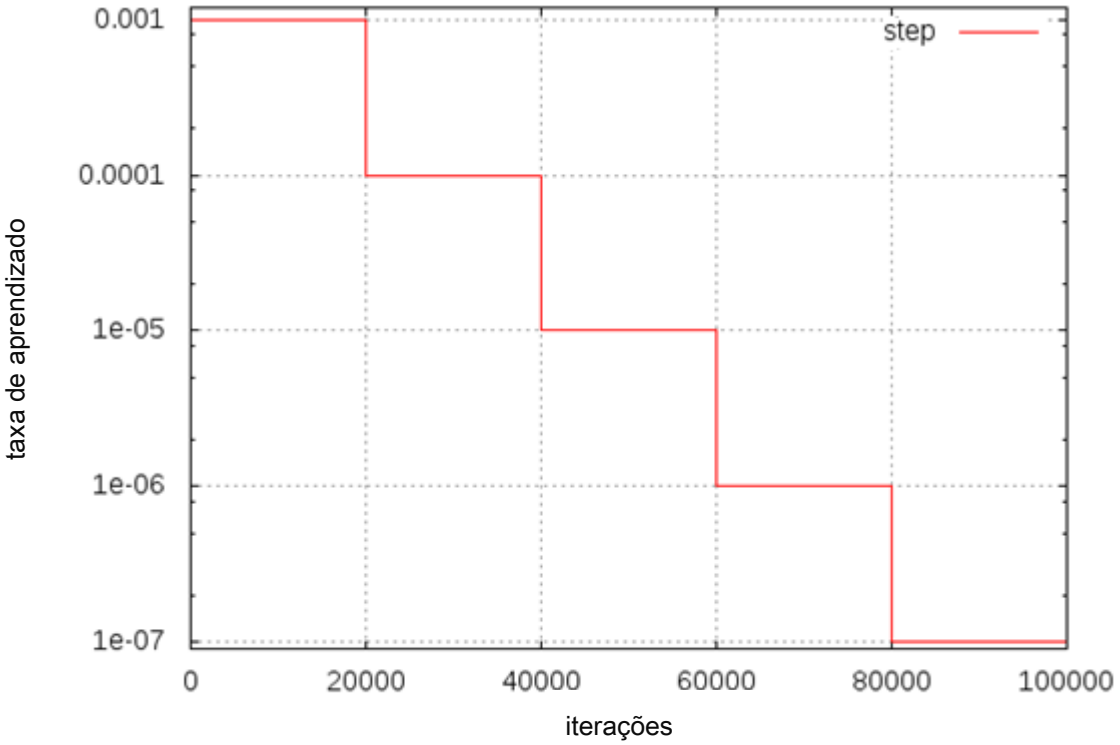


Figura 39. Taxa de aprendizado com a política "step" durante o treinamento com Adam.

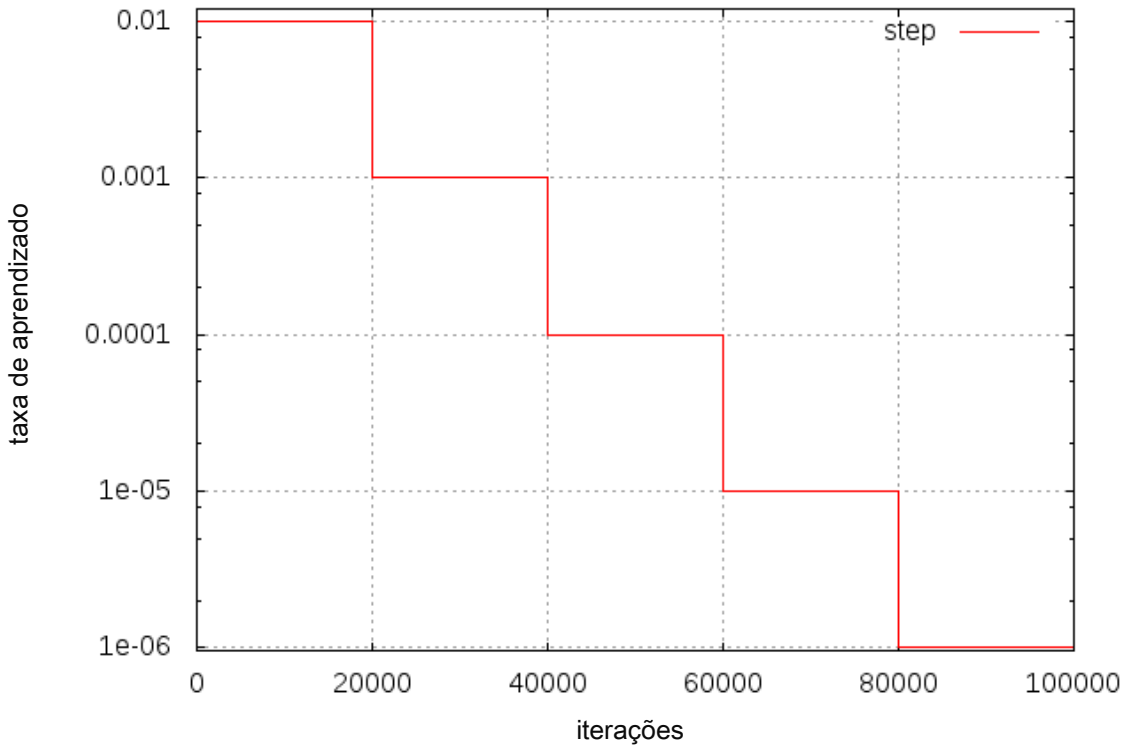


Figura 40. Taxa de aprendizado com a política "*step*" durante o treinamento com AdaDelta.

No caso do RMSProp, a taxa de aprendizado foi variando segundo a seguinte fórmula correspondente à política "inv":

$$lr_{new} = lr_{base} \cdot (1 + \gamma \cdot iter)^{-power} \tag{45}$$

em que, lr_{new} é a nova taxa de aprendizado calculada, lr_{base} é a taxa de aprendizado inicial, γ é um fator constante, *iter* é a iteração atual e *power* controla a velocidade da função de inversão.

Na Figura 41 é apresentado o gráficos da taxa de aprendizado segundo a política "inv" usada com o RMSProp.

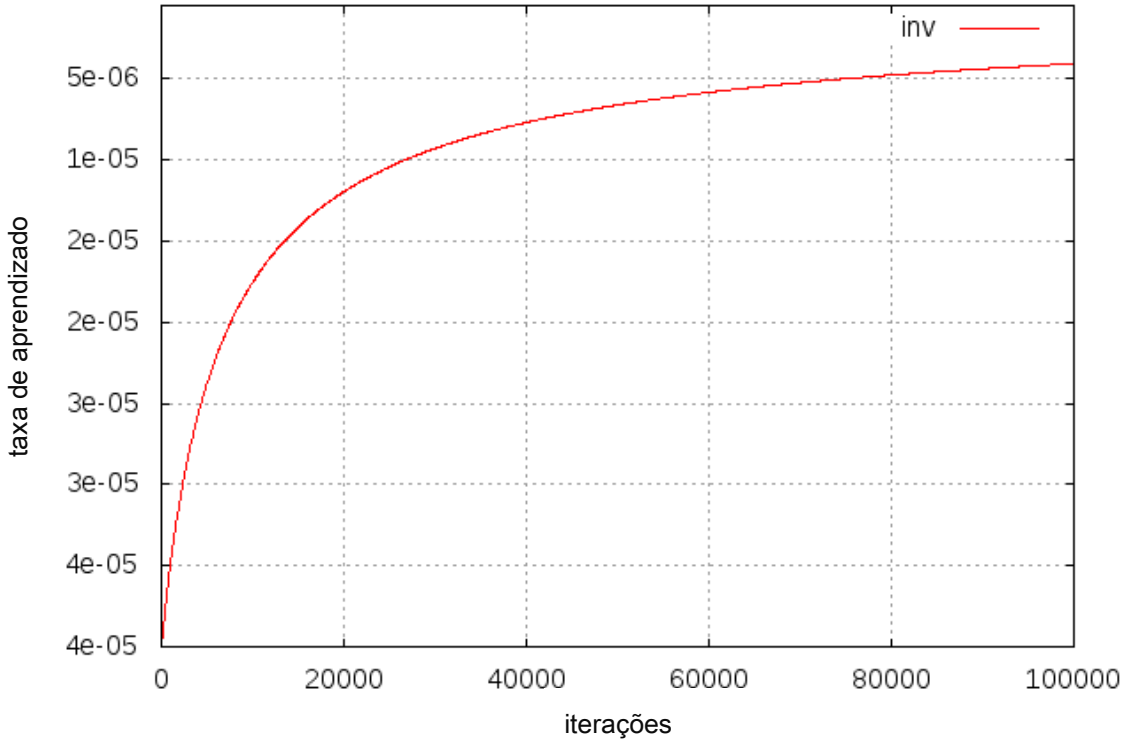


Figura 41. Taxa de aprendizado com a política "*inv*" durante o treinamento com o RMSProp.

Para o algoritmo AdaGrad foi usada a política denominada "poly", onde a taxa de aprendizado segue um decaimento polinomial e será igual a zero quando seja atingido o máximo de iterações, segundo a fórmula mostrada na equação 46:

$$lr_{new} = lr_{base} \cdot (1 - \frac{iter}{\max\_iter})^{-power} \quad (46)$$

em que, lr_{new} é a nova taxa de aprendizado calculada, lr_{base} é a taxa de aprendizado inicial, *iter* é a iteração atual, *max_iter* é o número máximo de iterações e *power* controla o decaimento.

Na Figura 42 é apresentado o gráfico da taxa de aprendizado segundo a política "poly" usada com o AdaGrad.

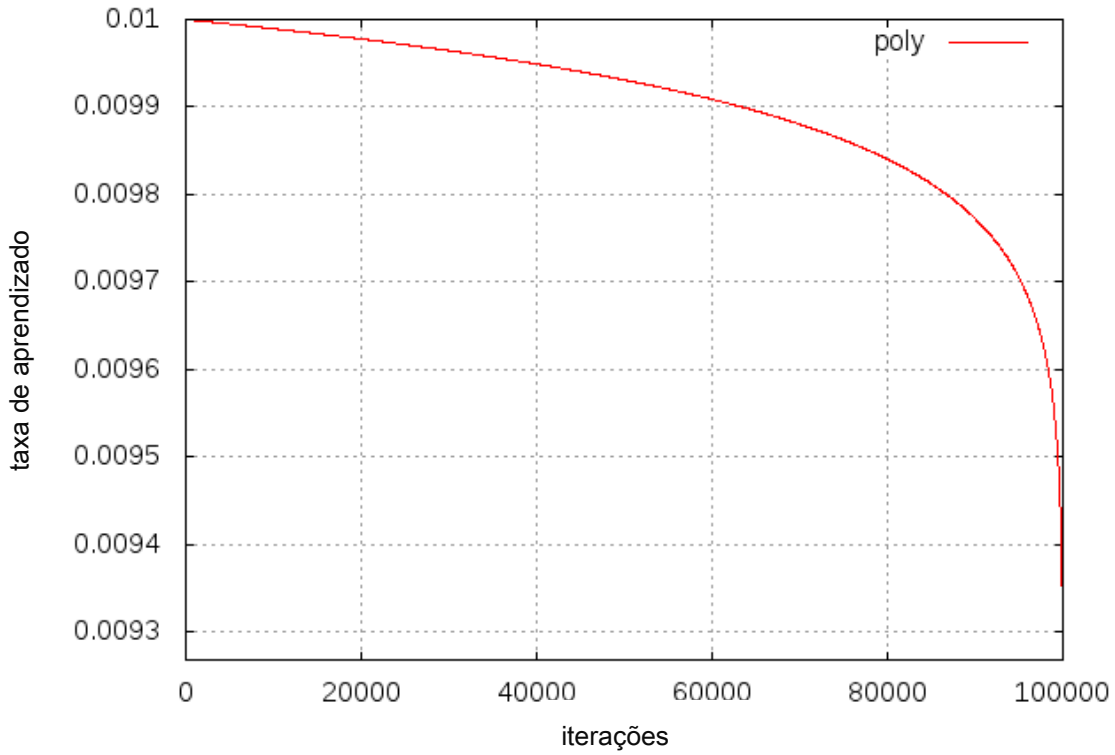


Figura 42. Taxa de aprendizado com a política "*poly*" durante o treinamnto com AdaGrad.

4.2.6 Métricas

Múltiplas métricas têm sido apresentadas para avaliar o desempenho dos métodos de segmentação. As mais comuns incluem as taxas de sucesso, as métricas de similaridade e as métricas de distância.

As taxas de sucesso e as métricas de similaridade comparam a consistência entre o *ground truth* e a segmentação automática. As métricas de distância espacial são amplamente utilizadas na avaliação da segmentação de imagens como medidas de dissimilaridade. Elas medem a diferença de distâncias entre os contornos do *ground truth* e da segmentação automática (SAFARZADEH, 2013).

Neste trabalho serão calculadas a sensibilidade e a especificidade, que são métricas que medem a taxa de sucesso; o coeficiente Dice que é uma métrica de similaridade; e a distância de Hausdorff, que é uma métrica de distância.

