UNIVERSIDADE FEDERAL DO AMAZONAS

FACULDADE DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

RONALDO DE SÁ PORTELA

SEGMENTAÇÃO DA REGIÃO PULMONAR EM IMAGENS DE RADIOGRAFIA TORÁCICA UTILIZANDO REDES NEURAIS CONVOLUTIVAS

Manaus

2020

RONALDO DE SÁ PORTELA

SEGMENTAÇÃO DA REGIÃO PULMONAR EM IMAGENS DE RADIOGRAFIA TORÁCICA 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 de Controle e Automação de Sistemas.

Orientador: Prof. Dr. Cícero Ferreira Fernandes Costa Filho

Coorientador: Prof. Dr. José Raimundo Gomes Pereira

Manaus

2020

Ficha Catalográfica

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

P843s

Portela, Ronaldo De Sá
Segmentação da região pulmonar em imagens de radiografia torácica utilizando redes neurais convolutivas / Ronaldo De Sá Portela . 2020
124 f.: il. color; 31 cm.

Orientador: Cícero Ferreira Fernandes Costa Filho
Coorientador: José Raimundo Gomes Pereira
Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Amazonas.

1. Segmentação pulmonar. 2. Radiografia torácica. 3. Redes neurais convolutivas. 4. Aprendizado profundo. I. Costa Filho, Cícero Ferreira Fernandes. II. Universidade Federal do Amazonas III. Título

RONALDO DE SÁ PORTELA

SEGMENTAÇÃO DA REGIÃO PULMONAR EM IMAGENS DE RADIOGRAFIA TORÁCICA 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 29 de maio de 2020.

BANCA EXAMINADORA

Prof. Dr. Cícero Ferreira Fernandes Costa Filho, Presidente

Universidade Federal do Amazonas

Prof. Dr. Jozias Parente de Oliveira, Membro

Universidade do Estado do Amazonas

Prof. Dr. José Mir Justino da Costa, Membro

Universidade Federal do Amazonas

AGRADECIMENTOS

Aos meus pais, que desde quando eu era criança, sempre colocaram a educação de todos os filhos como prioridade.

Aos meus orientadores, prof. Dr. Cícero Ferreira Fernandes Costa Filho e prof. Dr. José Raimundo Gomes Pereira, pelas sugestões de melhoria e por todo o conhecimento passado durante esse período.

À prof. Dra. Marly Guimarães Fernandes Costa, pela sua colaboração na elaboração do artigo.

À Universidade Federal do Amazonas, especialmente ao Centro de Tecnologia Eletrônica da Informação – CETELI, pela disponibilidade de toda sua estrutura para o desenvolvimento da pesquisa.

À Samsung, pelo financiamento da pesquisa.

Esta pesquisa, conforme previsto no Art. 48 do decreto n° 6.008/2006, foi financiada pela Samsung Eletrônica da Amazônia Ltda, nos termos da Lei Federal n° 8.387/1991, através de convênio n° 004, firmado com o Centro de P&D em Eletrônica e Tecnologia da Informação da Universidade Federal do Amazonas – CETELI/UFAM.

RESUMO

Segundo dados do Instituto Nacional de Câncer, o câncer de pulmão é um dos tumores mais frequentes na população brasileira. O processo para seu diagnóstico por vezes passa pela necessidade de segmentar a região pulmonar em um exame de imagem, fase essa que demanda horas de um profissional da área médica. Sendo assim, a utilização de ferramentas que aplicam técnicas automatizadas para realizar essa tarefa pode auxiliá-los. Esta dissertação desenvolve uma metodologia automática, baseada em redes neurais convolutivas, para segmentar a região pulmonar em imagens de radiografia torácica. São desenvolvidas três arquiteturas (CNN1, CNN2 e CNN3), onde as arquiteturas CNN1 e CNN2 são de rede direta, enquanto a arquitetura CNN3 é uma topologia de grafos acíclicos direcionados (DAG). Em conjunto com as arquiteturas são investigados três diferentes métodos de regularização (*Dropout*, L2 e *Dropout*+L2) e três diferentes métodos de otimização (SGDM, RMSPROP e ADAM). A base de dados utilizada para esse estudo é a *JSRT - Japanese Society of Radiological Technology*, que contém 247 imagens de radiografia torácica. Como forma de mensurar a performance das redes estudadas foram utilizados seis métricas de desempenho, são elas: Acurácia Global, Acurácia, Coeficiente de Jaccard, Coeficiente de Jaccard Ponderado, Score F1 e Índice Dice. Ao término de todas as simulações, os melhores resultados foram alcançados utilizando a rede CNN3, que faz uso da topologia DAG, conjuntamente com o método de regularização *Dropout*+L2 e método de otimização ADAM. As métricas obtidas foram: Acurácia Global igual a 0.99139 ± 0.00098; Acurácia igual a 0.98927 ± 0.00161; Coeficiente de Jaccard de 0.97967 ± 0.00232; Coeficiente de Jaccard Ponderado igual a 0.98294 ± 0.00191; Score F1 de 0.97475 ± 0.00357 e, por fim, um Índice Dice de 0.98921 ± 0.00163.