A seguir, serão feitas algumas definições que ajudaram ao melhor entendimento das métricas utilizadas: *G* é o conjunto de pixels que pertencem ao objeto segmentado manualmente pelos especialistas, é referido como o *ground truth* ou padrão ouro; *A* é o conjunto de pixels que pertencem ao objeto segmentado por um método de segmentação automático os quais são designados com um valor 1. Todos os outros pixels que não pertencem ao objeto segmentado são designados com valor 0. Um pixel *i* é considerado como:

- Verdadeiro positivo (TP, do inglês *True Positive*) quando seu valor no método automático A_i e seu respectivo valor de *ground truth* G_i são iguais a 1.
- Falso positivo (FP, do inglês *False Positive*) quando A_i é 1 e G_i é 0.
- Falso negativo (FN, do inglês *False Negative*) quando A_i é 0 e G_i é 1.
- Verdadeiro negativo (TN, do inglês *True Negative*) quando A_i e G_i são iguais a 0.

Na Figura 43 são ilustradas as referidas definições em termos de superposição de conjuntos.

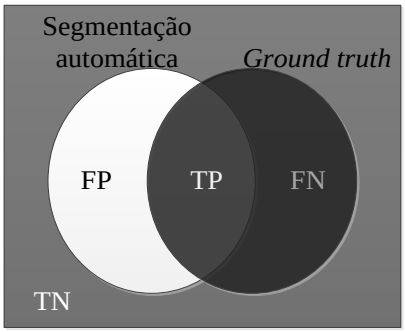


Figura 43. Diagrama de Venn baseado na comparação entre a segmentação automática e o *ground truth*
Fonte: Adaptada de (SAFARZADEH, 2013)

A sensibilidade mede a porção de pixels positivos no *ground truth* que também são identificados como positivos pela segmentação que está sendo avaliada. Analogamente, a especificidade mede a porção de pixels negativos (fundo) no *ground truth* que também são identificados como negativos pela segmentação que está sendo avaliada. As definições da sensibilidade e da especificidade são expressadas nas equações 47 e 48, respectivamente:

$$Sensibilidade = \frac{TP}{TP + FN} \tag{47}$$

$$Especificidade = \frac{TN}{TN + FP} \tag{48}$$

Essas duas métricas são altamente sensíveis ao tamanho do objetivo da segmentação e, por conseguinte, ao desbalanceamento de classes.

O coeficiente de similaridade Dice (DICE, 1945) mede a fracção de sobreposição espacial entre duas imagens binarias. É a métrica mais utilizada na validação de segmentações de imagens médicas (TAHA; HANBURY, 2015). Ela é definida como a intersecção entre duas regiões rotuladas *r* em *G* e *A* sobre a área média dessas duas regiões, conforme é mostrado na equação 49:

$$Dice = \frac{2|G_r \cap A_r|}{|G_r| + |A_r|} = \frac{2|TP|}{(2|TP| + |FP| + |FN|)} \tag{49}$$

Esta métrica varia na faixa de 0 a 1, onde o valor 0 indica que não existe nenhuma sobreposição espacial entre os dois conjuntos de segmentação binaria e 1 indica a sobreposição completa.

A distância de Hausdorff (HD, do inglês *Hausdorff distance*) (HUTTENLOCHER; KLANDERMAN; RUCKLIDGE, 1993) mede a distância entre duas curvas dadas, *G* e *A*, que correspondem aos contornos do *ground truth* e do método de segmentação automático, respectivamente. Em imagens digitais, esses dois contornos podem ser vistos como conjuntos de pontos $G = \{g_1, g_2, ..., g_m\}$ e $A = \{a_1, a_2, ..., a_m\}$, em que cada g_i e a_i representam as coordenadas dos pontos dos contornos. Ela é definida como o máximo das distâncias entre esses dois contornos até o ponto mais próximo entre elas:

$$HD(G, A) = max\,(max_i\{d(g_i, A)\}, max_j\{d(a_j, G)\}) \tag{50}$$

em que a distância de g_i para o ponto mais próximo na curva *A*, $d(g_i, A)$, é definida como:

$$d(g_i, A) = min_j\|a_j - g_i\| \tag{51}$$

em que $||\cdot||$ é alguma norma subjacente sobre os pontos de *A* e *G*, por exemplo a distância euclidiana.

5 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados os resultados obtidos com a arquitetura de rede neural totalmente convolutiva proposta. Inicialmente serão mostrados os gráficos da convergência do aprendizado nos treinamentos com cada método de otimização. Posteriormente, serão mostradas várias tabelas com os resultados numéricos das métricas aplicadas a imagens segmentadas na rede treinada com os diferentes métodos de otimização. Também serão apresentados gráficos que ilustram o desempenho destes métodos levando em conta os valores médios da métrica Dice em todas as imagens ao longo do treinamento. Serão mostrados exemplos de imagens segmentadas organizadas segundo as faixas do coeficiente Dice que resultaram da segmentação. Conforme são expostos os resultados será feita a discussão dos mesmos. Por fim, será realizada uma comparação com trabalhos similares, isto é, com aqueles onde foi utilizada a mesma base de dados.

5.1 Gráficos da convergência do aprendizado

Nas Figuras 44, 45, 46, 47, 48 e 49 são apresentados os gráficos da convergência da rede durante o treinamento com os algoritmos de otimização SGD, Nesterov, RMSProp, Adam, AdaDelta e AdaGrad, respectivamente. No eixo *x* encontram-se as iterações e no eixo *y* os valores de perda. A perda é calculada conforme à equação 42, apresentada anteriormente.

Com o decorrer das iterações, os pesos da rede vão se-ajustando de forma que o valor da perda vai diminuindo, isto é, o algoritmo de otimização vai convergindo para o mínimo. Este processo é totalmente estocástico, a inicialização dos pesos é aleatória,

fato pelo qual, em treinamentos sucessivos sobre um mesmo conjunto de dados, os pesos não vão ficar iguais.

Os métodos de otimização utilizados apresentam diferentes velocidades de convergência, sendo o RMSProp, Nesterov e AdaGrad os mais rápidos em convergir nas primeiras 20.000 iterações. Foi observado que, de forma geral, o valor da perda fica menor e mais estável após a iteração 20.000, significando que a rede não aprende muito mais.

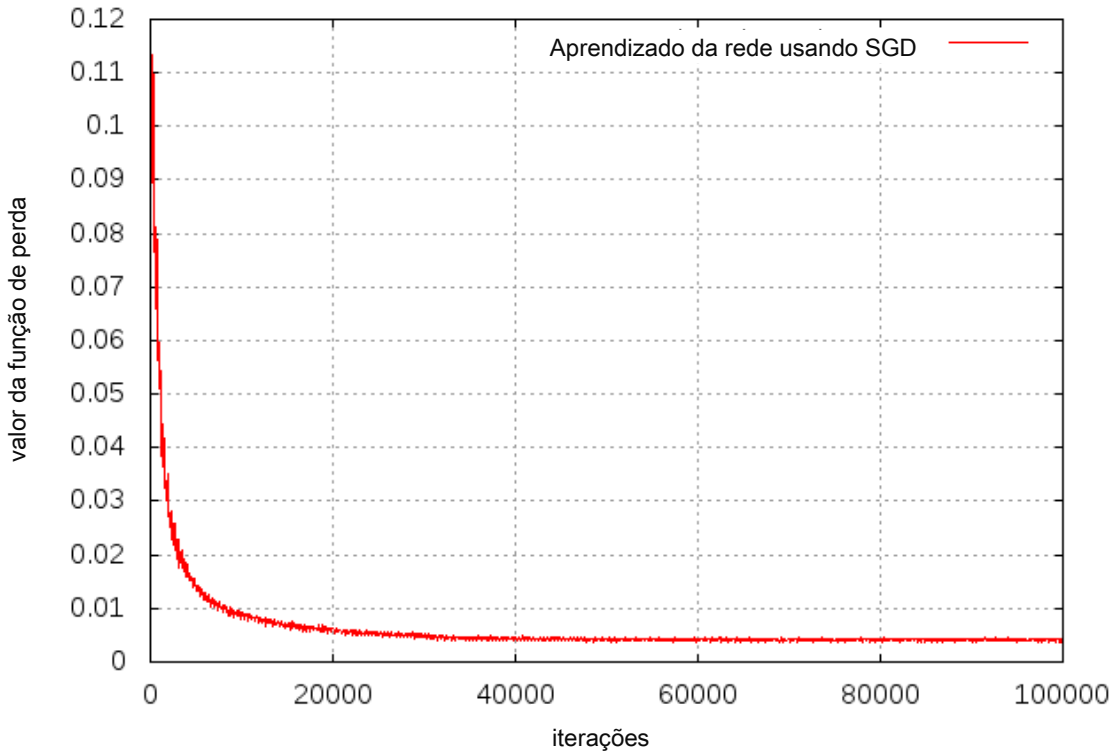


Figura 44. Aprendizado da rede treinada com o algoritmo SGD.

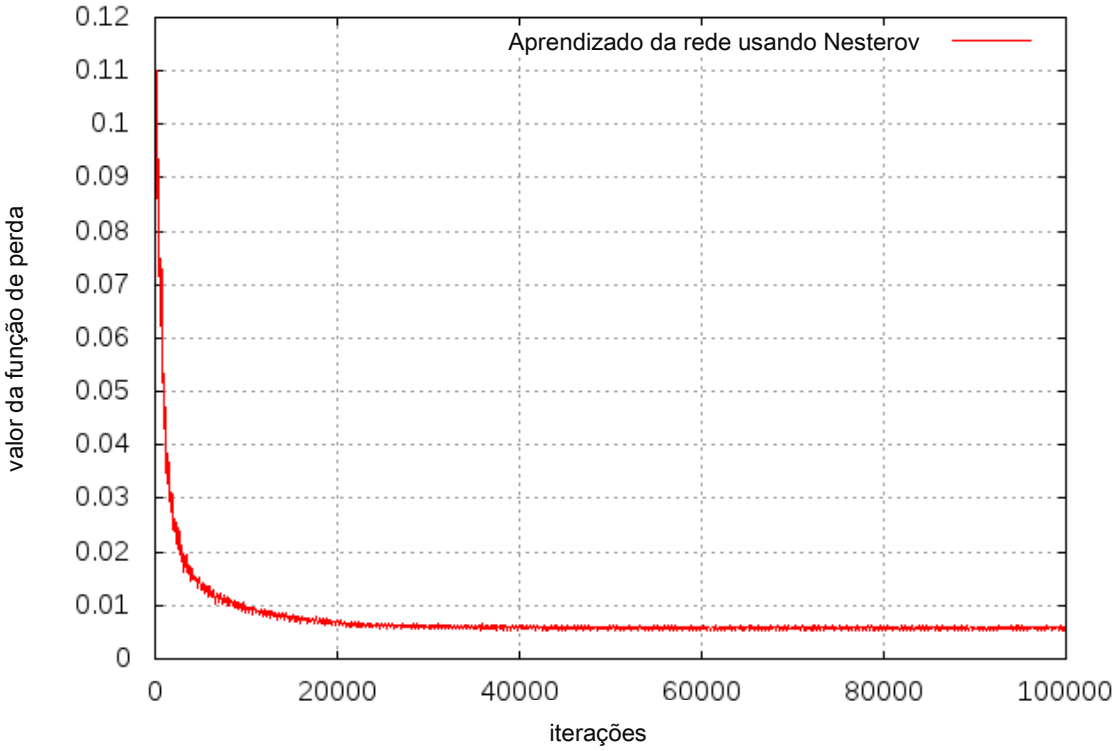


Figura 45. Aprendizado da rede treinada com o algoritmo Nesterov.

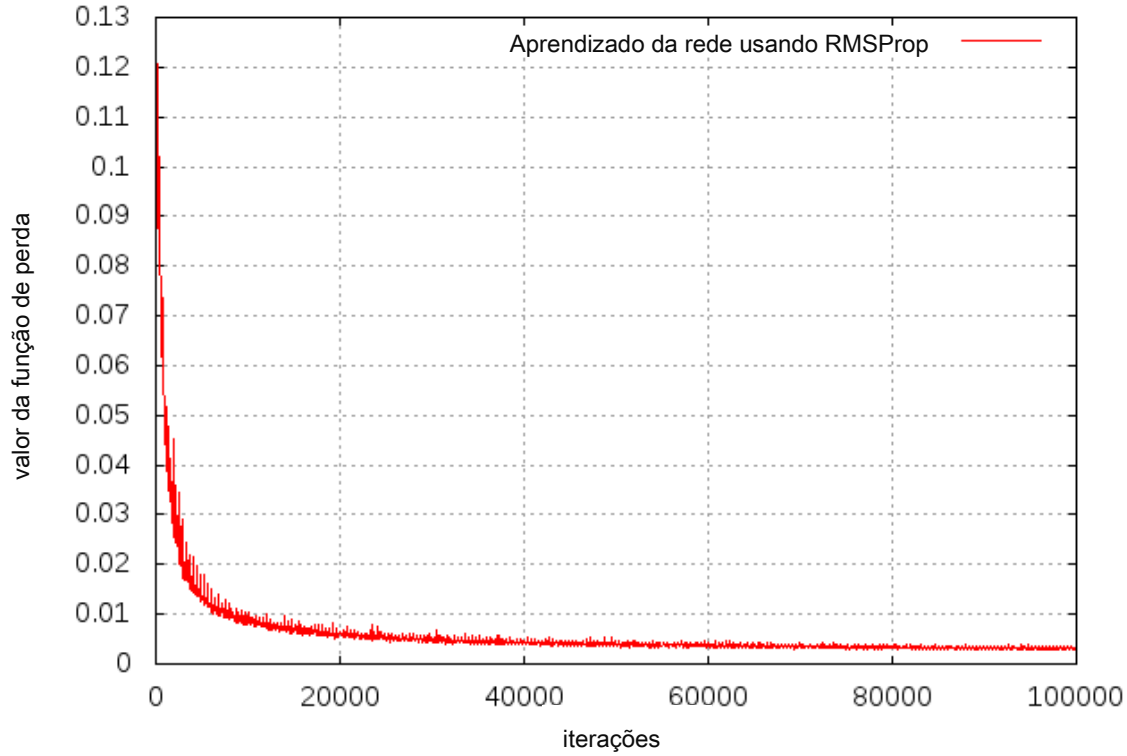


Figura 46. Aprendizado da rede treinada com o algoritmo RMSProp.

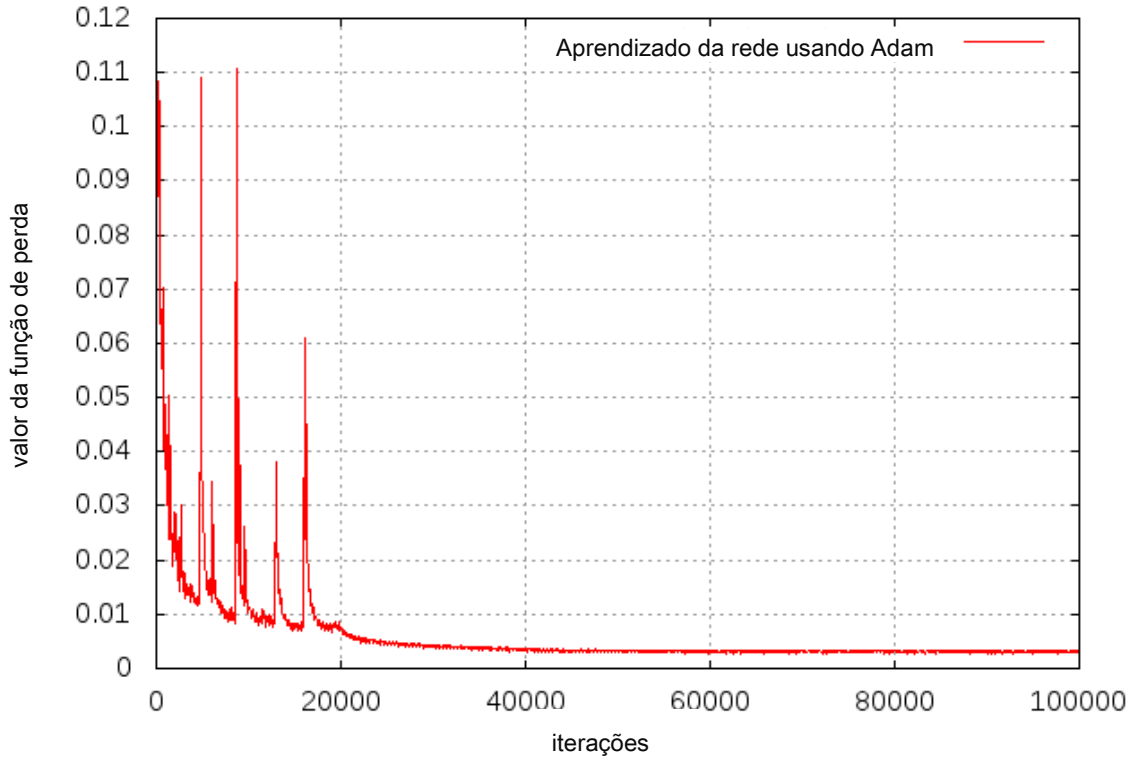


Figura 47. Aprendizado da rede treinada com o algoritmo Adam.

Figura 48. Aprendizado da rede treinada com o algoritmo AdaDelta.

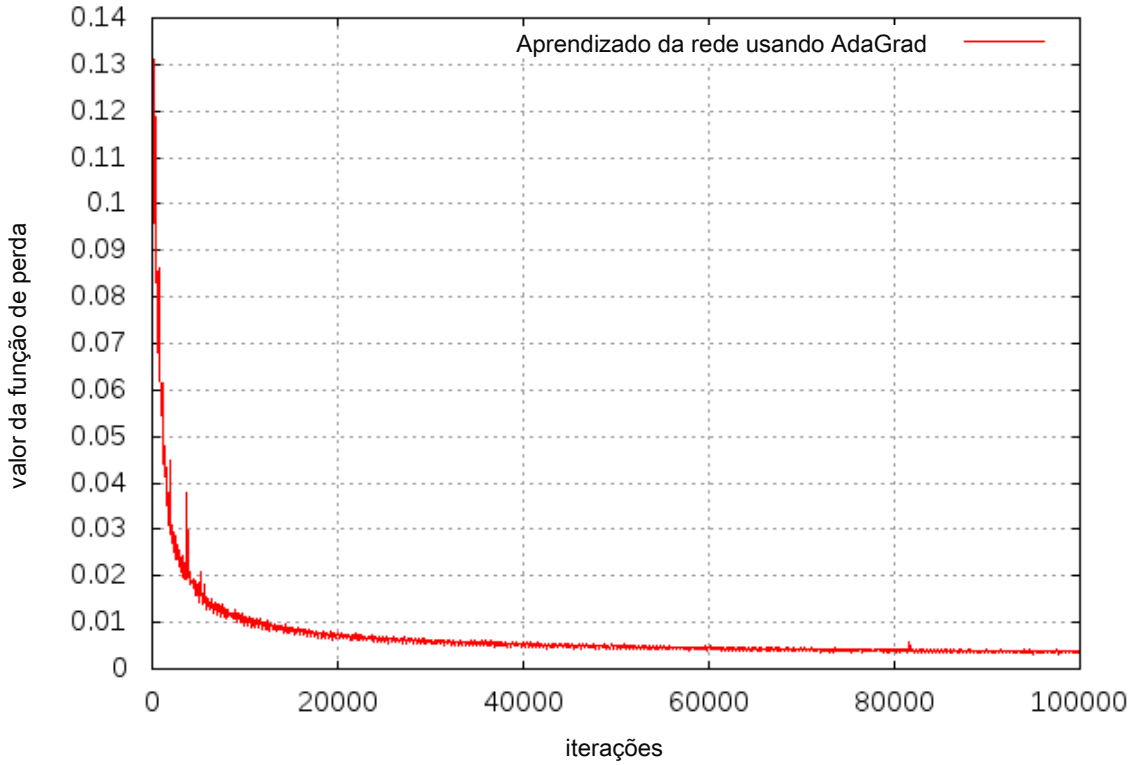


Figura 49. Aprendizado da rede treinada com o algoritmo AdaGrad.

5.2 Resultados da segmentação segundo as métricas de desempenho

As métricas explicadas anteriormente foram calculadas nas 420 imagens usando como referência os *ground truths* fornecidos pela base de dados Sunnybrook. Nas Tabelas 4, 5, 6, 7, 8 e 9 são apresentados os valores médios das métricas aplicadas a cada uma das 42 imagens de teste das dez pastas para o endocárdio/epicárdio na rede treinada com o SGD, Nesterov, RMSProp, Adam, AdaDelta e AdaGrad, respectivamente. No final de cada tabela encontra-se a média e o desvio padrão dos resultados de cada conjunto. Os valores numéricos obtidos em cada pasta e a média final foram expressados até a quarta casa decimal; os desvios padrões foram expressados até a segunda casa decimal exceto no caso da especificidade que, por serem valores muito pequenos, foram expressados até a quinta casa decimal.

Todos os valores mostrados nas tabelas são correspondentes ao estado da rede na iteração 60.000 devido a que, conforme os gráficos de aprendizado apresentados, nesse ponto do treinamento os seis métodos de otimização utilizados estão estáveis e mais à frente não têm uma melhoria significativa.

Dentre as métricas usadas, a Dice é a que fornece maior informação sobre o desempenho da metodologia e é a mais usada em trabalhos similares, como pode ser corroborado na tabela comparativa apresentada no capítulo de revisão bibliográfica (Tabela 1). Tomando essa métrica como referência para comparação, podemos dizer que, na iteração 60.000, o método RMSProp teve o melhor desempenho tanto para segmentar o contorno do endocárdio quanto para o epicárdio. Ele foi ligeiramente melhor que o SGD, que obteve o segundo lugar. O pior resultado foi com o AdaDelta. A Figura 50 permite a fazer essa comparação rapidamente.

Os valores de distância de Hausdorff obtidos representam o quanto pode ser a separação máxima entre os contornos em um dado ponto. Os melhores resultados, ou seja, os menores valores, foram obtidos com a rede treinada com os algoritmos de Adam e Nesterov.

A sensibilidade dá ideia do desempenho da classificação dos pixels pertencentes à área do endocárdio (ou epicárdio), isto é, a classificação de um pixel como sendo da área alvo quando de fato é. Os métodos de otimização com os quais a rede apresentou maior sensibilidade foram o SGD, o Nesterov e o RMSProp; os de piores resultados foram o Adam, o AdaDelta e o AdaGrad.

No que diz respeito à especificidade, pode ser observado nas tabelas que, em todos os casos, foram obtidos altos valores. Isso permite afirmar que a rede tem uma grande precisão ao classificar pixels que não pertencem ao endocárdio (ou epicárdio).

Tabela 4 Resultados experimentais para o endocárdio (endo) e epicárdio (epi) na iteração 60.000 usando o SGD.

Conjunto		Dice	Hausdorff	Sensibilidade	Especificidade
1	endo	0,8957	6,9867	0,9210	0,9983
	epi	0,9114	6,7480	0,9089	0,9987
2	endo	0,9138	10,8224	0,9498	0,9979
	epi	0,9311	11,9072	0,9287	0,9986
3	endo	0,8857	12,6731	0,9247	0,9985
	epi	0,9009	12,4068	0,9191	0,9983
4	endo	0,8914	11,7551	0,9157	0,9986
	epi	0,9022	11,5736	0,8997	0,9986
5	endo	0,9146	9,0121	0,9254	0,9989
	epi	0,9091	9,9374	0,8939	0,9989
6	endo	0,9411	5,9172	0,9634	0,9986
	epi	0,9537	6,1966	0,9574	0,9986
7	endo	0,9413	8,1275	0,9467	0,9990
	epi	0,9445	8,3280	0,9416	0,9989
8	endo	0,8781	19,1864	0,8864	0,9987
	epi	0,8837	18,7904	0,8788	0,9986
9	endo	0,9026	11,4145	0,9250	0,9986
	epi	0,9125	11,7840	0,9197	0,9985
10	endo	0,8910	9,3495	0,9055	0,9988
	epi	0,8976	9,5681	0,8881	0,9989
média ± desvio padrão	endo	0,9055 ± 0,02	10,5244 ± 3,53	0,9263 ± 0,02	0,9985 ± 0,0002
	epi	0,9146 ± 0,02	10,7240 ± 3,39	0,9135 ± 0,02	0,9986 ± 0,0001

Tabela 5 Resultados experimentais para o endocárdio (endo) e epicárdio (epi) na iteração 60.000 usando o Nesterov.

Conjunto		Dice	Hausdorff	Sensibilidade	Especificidade
1	endo	0,8780	6,9978	0,8978	0,9985
	epi	0,8908	9,4110	0,8823	0,9989
2	endo	0,9280	8,8424	0,9544	0,9980
	epi	0,9402	8,6019	0,9288	0,9986
3	endo	0,8838	10,5976	0,9196	0,9985
	epi	0,8982	10,0639	0,9093	0,9985
4	endo	0,8789	12,1022	0,9067	0,9984
	epi	0,8984	12,7337	0,8893	0,9989
5	endo	0,9109	7,8806	0,9352	0,9986
	epi	0,9086	9,2821	0,8895	0,9989
6	endo	0,9366	6,8431	0,9625	0,9985
	epi	0,9550	6,8214	0,9576	0,9987
7	endo	0,9213	7,5485	0,9380	0,9986
	epi	0,9346	7,9875	0,9267	0,9989
8	endo	0,8783	10,8211	0,8949	0,9986
	epi	0,8830	12,5763	0,8700	0,9990
9	endo	0,8960	8,5671	0,9331	0,9980
	epi	0,9129	10,3877	0,9162	0,9986
10	endo	0,8881	5,7262	0,9144	0,9985
	epi	0,9016	6,2668	0,8875	0,9990
média ± desvio padrão	endo	0,8999 ± 0,02	8,5926 ± 1,91	0,9256 ± 0,02	0,9984 ± 0,0002
	epi	0,9123 ± 0,02	9,4132 ± 2,04	0,9057 ± 0,02	0,9988 ± 0,0001

Tabela 6 Resultados experimentais para o endocárdio (endo) e epicárdio (epi) na iteração 60.000 usando o RMSProp.

Conjunto		Dice	Hausdorff	Sensibilidade	Especificidade
1	endo	0,9008	8,0374	0,9111	0,9989
	epi	0,9061	8,5642	0,9020	0,9989
2	endo	0,9342	9,6225	0,9404	0,9986
	epi	0,9448	11,8569	0,9385	0,9985
3	endo	0,8986	10,2723	0,9176	0,9989
	epi	0,9037	11,1759	0,9095	0,9986
4	endo	0,8960	8,2746	0,9038	0,9990
	epi	0,9054	8,9429	0,8914	0,9991
5	endo	0,9092	11,7059	0,9147	0,9989
	epi	0,9087	11,7416	0,8927	0,9989
6	endo	0,9443	7,8723	0,9525	0,9989
	epi	0,9534	7,8710	0,9515	0,9988
7	endo	0,9364	3,5353	0,9318	0,9992
	epi	0,9389	3,9390	0,9294	0,9990
8	endo	0,8870	9,9028	0,8938	0,9986
	epi	0,8950	14,0302	0,8861	0,9986
9	endo	0,8997	11,6084	0,9285	0,9983
	epi	0,9218	10,0571	0,9265	0,9984
10	endo	0,8922	9,5896	0,9060	0,9987
	epi	0,8901	9,4844	0,8888	0,9985
média ± desvio padrão	endo	0,9098 ± 0,01	9,0421 ± 2,23	0,9200 ± 0,01	0,9988 ± 0,0002
	epi	0,9167 ± 0,02	9,7663 ± 2,61	0,9116 ± 0,02	0,9987 ± 0,0002

Tabela 7 Resultados experimentais para o endocárdio (endo) e epicárdio (epi) na iteração 60.10000 usando o Adam.