Palavras-chave: Segmentação Pulmonar, Radiografia Torácica, Redes Neurais Convolutivas, Aprendizado Profundo.

ABSTRACT

According to data from the National Cancer Institute, lung cancer is one of the most frequent tumors in the Brazilian population. The process for its diagnosis sometimes involves the need to segment the pulmonary region in an image exam, a phase that requires hours from a medical professional. Therefore, the use of tools that apply automated techniques to accomplish this task could help them. This dissertation develops an automatic methodology, based on convolutive neural networks, to segment the lung region in chest X-ray images. Three architectures are developed (CNN1, CNN2 and CNN3), where the CNN1 and CNN2 architectures are of direct network, while the CNN3 architecture is a topology of directed acyclic graphs (DAG). In conjunction with the architectures, three different regularization methods (Dropout, L2 and Dropout + L2) and three different optimization methods (SGDM, RMSPROP and ADAM) are investigated. The database used for this study is the JSRT - Japanese Society of Radiological Technology, which contains 247 images of chest radiography. As a way of measuring the performance of the studied networks, six performance metrics were used, they are: Global Accuracy, Accuracy, Jaccard Coefficient, Weighted Jaccard Coefficient, Score F1 and Dice Index. At the end of all simulations, the best results were achieved using the CNN3 network, which makes use of the DAG topology, together with the Dropout + L2 regularization method and the ADAM optimization method. The metrics obtained were: Global Accuracy equal to 0.99139 ± 0.00098; Accuracy equal to 0.98927 ± 0.00161; Jaccard coefficient of 0.97967 ± 0.00232; Weighted Jaccard coefficient equal to 0.98294 ± 0.00191; F1 Score of 0.97475 ± 0.00357 and, finally, a Dice Index of 0.98921 ± 0.00163.

Keywords: Pulmonary segmentation, chest radiography, convolutive neural networks, deep learning.

LISTA DE FIGURAS

FIGURA 1 EXEMPLO DE RADIOGRAFIA TORÁCICA 10
FIGURA 2 EXEMPLOS DE IMAGENS DE RADIOGRAFIA 12
FIGURA 3 (A) REPRESENTAÇÃO DO NEURÔNIO BIOLÓGICO. (B) REPRESENTAÇÃO DO NEURÔNIO ARTIFICIAL 21
FIGURA 4 FUNÇÃO DE ATIVAÇÃO DEGRAU 22
FIGURA 5 FUNÇÃO DE ATIVAÇÃO DEGRAU BIPOLAR 23
FIGURA 6 FUNÇÃO DE ATIVAÇÃO LOGÍSTICA 23
FIGURA 7 FUNÇÃO DE ATIVAÇÃO TANGENTE HIPERBÓLICA 24
FIGURA 8 EXEMPLO DE REDE *FEEDFORWARD* DE CAMADAS MÚLTIPLAS 25
FIGURA 9 NOTAÇÃO DAS VARIÁVEIS PARA ELABORAÇÃO DO ALGORITMO *BACKPROPAGATION* 27
FIGURA 10 ARQUITETURA DA LENET-5, UMA REDE NEURAL CONVOLUTIVA USADA PARA RECONHECIMENTO DE DÍGITOS 30
FIGURA 11 (A) SINAL A SER CONVOLUÍDO COM O *KERNEL*. (B) *KERNEL* 31
FIGURA 12 (A) PRIMEIRO PASSO DA CONVOLUÇÃO. (B) SEGUNDO PASSO DA CONVOLUÇÃO 32
FIGURA 13 RESULTADO DA CONVOLUÇÃO 32
FIGURA 14 CONVOLUÇÃO UTILIZANDO FUNÇÕES 2-D 33
FIGURA 15 (A) ESTRUTURA DE UMA CNN COM CONECTIVIDADE ESPARSA. (B) ESTRUTURA DE UMA RNN TRADICIONAL COMPLETAMENTE CONECTADA 34
FIGURA 16 CONVOLUÇÃO MOSTRADA COMO MULTIPLICAÇÃO DE MATRIZES 34
FIGURA 17 FUNÇÃO DE ATIVAÇÃO *RELU* E SUA DERIVADA 36
FIGURA 18 OPERAÇÃO *MAX-POOLING* 37
FIGURA 19 OPERAÇÃO *MAX-POOLING* UTILIZANDO EXEMPLO NUMÉRICO 37
FIGURA 20 OPERAÇÃO DE DECONVOLUÇÃO OU SOBRE-AMOSTRAGEM 38
FIGURA 21 FUNÇÃO *SOFTMAX* 39
FIGURA 22 ILUSTRAÇÃO DA AÇÃO DA CAMADA *DROPOUT*. (A) REDE E SUAS CONEXÕES. (B) RESULTADO DA APLICAÇÃO DO *DROPOUT* 40
FIGURA 23 COMPORTAMENTO DA FUNÇÃO SIGMOIDE EM REDES COM UMA, DUAS, TRÊS E QUATRO CAMADAS 41
FIGURA 24 FUNÇÃO DE PERDA (*LOSS FUNCTION*) DE UMA REDE 42
FIGURA 25 TRAJETÓRIA PERCORRIDA PELOS PESOS SINÁPTICOS DESDE SUA INICIALIZAÇÃO ALEATÓRIA ATÉ ALCANÇAR O MÍNIMO DA FUNÇÃO DE PERDA 43
FIGURA 26 TRAJETÓRIA PERCORRIDA PELOS PESOS SINÁPTICOS UTILIZANDO UMA TAXA DE APRENDIZADO GRANDE 43
FIGURA 27 ILUSTRAÇÃO DE UM PONTO DE MÍNIMO LOCAL, GLOBAL E UM PONTO DE SELA 44
FIGURA 28 EXEMPLO DE UMA CNN UTILIZADA PARA REALIZAR SEGMENTAÇÃO SEMÂNTICA 48
FIGURA 29 DIAGRAMA DE BLOCOS DA METODOLOGIA UTILIZADA PARA REALIZAÇÃO DA TAREFA DE SEGMENTAÇÃO DA REGIÃO DO PULMÃO 49
FIGURA 30 EXEMPLOS DE RADIOGRAFIAS TORÁCICAS DA BASE DE DADOS. (A) IMAGEM COM UM NÓDULO (B) IMAGEM SEM NÓDULO 50