Conjunto		Dice	Hausdorff	Sensibilidade	Especificidade
1	endo	0,8960	5,6807	0,8670	0,9995
	epi	0,9154	8,0149	0,9168	0,9986
2	endo	0,9213	8,6739	0,8971	0,9993
	epi	0,9390	8,3411	0,9191	0,9990
3	endo	0,8758	7,7567	0,8485	0,9995
	epi	0,9006	6,0017	0,9156	0,9984
4	endo	0,8845	10,1142	0,8708	0,9993
	epi	0,9064	13,8600	0,9188	0,9983
5	endo	0,8908	5,6646	0,8577	0,9995
	epi	0,9091	6,6294	0,8953	0,9989
6	endo	0,9244	3,5136	0,8797	0,9997
	epi	0,9510	4,2945	0,9503	0,9987
7	endo	0,9071	9,6802	0,8691	0,9996
	epi	0,9255	10,8606	0,9290	0,9986
8	endo	0,8499	13,5317	0,8219	0,9996
	epi	0,8651	15,1004	0,8651	0,9986
9	endo	0,8863	5,9482	0,8499	0,9996
	epi	0,9167	6,8948	0,9321	0,9977
10	endo	0,8792	9,2024	0,8772	0,9992
	epi	0,8906	7,6515	0,8782	0,9992
média ± desvio padrão	endo	0,8915 ± 0,02	7,9766 ± 2,73	0,8638 ± 0,01	0,9994 ± 0,0001
	epi	0,9119 ± 0,02	8,7648 ± 3,28	0,9120 ± 0,02	0,9986 ± 0,0003

Tabela 8 Resultados experimentais para o endocárdio (endo) e epicárdio (epi) na iteração 60.000 usando o AdaDelta.

Conjunto		Dice	Hausdorff	Sensibilidade	Especificidade
1	endo	0,8846	12,8023	0,9100	0,9981
	epi	0,8921	10,5783	0,8817	0,99833
2	endo	0,8961	14,3094	0,9133	0,9982
	epi	0,9021	13,7144	0,8824	0,9986
3	endo	0,8693	13,9529	0,9066	0,9983
	epi	0,8774	12,3495	0,8820	0,9983
4	endo	0,8625	9,1949	0,8738	0,9987
	epi	0,8761	16,4888	0,8542	0,9988
5	endo	0,8747	16,0415	0,8933	0,9984
	epi	0,8794	16,7613	0,8616	0,9986
6	endo	0,9120	7,8458	0,9179	0,9985
	epi	0,9220	9,1696	0,9082	0,9985
7	endo	0,9072	5,8261	0,9131	0,9984
	epi	0,9118	7,7052	0,8917	0,9986
8	endo	0,8562	19,8294	0,8716	0,9982
	epi	0,8572	19,2331	0,8461	0,9981
9	endo	0,8682	12,3030	0,9000	0,9978
	epi	0,8802	13,2016	0,8845	0,9980
10	endo	0,8386	10,1073	0,8513	0,9985
	epi	0,8500	11,9388	0,8349	0,9987
média ± desvio padrão	endo	0,8769 ± 0,02	12,2212 ± 3,92	0,8950 ± 0,02	0,9983 ± 0,0002
	epi	0,8848 ± 0,02	13,1140 ± 3,40	0,8727 ± 0,02	0,9984 ± 0,0002

Tabela 9 Resultados experimentais para o endocárdio (endo) e (epi) epicárdio na iteração 60.000 usando o AdaGrad.

Conjunto		Dice	Hausdorff	Sensibilidade	Especificidade
1	endo	0,8994	5,3026	0,9043	0,9988
	epi	0,9239	5,2799	0,9165	0,9988
2	endo	0,9104	10,9515	0,8943	0,9990
	epi	0,9179	11,6263	0,9128	0,9983
3	endo	0,8751	8,3599	0,8850	0,9989
	epi	0,8873	15,7366	0,9006	0,9983
4	endo	0,8922	9,6354	0,9121	0,9988
	epi	0,9023	11,3537	0,8964	0,9988
5	endo	0,8294	11,7596	0,8148	0,9991
	epi	0,8561	10,825	0,8228	0,9993
6	endo	0,9228	4,6320	0,9294	0,9989
	epi	0,9483	6,9124	0,9418	0,9988
7	endo	0,9187	5,5428	0,9106	0,9991
	epi	0,9220	9,5607	0,9127	0,9990
8	endo	0,8589	14,7552	0,8600	0,9987
	epi	0,8830	18,1752	0,8902	0,9981
9	endo	0,9055	7,5699	0,9153	0,9989
	epi	0,9128	7,2381	0,9128	0,9988
10	endo	0,8665	8,3100	0,8586	0,9991
	epi	0,8906	10,7787	0,8802	0,9989
média ± desvio padrão	endo	0,8879 ± 0,02	8,6819 ± 3,02	0,8884 ± 0,03	0,9989 ± 0,0001
	epi	0,9044 ± 0,02	10,7486 ± 3,73	0,8986 ± 0,02	0,9987 ± 0,0003

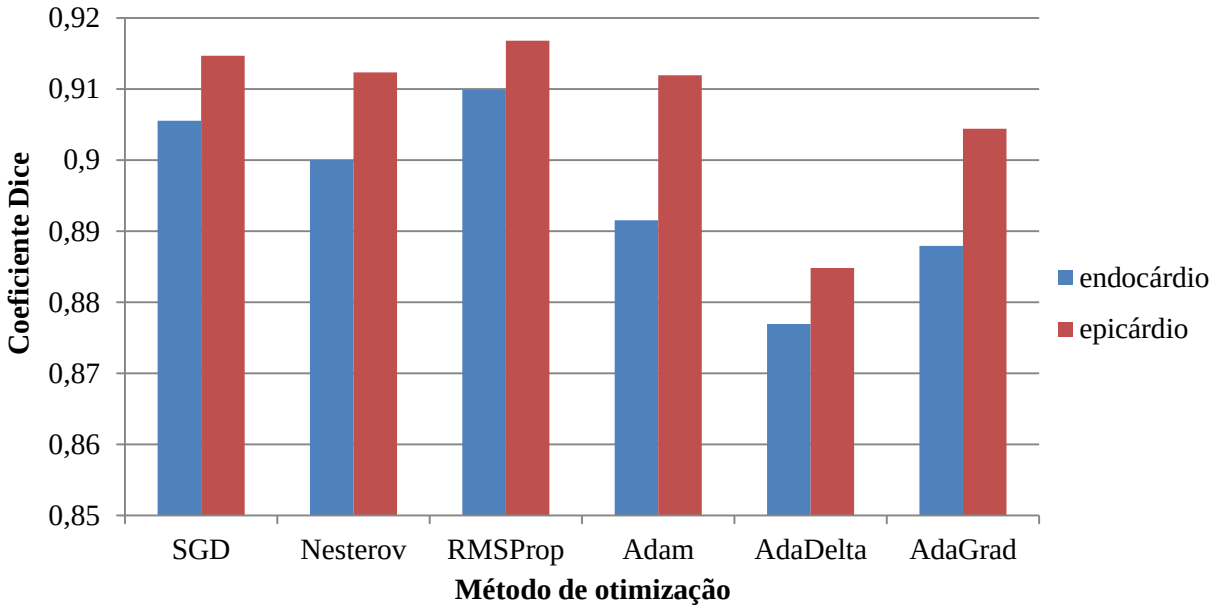


Figura 50. Valores médios do coeficiente Dice no endocárdio e epicárdio obtidos com cada método de otimização e calculados no total de imagens.

As Figuras 51 e 52 apresentam os valores médios do coeficiente Dice do endocárdio e epicárdio respectivamente, obtidos com os diferentes métodos de otimização a cada 5000 iterações. Esses gráficos fornecem uma visão geral do comportamento da rede quando treinada com os diferentes métodos de otimização. Eles reúnem uma grande quantidade de informação devido a que cada ponto no gráfico corresponde ao valor médio do coeficiente Dice calculado em todas as imagens da base de dados.

Pode ser observado que, no caso do endocárdio, os melhores resultados foram obtidos quando a rede foi treinada com os métodos de otimização SGD e RMSProp, os quais alternam a primeira posição no gráfico. Os piores resultados foram alcançados com o AdaGrad e o AdaDelta.

No caso do epicárdio, igualmente os melhores resultados foram obtidos quando a rede foi treinada com os métodos de otimização SGD e RMSProp, alternando a primeira posição em diferentes iterações, embora o Nesterov e o Adam encontram-se

bem próximos. De igual forma que para o endocárdio, os piores resultados foram alcançados com o AdaGrad e o AdaDelta.

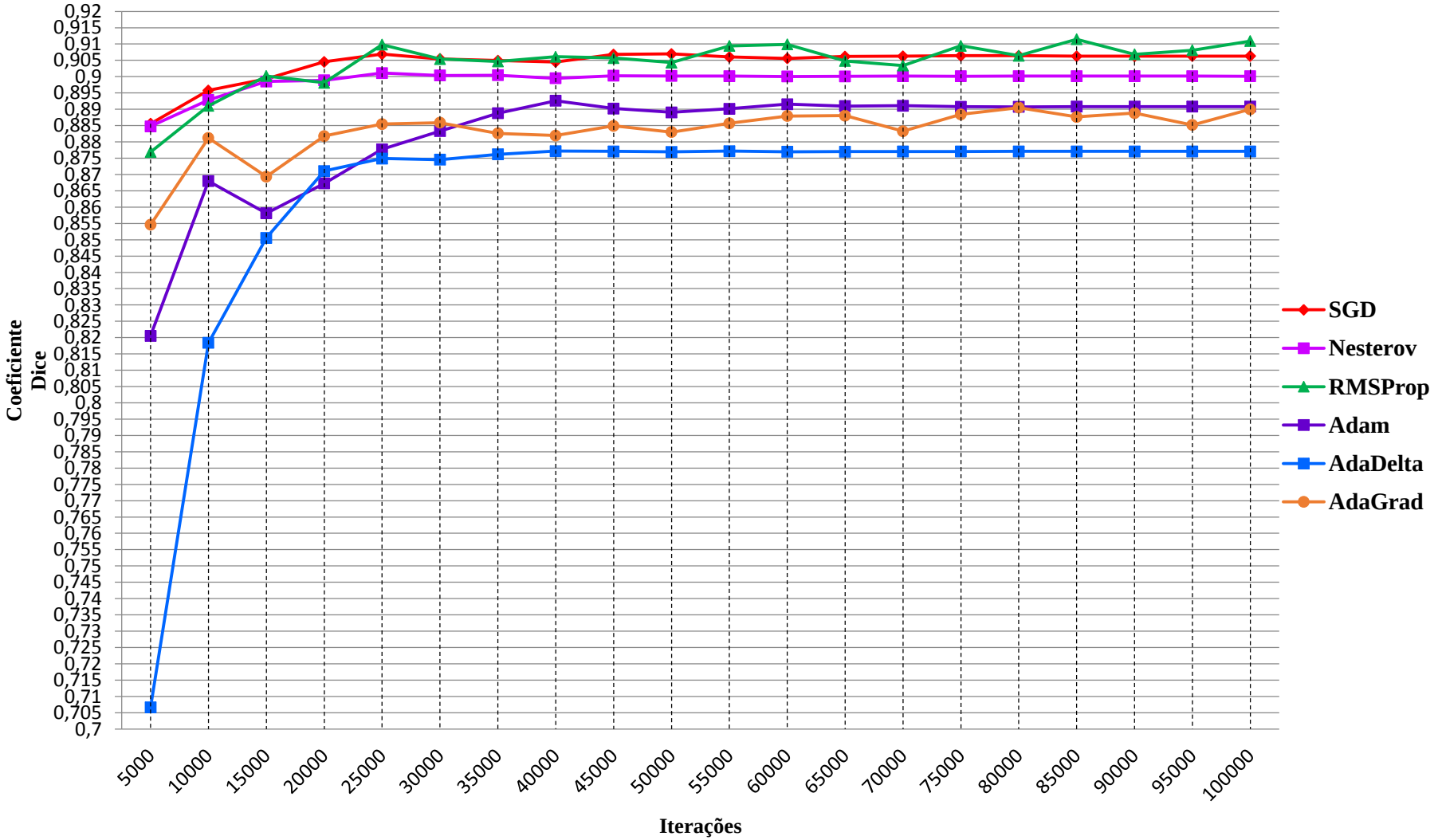


Figura 51. Valores médios do coeficiente Dice do endocárdio obtidos com os diferentes métodos de otimização cada 5000 iterações.

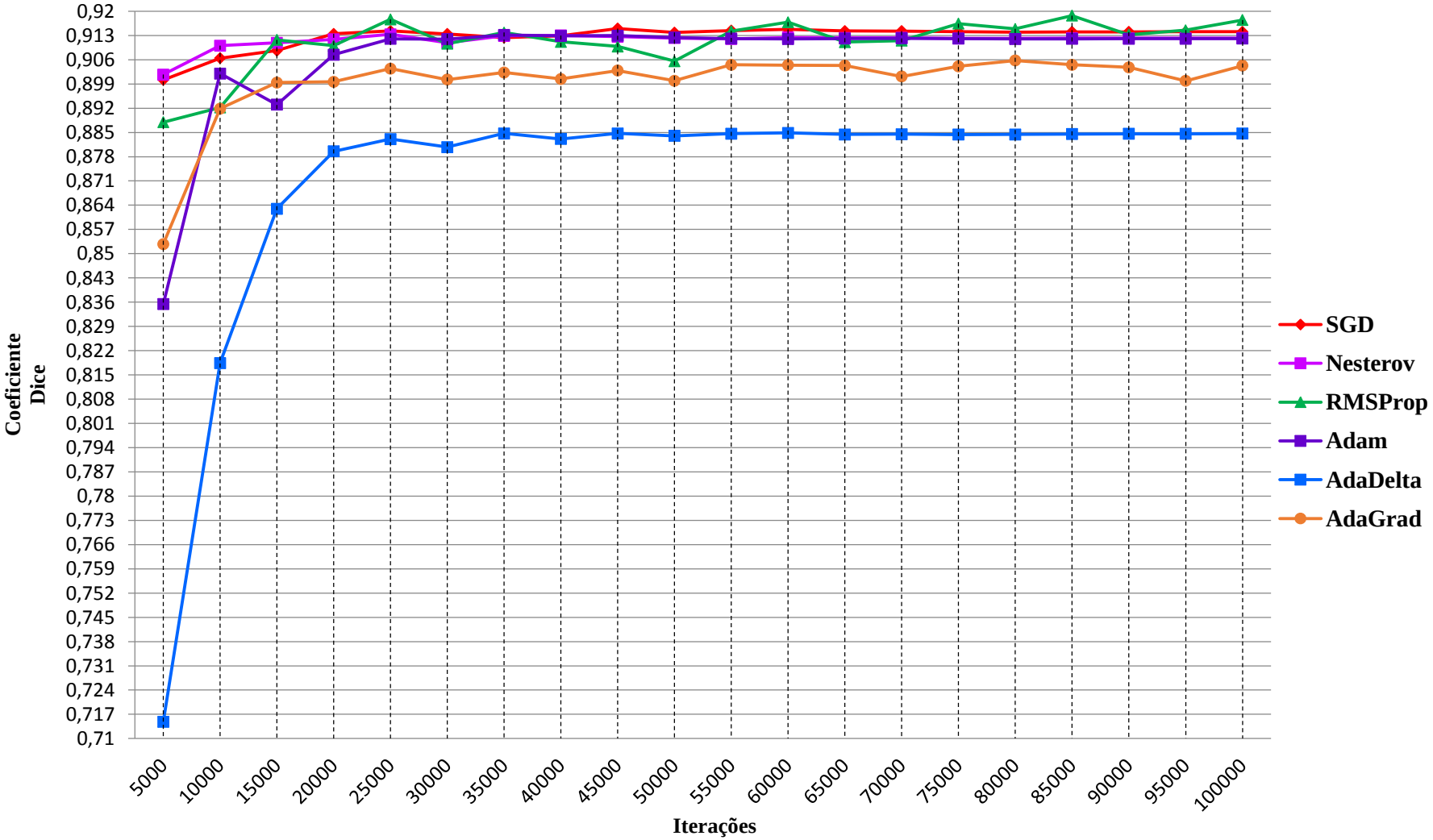


Figura 52. Valores médios do coeficiente Dice do epicárdio obtidos com os diferentes métodos de otimização cada 5000 iterações.

5.3 Imagens segmentadas

Nas Figuras 53, 54, 55 e 56 são mostradas as segmentações obtidas pela rede em imagens de pacientes com insuficiência cardíaca que tiveram infarto, com insuficiência cardíaca sem infarto, com hipertrofia e saudáveis, respectivamente. A linha verde representa os contornos obtidos pela rede, enquanto a linha vermelha representa os contornos *ground truth*. Em todas as imagens apresentadas, os valores do coeficiente de similaridade Dice, tanto para o endocárdio quanto para o epicárdio, são maiores que 0,90. Em cada caso é indicada a pasta e o nome da imagem dentro dela. Fazendo uma análise qualitativa por inspeção visual, pode ser verificado que as segmentações apresentam uma boa precisão, ou seja, os contornos tem um ótimo ajuste aos do padrão ouro.

Na Figura 57 são mostradas segmentações que obtiveram valores de coeficiente Dice entre 0,80 e 0,90 para o endocárdio e o epicárdio. Geralmente nestes casos encontram-se imagens da base e do apex cardíaco, nas quais aumenta a complexidade da segmentação. Em alguns cortes da base cardíaca o miocárdio aparece incompleto. Nas imagens do apex cardíaco a cavidade ventricular esquerda torna-se difusa e às vezes até imperceptível. Esses fatos afetam o desempenho dos algoritmos de segmentação conforme foi reportado no trabalho de Krasnobaev e Sozykin (2016). As segmentações obtidas demonstram a robustez diante da variabilidade de morfologia, posicionamento da cavidade ventricular e patologias presentes nas imagens.

Na Figura 58 são apresentados alguns casos em que a rede teve dificuldade para segmentar. Na coluna esquerda estão as imagens originais e sua pasta correspondente e na direita o resultado da segmentação. A primeira, terceira e quarta filas contêm imagens de pacientes com insuficiência cardíaca e infarto, hipertrofia e saudável,

respectivamente. Elas pertencem a cortes no apex cardíaco que tem características peculiares mencionadas anteriormente. A segunda fila apresenta uma imagem basal de um paciente com insuficiência cardíaca (não infartado), na qual a rede realizou uma segmentação de forma parcial.

A pesar de que nesses casos, mostrados na Figura 58, a rede neural totalmente convolutiva não alcançou um ajuste ótimo dos contornos, foi comprovado que ela sabe reconhecer a localização espacial do miocárdio na imagem. Incluir um maior número de imagens da base e do apex cardíaco no conjunto de treinamento seria uma boa estratégia para tentar melhorar o aprendizado da rede e, por conseguinte, seu desempenho nestas imagens de maior complexidade.

5.4 Comparação com o estado da arte

Os resultados obtidos foram comparados com os trabalhos analisados no Capítulo 2 que utilizaram a mesma base de dados. A Tabela 10 apresenta um resumo com as principais informações. Na segunda e terceira colunas são mostradas algumas observações que consideramos importantes para a comparação. O fato de que o método seja semiautomático ou automático tem um grande peso devido a que estes últimos são os mais desejados por ter maior independência da interação humana. Além disso, no caso do uso da ROI, é especificado se a extração é feita de forma fixa ou por meio de um método automático; por exemplo, os trabalhos de Wang e colaboradores (2015) e Tran (2016) a ROI é extraída supondo que a cavidade ventricular esquerda está localizada sempre no meio da imagem. Isso pode trazer imprecisões se fosse aplicado em outros dados.

O número de estudos da base de dados que são incluídos na avaliação tem uma influência direta nos valores numéricos reportados, alguns autores utilizaram 7, 15, 30 ou 45. Neste trabalho a comparação está baseada no coeficiente de similaridade Dice por ser a métrica comum entre os métodos confrontados. Dentre os resultados da referida métrica, obtidos com os diferentes algoritmos de otimização na iteração 60.000, foi exibido o correspondente ao RMSProp por ser o maior deles.

O modelo proposto, treinado com o algoritmo de otimização RSMProp, alcançou um coeficiente de similaridade médio aproximado de 0,91 na segmentação do endocárdio e 0,92 na segmentação do epicárdio, levando em conta o total de estudos da base de dados (45 *datasets*). Ele não precisa extração de ROI e é totalmente automático.

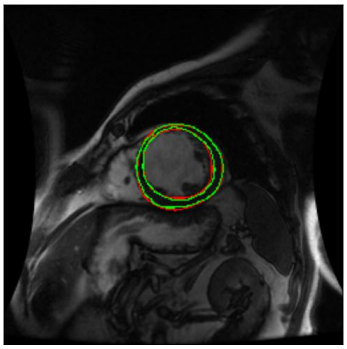
Na segmentação do contorno do endocárdio nas imagens da base Sunnybrook, a faixa do coeficiente Dice varia entre 0,82 e 0,94; no caso do epicárdio varia entre 0,91 e 0,96. Wang e colaboradores (2015) só apresentaram o resultado atingido para o endocárdio. Esse valor, de 0,94, é o maior entre os trabalhos comparados. No entanto, foi obtido considerando as imagens de apenas sete estudos. De forma similar, o trabalho de Tran (2016) reportou o melhor resultado para o epicárdio, utilizando um menor número de estudos (30).

O trabalho de Dreijer, Herbst e Du Preez (2013) apresentou resultados similares avaliados em 15 conjuntos. Esse método é semiautomático e de alta complexidade computacional. Dentre os trabalhos elencados na Tabela 10, o de Constantindes e colaboradores (2012) foi o único que também avaliou utilizando os 45 conjuntos de imagens, o qual permite o *benchmark* com os resultados deste trabalho. Os referidos autores obtiveram coeficientes Dice de 0,86 e 0,91 para o endocárdio e epicárdio, respectivamente. Esses valores foram superados pelo método proposto.

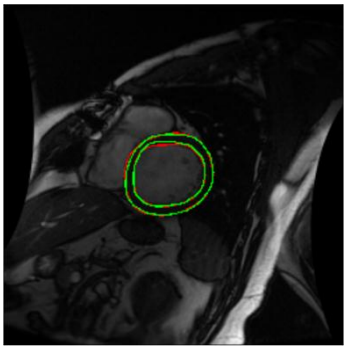
Em resumo, o trabalho desenvolvido evidenciou um bom desempenho na segmentação do miocárdio e apresentou algumas vantagens em comparação com o estado da arte. O método é totalmente automático pelo qual não precisa de inicialização de parâmetros pelo usuário. Uma vez que a rede é treinada, basta com apresentar uma imagem na sua entrada e ela vai dar uma saída com a segmentação estimada. No que diz respeito ao custo computacional, a fase de treinamento é a de maior demanda devido ao ajuste iterativo dos pesos utilizando os gradientes. No caso da fase de teste é bem mais leve, já que só precisa fazer a inferência da saída a partir da imagem de entrada. O tempo para realizar isto, nas condições de *hardware* descritas, é de aproximadamente um segundo.

Tabela 10. Comparação do desempenho da segmentação entre o método proposto e outros trabalhos que utilizam a mesma base de dados.

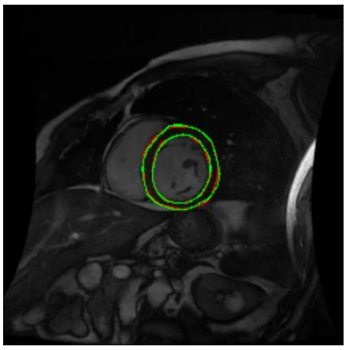
Método	Observações	# conjuntos de imagens	Métrica Dice média ± desvio padrão (endocárdio/epicárdio)
(Constantindes et al., 2012)	Método automático Extração automática de ROI	45 (15 treinamento, 15 validação, 15 *online*)	0,86±0,05/0,91±0,03
(Uzunbas et al., 2012)	Método semiautomático	15 (validação)	0,82±0,06/0,91±0,03
(Dreijer; Herbst; Du Preez, 2013)	Método semiautomático Alta complexidade computacional	15 (validação)	0,91±0,02/0,93±0,02
(Wang et al., 2015)	Método automático Extração fixa de ROI	7 (4 treinamento, 3 validação)	0,94 (endocárdio)
(Tran, 2016)	Método automático Extração fixa de ROI	30 (15 validação, 15 *online*)	0,92±0,03/0,96±0,01
Método proposto FCN treinada com o RMSProp (iteração 60.000)	**Método automático**	**45 (15 treinamento, 15 validação, 15 *online*)**	**0,91±0,01/0,92±0,02**



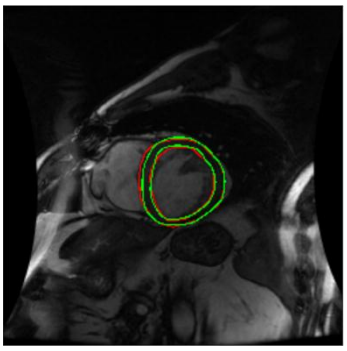
SC-HF-I-04/IM-0004-0100
Dice_endo = 0,9644
Dice_epi = 0,9771



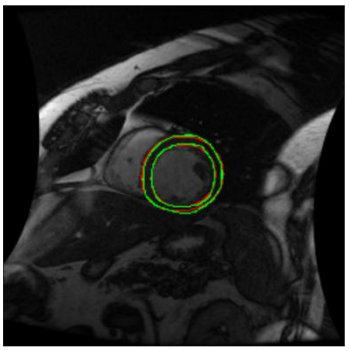
SC-HF-I-05/IM-0156-0060
Dice_endo = 0,9709
Dice_epi = 0,9820



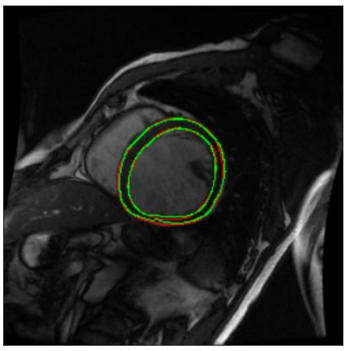
SC-HF-I-06/IM-0180-0059
Dice_endo = 0,9778
Dice_epi = 0,9762



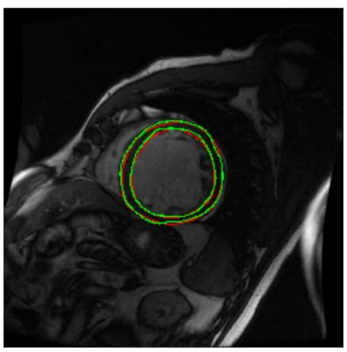
SC-HF-I-07/IM-0209-0020
Dice_endo = 0,9616
Dice_epi = 0,9728



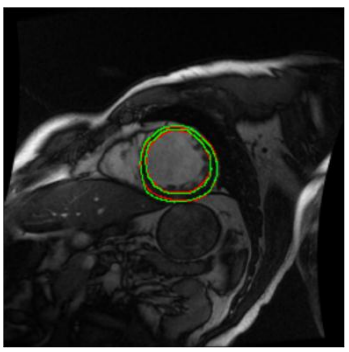
SC-HF-I-08/IM-0226-0080
Dice_endo = 0,9782
Dice_epi = 0,9717



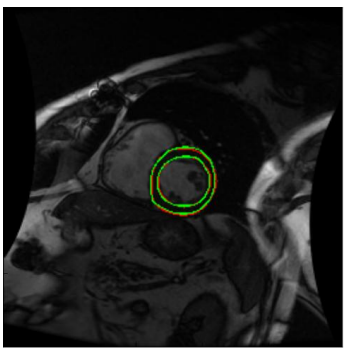
SC-HF-I-10/IM-0024-0060
Dice_endo = 0,9824
Dice_epi = 0,9806



SC-HF-I-10/IM-0024-0100
Dice_endo = 0,9718
Dice_epi = 0,9764

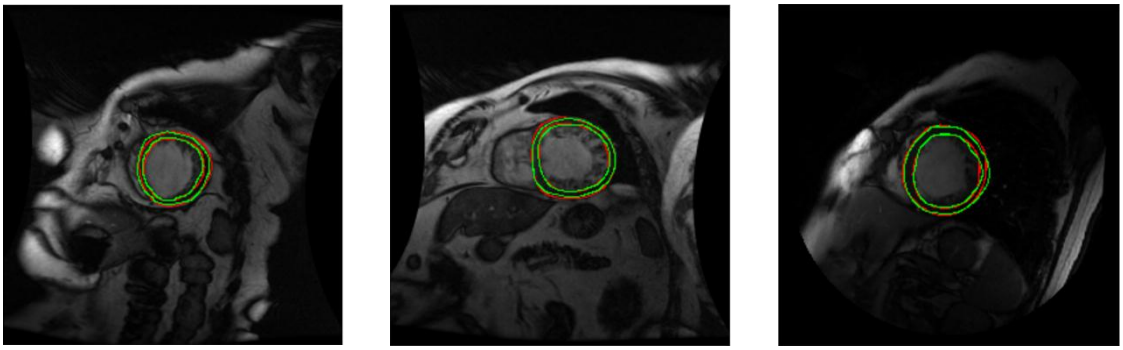


SC-HF-I-11/IM-0043-0120
Dice_endo = 0,9618
Dice_epi = 0,9710



SC-HF-I-40/IM-0134-0080
Dice_endo = 0,9788
Dice_epi =0,9696

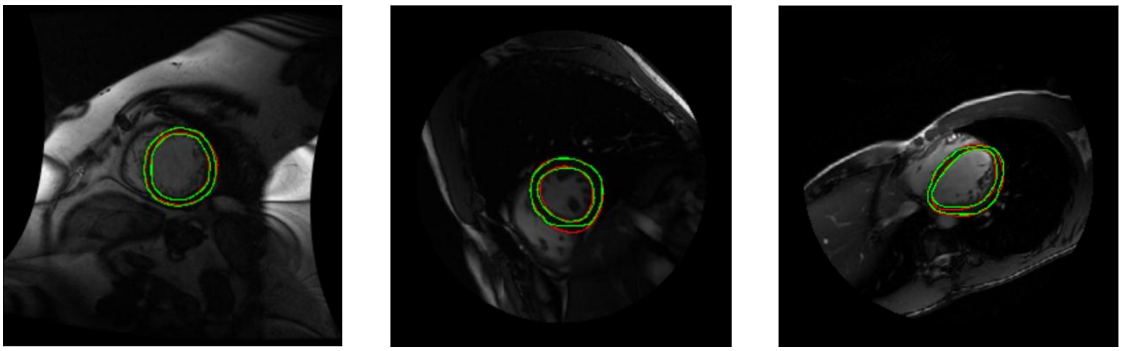
Figura 53. Segmentações com coeficiente de similaridade Dice maior que 0,90 para imagens de pacientes com insuficiência cardíaca e que tiveram infarto (HF-I). Para cada imagem é apresentado o nome da pasta/nome do arquivo, valor de Dice para o endocárdio e para o epicárdio. A linha verde representa os contornos obtidos pela rede e a vermelha os contornos *ground truth*.



SC-HF-NI-03/IM-0379-180
Dice_endo = 0,9622
Dice_epi = 0,9539

SC-HF-NI-04/IM-0501-0081
Dice_endo = 0,9760
Dice_epi = 0,9703

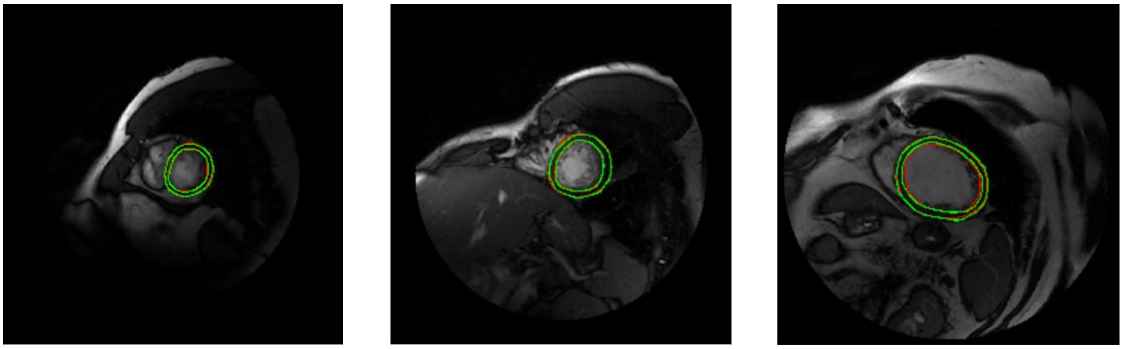
SC-HF-NI-11/IM-0270-0159
Dice_endo = 0,9749
Dice_epi = 0,9672



SC-HF-NI-12/IM-0286-0200
Dice_endo = 0,9789
Dice_epi = 0,9740

SC-HF-NI-14/IM-0331-0136
Dice_endo = 0,9567
Dice_epi = 0,9733

SC-HF-NI-15/IM-0359-0120
Dice_endo = 0,9614
Dice_epi = 0,9710

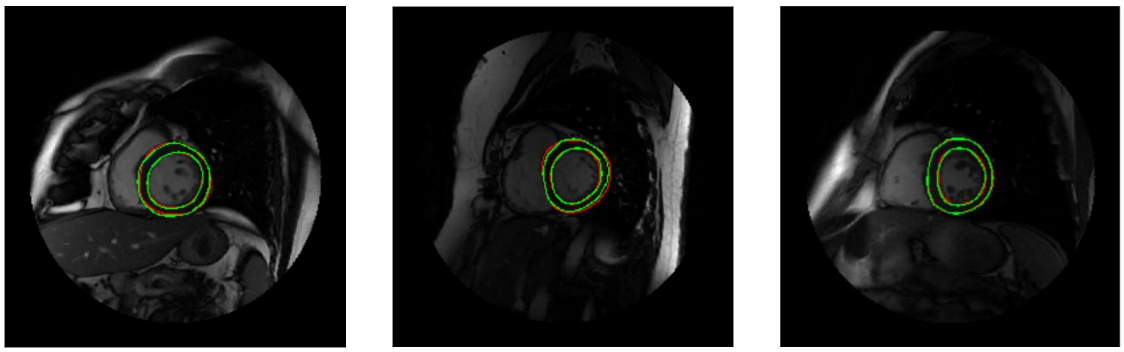


SC-HF-NI-31/IM-0401-0220
Dice_endo = 0,9657
Dice_epi = 0,9727

SC-HF-NI-33/IM-0424-0160
Dice_endo = 0,9782
Dice_epi = 0,9757

SC-HF-NI-36/IM-0474-0200
Dice_endo = 0,9618
Dice_epi = 0,9811

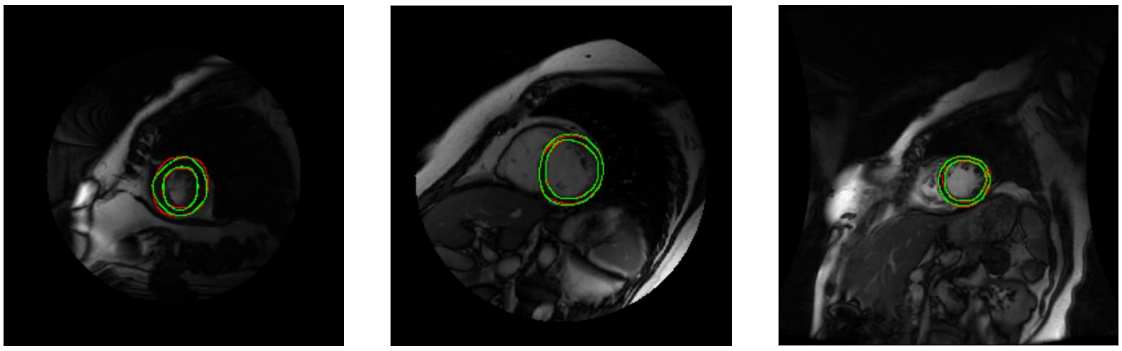
Figura 54. Segmentações com coeficiente de similaridade Dice maior que 0,90 para imagens de pacientes com insuficiência cardíaca sem infarto (HF-NI). Para cada imagem é apresentado o nome da pasta/nome do arquivo, valor de Dice para o endocárdio e para o epicárdio. A linha verde representa os contornos obtidos pela rede e a vermelha os contornos *ground truth*.



SC-HYP-03/IM-0650-0040
Dice_endo = 0,9855
Dice_epi = 0,9703

SC-HYP-06/IM-0767-0060
Dice_endo = 0,9139
Dice_epi = 0,9237

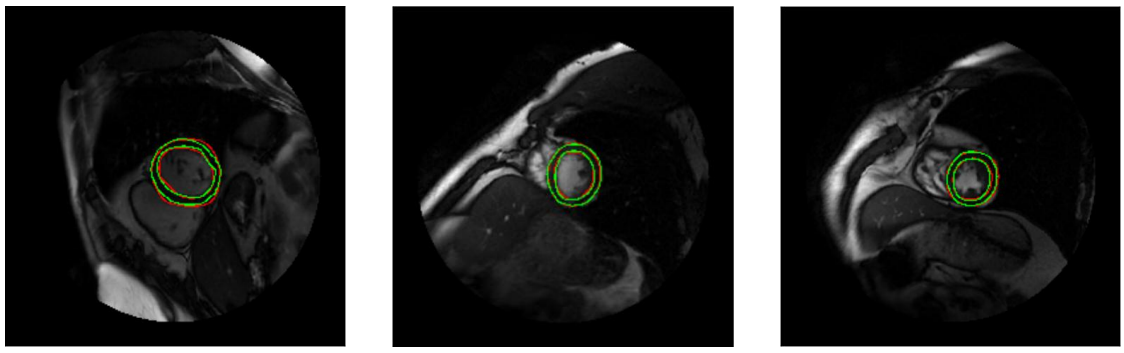
SC-HF-I-06/IM-0180-0059
Dice_endo = 0, 9667
Dice_epi = 0, 9855



SC-HYP-08/IM-0796-0220
Dice_endo = 0,9569
Dice_epi = 0,9403

SC-HYP-09/IM-0003-0120
Dice_endo = 0,9884
Dice_epi = 0,9691

SC-HYP-12/IM-0629-0120
Dice_endo = 0,9473
Dice_epi = 0,9773

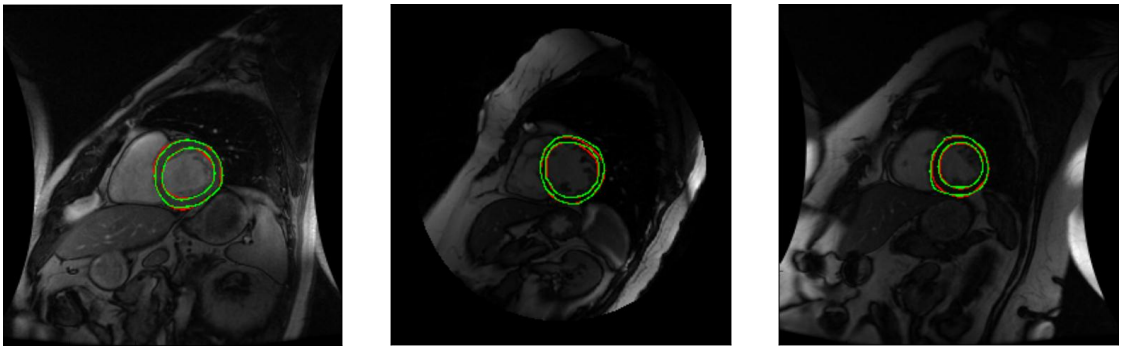


SC-HYP-37/IM-0702-0080
Dice_endo = 0,9515
Dice_epi = 0,9530

SC-HYP-38/IM-0734-0138
Dice_endo = 0,9550
Dice_epi = 0,9775

SC-HYP-40/IM-0755-0159
Dice_endo = 0,9580
Dice_epi = 0,9704

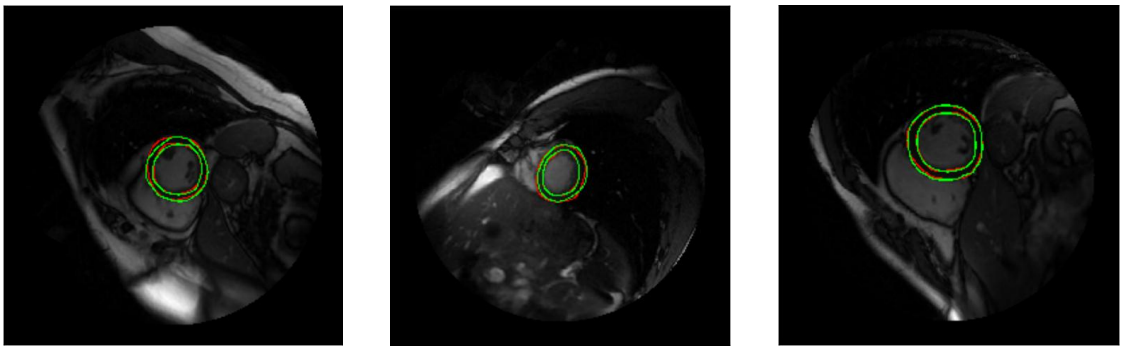
Figura 55. Segmentações com coeficiente de similaridade Dice maior que 0,90 para imagens de pacientes com hipertrofia (HYP). Para cada imagem é apresentado o nome da pasta/nome do arquivo, valor de Dice para o endocárdio e para o epicárdio. A linha verde representa os contornos obtidos pela rede e a vermelha os contornos *ground truth*.



SC-N-02/IM-0898-0040
Dice_endo = 0,9735
Dice_epi = 0,9768

SC-N-03/IM-0915-0260
Dice_endo = 0,9646
Dice_epi = 0,9695

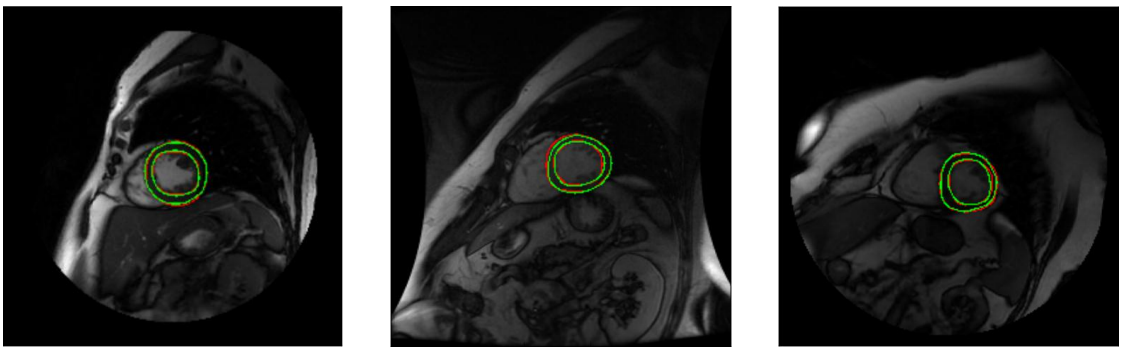
SC-N-05/IM-0963-0080
Dice_endo = 0, 9665
Dice_epi = 0, 9608



SC-N-06/IM-0984-0099
Dice_endo = 0,9620
Dice_epi = 0,9559

SC-N-07/IM-1009-0160
Dice_endo = 0,9800
Dice_epi = 0,9478

SC-N-09/IM-1031-0060
Dice_endo = 0,9734
Dice_epi = 0,9762



SC-N-10/IM-0851-0120
Dice_endo = 0,9648
Dice_epi = 0,9728

SC-N-11/IM-0878-0100
Dice_endo = 0,9390
Dice_epi = 0,9506

SC-N-40/IM-0944-0160
Dice_endo = 0,9603
Dice_epi =0,7070

Figura 56. Segmentações com coeficiente de similaridade Dice maior que 0,90 para imagens de pacientes saudáveis (N). Para cada imagem é apresentado o nome da pasta/nome do arquivo, valor de Dice para o endocárdio e para o epicárdio. A linha verde representa os contornos obtidos pela rede e a vermelha os contornos *ground truth*.

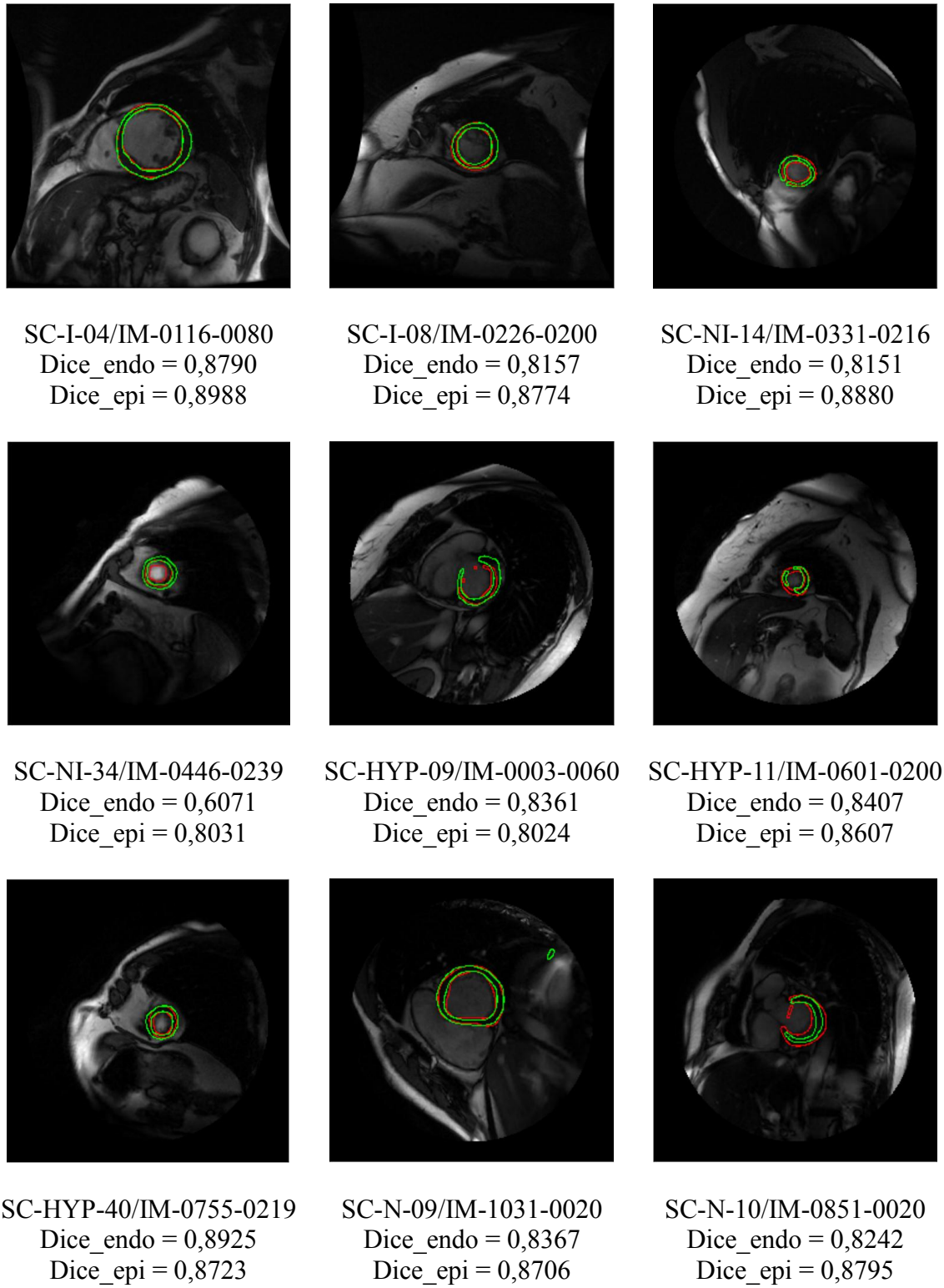
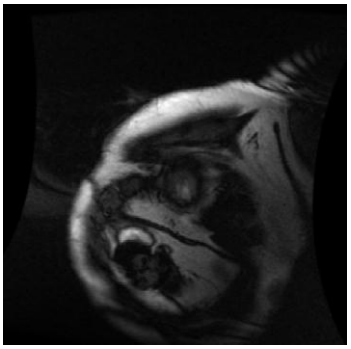
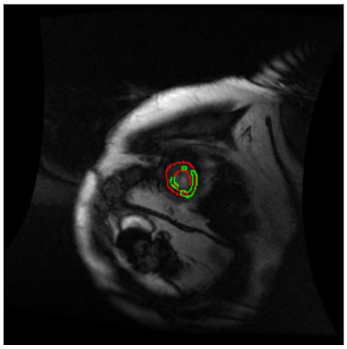


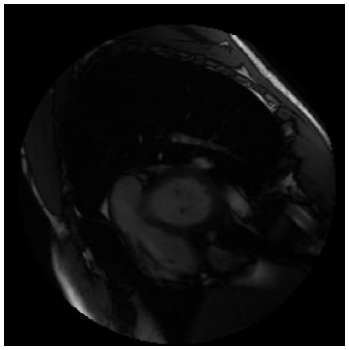
Figura 57. Segmentações com coeficiente de similaridade Dice menor que 0,90. Para cada imagem é apresentado o nome da pasta/nome do arquivo, valor de Dice para o endocárdio e para o epicárdio. A linha verde representa os contornos obtidos pela rede e a vermelha os contornos *ground truth*.



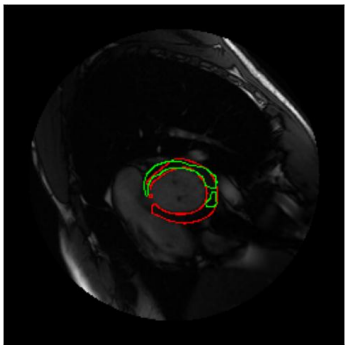
SC-I-09/IM-0241-0278
Imagem original (apex)



Dice_endo = 0,8372
Dice_epi = 0,7297

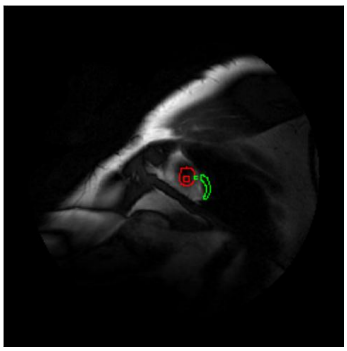


SC-NI-14/IM-0331-0076
Imagem original (base)

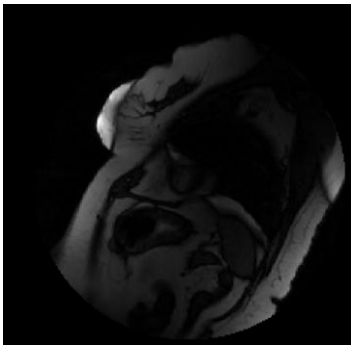


Dice_endo = 0,6938
Dice_epi = 0,6656

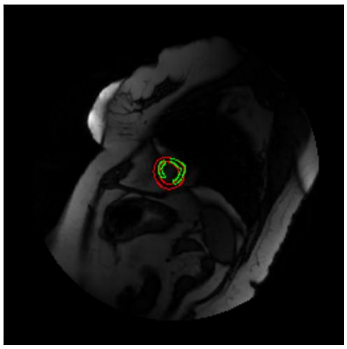
SC-HYP-38/IM-0734-0218
Imagem original (apex)



Dice_endo = 0
Dice_epi = 0,0771



SC-N-03/IM-0915-0360
Imagem original (apex)



Dice_endo = 0,8201
Dice_epi = 0,7385

Figura 58. Alguns casos de imagens em que a rede teve dificuldade para segmentar. A linha verde representa os contornos obtidos pela rede e a vermelha os contornos *ground truth*.

6 CONCLUSÕES

A revisão bibliográfica realizada sobre os métodos de segmentação do miocárdio propostos nos últimos anos permitiu identificar os bancos de imagens de ressonância magnética cardíaca existentes e as métricas de avaliação comumente utilizadas. No que diz respeito ao primeiro, o Sunnybrook é o mais empregado. Em relação às métricas, o coeficiente de similaridade Dice resultou o mais recorrente nos trabalhos revisados. Essa pesquisa também permitiu conhecer o embasamento teórico dos métodos, bem como as fraquezas e desvantagens que apresentam alguns deles, entre as quais estão: a necessidade de intervenção manual para inicializar um determinado algoritmo, a complexidade computacional e a dificuldade na generalização.

Foram estudados os fundamentos teóricos da técnica de imageamento de ressonância magnética e das redes neurais convolutivas. Estas últimas estão alcançando grande sucesso em várias aplicações, e cada vez são mais usadas para o processamento de imagens médicas.

O principal resultado deste trabalho foi a criação de uma arquitetura de rede neural totalmente convolutiva treinada com seis métodos de otimização para executar a segmentação dos contornos do endocárdio e do epicárdio em imagens do eixo curto de ressonância magnética cardíaca. Dentre esses métodos, o gradiente descendente estocástico e o RMSProp foram os de melhores desempenhos. Os resultados numéricos das métricas, em conjunto com a análise qualitativa, evidenciaram a viabilidade do uso do aprendizado profundo para a segmentação das imagens cardíacas.

A metodologia desenvolvida é totalmente automática. Ela demonstrou funcionar muito bem na grande maioria das imagens, especialmente nas pertencentes às faixas

ventriculares médias. Para melhorar a acurácia da segmentação em imagens que exibem bordas ambíguas ou imperceptíveis, como aquelas obtidas a partir dos cortes basal e apical, a comunidade científica deve dedicar esforços na coleta de exemplos anotados nestas localizações, de forma que possam ser usados na etapa de treinamento.

O método apresentado demostrou ser competitivo com o estado da arte. Os coeficientes de similaridade obtidos tanto para o endocárdio quanto para o epicárdio são maiores que em vários trabalhos confrontados. Nos casos onde são menores tem que ser levado em conta fatores como a quantidade de imagens usadas e o nível de interação humana requerido.

Vale a pena ressaltar algumas vantagens do modelo proposto, ganhadas fundamentalmente pela sua natureza de arquitetura profunda: primeira, as imagens de entrada não precisam de nenhum tipo de pré-processamento. Esta etapa, imprescindível nos métodos tradicionais, não é necessária ao trabalhar com as redes neurais convolutivas.

O modelo desenvolvido permite uma grande generalização em relação ao tipo de imagens, ou seja, durante o treinamento, ele aprende a segmentar imagens que possuem variações da iluminação e não homogeneidades do campo de polarização, bem como imagens de indivíduos que apresentam diferentes morfologias nas cavidades ventriculares, próprias de doenças cardíacas ou em casos normais (saudáveis).

A rede implementada pode descobrir diretamente características dos dados de treinamento e, portanto, o esforço do processo de extração de características é eliminado. Os mapas de características criados pelos filtros nas camadas convolutivas podem igualar e inclusive superar o poder discriminativo dos métodos convencionais de extração de características.

Outra vantagem é que a interação e a hierarquia das características podem ser exploradas conjuntamente dentro da própria arquitetura, isto significa que o processo de seleção de características é feito intrinsecamente dentro do treinamento da rede. De modo geral, as etapas de extração de características, seleção e classificação de pixels de forma supervisionada, podem ser realizadas dentro da otimização da arquitetura. De essa forma, o desempenho pode ser aprimorado mais facilmente e de uma forma flexível e sistemática.

O trabalho realizado até aqui marca um ponto de partida para uma série de trabalhos futuros que podem ser feitos a fim de explorar ainda mais as redes neurais totalmente convolutivas e sua aplicação ao processamento de imagens de ressonância magnética cardíaca. Recomenda-se o uso de outros bancos de imagens, a aplicação da estratégia do aumento de dados para o treinamento da rede com o intuito de comprovar sua incidência nos resultados dos testes.

Outra linha de trabalho é o estudo aprofundado do funcionamento dos métodos de otimização. Isso vai permitir ajustar os parâmetros de configuração de forma que melhores resultados podem ser obtidos. Também, a arquitetura da rede pode ser submetida a leves mudanças para estudar como isso influência no seu comportamento.

O potencial demonstrado do modelo proposto é apenas o início. Com maior quantidade de dados para os treinamentos, os modelos FCN podem assumir a liderança na segmentação cardíaca automatizada em aplicações clínicas com velocidade, precisão e confiabilidade.

REFERÊNCIAS

ANDREOPOULOS, A.; TSOTSOS, J. K. Efficient and generalizable statistical models of shape and appearance for analysis of cardiac MRI. Medical Image Analysis, v. 12, n. 3, p. 335–357, 2008.

AVENDI, M. R.; KHERADVAR, A.; JAFARKHANI, H. A combined deep-learning and deformable-model approach to fully automatic segmentation of the left ventricle in cardiac MRI. Medical Image Analysis, v. 30, p. 108–119, 2016.

BERKELEY VISION AND LEARNING CENTER. Caffe. Disponível em: <http://caffe.berkeleyvision.org/>. Acesso em: 17 ago. 2016.

BLINK, E. J. MRI Physics. Disponível em: <http://www.mri-physics.net/>. Acesso em: 16 nov. 2016.

BOYKOV, Y.; FUNKA-LEA, G. Graph Cuts and Efficient N-D Image Segmentation. International Journal of Computer Vision LLC, v. 70, n. 2, p. 109–131, 2006.

BOYKOV, Y.; JOLLY, M.-P. Interactive Organ Segmentation using Graph Cuts. LNCS, p. 276–286, 2000.

CAUCHY, A. Méthode générale pour la résolution de systèmes d'équations simultanées. Compte rendu des séances de l'académie des sciences, p. 536–538, 1847.

CHICCO, D.; SADOWSKI, P.; BALDI, P. Deep autoencoder neural networks for gene ontology annotation predictions. Proceedings of the 5th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics. Anais...Newport Beach, California: ACM Press, 2014

CIRESAN, D. C. et al. Flexible, High Performance Convolutional Neural Networks for Image Classification. Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, v. 2, p. 1237–1242, 2011.

CIREŞAN, D.; MEIER, U.; SCHMIDHUBER, J. Multi-column Deep Neural Networks for Image Classification. CVPR 2012, p. 3642–3649, 2012.

CONSTANTINIDES, C. et al. Fully automated segmentation of the left ventricle applied to cine MR images: Description and results on a database of 45 Subjects. Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS. Anais...2012

DAKUA, S. P.; SAHAMBI, J. S. Weighting function in random walk based left ventricle segmentation. 18th IEEE International Conference on Image Processing. Anais..., 2011

DENG, L. An Overview of Deep-Structured Learning for Information Processing. Proc. Asian-Pacific Signal & Information Proc. Annual Summit & Conference (APSIPA-ASC). Anais...2011

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

DREIJER, J. F.; HERBST, B. M.; DU PREEZ, J. A. Left ventricular segmentation from MRI datasets with edge modelling conditional random fields. BMC Medical Imaging, v. 13, 2013.

DUCHI, J.; HAZAN, E.; SINGER, Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. The Journal of Machine Learning Research, 2011.

FERNÁNDEZ, S.; GRAVES, A.; SCHMIDHUBER, J. An Application of Recurrent Neural Networks to Discriminative Keyword Spotting. International Conference on Artificial Neural Networks. Anais...Springer Berlin Heidelberg, 2007

FLEAGLE, S. R. et al. Automated identification of left ventricular borders from spin-echo magnetic resonance images. Experimental and clinical feasibility studies.

Investigative radiology, v. 26, n. 4, p. 295–303, abr. 1991.

FRANGI, A. F.; NIESSEN, W. J.; VIERGEVER, M. A. Three-Dimensional Modeling for Functional Analysis of Cardiac Images: A Review. IEEE TRANSACTIONS ON MEDICAL IMAGING, v. 20, n. 1, 2001.

FUKUSHIMA, K.; MIYAKE, S. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. Pattern Recognition, v. 15, n. 6, p. 455–469, jan. 1982.

GINAT, D. T. et al. Cardiac imaging: Part 1, MR pulse sequences, imaging planes, and basic anatomy. American Journal of Roentgenology, v. 197, n. 4, p. 808–815, 2011.

GIRSHICK, R. et al. Rich feature hierarchies for accurate object detection and semantic segmentation. ArXiv:1311.2524, 2013.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. 13th International Conference on Artificial Intelligence and Statistics (AISTATS) . Anais...Sardinia, Italy: 2010

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep Learning. Disponível em: <http://www.deeplearningbook.org>. Acesso em: 14 nov. 2016.

HE, K. et al. Deep Residual Learning for Image Recognition. ArXiv:1408.50931512.03385, 2015.

HINTON, G. E. et al. Improving neural networks by preventing co-adaptation of feature detectors. ArXiv:1207.0580v1, 2012.

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. Science, v. 313, n. 5786, p. 504–507, 2006.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. The Journal of physiology, v. 160, n. 1, p. 106–54, jan. 1962.

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:1502.03167, 2015.

JIA, Y. et al. Caffe: Convolutional Architecture for Fast Feature Embedding. ArXiv preprint arXiv:1408.5093, 2014.

KARPATHY, A. Convolutional Neural Networks for Visual Recognition. Disponível em: <http://cs231n.github.io/>. Acesso em: 21 ago. 2016.

KINGMA, D. P.; BA, J. L. Adam: A Method for Stochastic Optimization. International Conference for Learning Representations. Anais...2015

KRASNOBAEV, A.; SOZYKIN, A. An overview of techniques for cardiac left ventricle segmentation on short-axis MRI. ITM Web of Conference, v. 8, n. 01003, 2016.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems, 2012.

LECUN, Y. A. et al. Efficient BackProp. In: Neural Networks: Tricks of the Trade. [s.l.] Springer Berlin Heidelberg, 1998. p. 9–50.

LIU, J. et al. Myocardium Segmentation Combining T2 and De Mri Using Multi-Component Bivariate Gaussian Mixture Model. Isbi, p. 886–889, 2014.

LONG, J.; SHELHAMER, E.; DARRELL, T. Fully Convolutional Networks for Semantic Segmentation. 2015.

M. A. HADHOUD, M. et al. Left Ventricle Segmentation in Cardiac MRI Images. American Journal of Biomedical Engineering, v. 2, n. 3, p. 131–135, 2012.

MARINO, M. et al. Fully automated assessment of left ventricular volumes, function and mass from cardiac MRI. Annual International Conference of IEEE Engineering in Medicine and Biology Society. Anais...2014

MESNIL, G. et al. Using Recurrent Neural Networks for Slot Filling in Spoken Language Understanding. IEEE/ACM Transactions on Audio, Speech, and Language Processing, v. 23, n. 3, p. 530–539, mar. 2015.

MOZAFFARIAN, D. et al. Heart disease and stroke statistics-2015 update : A report from the American Heart Association. Circulation, v. 131, 2015.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. 27th International Conference on Machine Learning. Anais...2010

NESTEROV, Y. A Method of Solving a Convex Programming Problem with Convergence Rate O(1/√2). Soviet Mathematics Doklady, 1983.

NG, A. et al. Deep Learning Tutorial. Disponível em: <http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/>.

NIELSEN, M. Neural Networks and Deep Learning. Disponível em: <http://neuralnetworksanddeeplearning.com/>. Acesso em: 14 nov. 2016.

PENG, P. et al. A review of heart chamber segmentation for structural and functional analysis using cardiac magnetic resonance imagingMagnetic Resonance Materials in Physics, Biology and Medicine, 2016.

PERONA, P.; MALIK, J. Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 12, n. 7, p. 629–639, jul. 1990.

PETITJEAN, C. et al. Right Ventricle Segmentation from cardiac MRI: A collation study. Medical Image Analysis, v. 19, n. 1, p. 187–202, 2015.

PETITJEAN, C.; DACHER, J. N. A review of segmentation methods in short axis

cardiac MR images. Medical Image Analysis, v. 15, n. 2, p. 169–184, 2011.

POLYAK, B. T. Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, v. 4, n. 5, p. 1–17, 1964.

PUDDEPHAT, M. The principles of magnetic resonance imaging. Disponível em: <http://www.voxelcube.com/articles/1/>. Acesso em: 1 jan. 2016.

RADAU, P. et al. Evaluation Framework for Algorithms Segmenting Short Axis Cardiac MRI. MIDAS Journal, 2009.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. Nature, v. 323, n. 6088, p. 533–536, 9 out. 1986.

SAFARZADEH KHOOSHABI, G. Segmentation Validation Framework. Master thesis, Department of Biomedical Engineering, Linköping University, 2013.

SEDERBERG, T. W. et al. Free-form deformation of solid geometric models. ACM SIGGRAPH Computer Graphics, v. 20, n. 4, p. 151–160, 31 ago. 1986.

SHI, W. Department of Computing An image segmentation and registration approach to cardiac function analysis using MRI. n. August, 2012.

SIMONYAN, K.; ZISSERMAN, A. VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION. 2015.

SRIVASTAVA, N. et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, v. 15, p. 1929–1958, 2014.

SUINESIAPUTRA, A. et al. A collaborative resource to build consensus for automated left ventricular segmentation of cardiac MR images. Medical image analysis, v. 18, n. 1, p. 50–62, jan. 2014.

SUTSKEVER, I. et al. On the Importance of Initialization and Momentum in Deep Learning. Proceedings of the 30th International Conference on Machine Learning. Anais...2013

TAHA, A. A.; HANBURY, A. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. BMC Medical Imaging, v. 15, n. 1, p. 29, 2015.

TANG, Y. Deep Learning using Linear Support Vector Machines. ArXiv:1306.0239, 2014.

TRAN, P. V. A Fully Convolutional Neural Network for Cardiac Segmentation in Short-Axis MRI. ArXiv 1604.00494, 2016.

UZUNBAS, M. G. et al. Segmentation of myocardium using deformable regions and graph cuts. Proceedings - International Symposium on Biomedical Imaging, p. 254–257, 2012.

WAN, J. et al. Deep Learning for Content-Based Image Retrieval. Proceedings of the ACM International Conference on Multimedia. Anais...New York: ACM Press, 2014

WANG, L. et al. Automatic Left Ventricle Segmentation in Cardiac MRI via Level Set and Fuzzy C-Means. Proceedings of 2015 RAECS UIET. Anais...Chandigarh: 2015

WU, H.; GU, X. Towards dropout training for convolutional neural networks. Neural Networks, v. 71, p. 1–10, 2015.

WU, K.; LIMA, J. Noninvasive imaging of myocardial viability current techniques and future developments. Circulation research, 2003.

Y. LECUN, L. BOTTOU, Y. BENGIO, P. H. Gradient-based learning applied to document recognition. In: Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

ZEILER, M. ADADELTA: AN ADAPTIVE LEARNING RATE METHOD. ArXiv:1212.5701, 2012.