FIGURA 31 EXEMPLO DE PADRÃO OURO DAS IMAGENS DA FIGURA ANTERIOR. (A) PADRÃO OURO DA FIGURA 30(A). (B) PADRÃO OURO DA FIGURA 30(B) 50
FIGURA 32 ILUSTRAÇÃO DO PROCEDIMENTO DE REDIMENSIONAMENTO DAS IMAGENS DE ENTRADA DA REDE. (A) IMAGEM 2048X2048. (B) IMAGEM APÓS O REDIMENSIONAMENTO, EM TAMANHO 512X512 51
FIGURA 33 ILUSTRAÇÃO DO PROCEDIMENTO DE SOMA E REDIMENSIONAMENTO DAS IMAGENS PADRÃO OURO. (A) IMAGEM CONTENDO O PULMÃO ESQUERDO E O DIREITO, SEPARADAMENTE. (B) IMAGEM RESULTANTE DA SOMA DAS DUAS IMAGENS, EM TAMANHO 2048X2048. (C) IMAGEM RESULTANTE DO REDIMENSIONAMENTO, EM TAMANHO 512X512 51
FIGURA 34 DIVISÃO DA BASE DE DADOS EM TREINAMENTO, VALIDAÇÃO E TESTE 52
FIGURA 35 PROCEDIMENTO PARA DIVISÃO DA BASE DE DADOS EM TREINAMENTO, VALIDAÇÃO E TESTE 52
FIGURA 36 METODOLOGIA PARA A ESCOLHA DA MELHOR ARQUITETURA, MELHOR MÉTODO DE REGULARIZAÇÃO E MELHOR TÉCNICA DE TREINAMENTO PARA REALIZAÇÃO DA TAREFA DE SEGMENTAÇÃO PULMONAR. A ESCOLHA SERÁ FEITA ATRAVÉS DO DESEMPENHO NO CONJUNTO DE VALIDAÇÃO. 53
FIGURA 37 PROCESSO DE VALIDAÇÃO CRUZADA DA REDE 54
FIGURA 38 PRIMEIRA ARQUITETURA (CNN-1) PROPOSTA – REDE DIRETA 55
FIGURA 39 SEGUNDA ARQUITETURA (CNN-2) PROPOSTA – REDE DIRETA 56
FIGURA 40 TERCEIRA ARQUITETURA (CNN-3) PROPOSTA – GRAFO ACÍCLICO DIRETO 57
FIGURA 41 DIAGRAMA DE VENN 59
FIGURA 42 EXEMPLO DE UMA MATRIZ DE CONFUSÃO 61
FIGURA 43 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 63
FIGURA 44 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 64
FIGURA 45 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 65
FIGURA 46 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 65
FIGURA 47 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 66
FIGURA 48 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 67
FIGURA 49 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 67
FIGURA 50 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 68
FIGURA 51 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 69
FIGURA 52 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM . 69
FIGURA 53 MATRIZ DE CONFUSÃO PERDA PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 70
FIGURA 54 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 71

FIGURA 55 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 71
FIGURA 56 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 72
FIGURA 57 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 73
FIGURA 58 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 73
FIGURA 59 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 74
FIGURA 60 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 75
FIGURA 61 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 75
FIGURA 62 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 76
FIGURA 63 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 77
FIGURA 64 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 77
FIGURA 65 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 78
FIGURA 66 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 79
FIGURA 67 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN1, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 79
FIGURA 68 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 80
FIGURA 69 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 81
FIGURA 70 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 81
FIGURA 71 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 82
FIGURA 72 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 83
FIGURA 73 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 83
FIGURA 74 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 84
FIGURA 75 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 85

FIGURA 76 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 85
FIGURA 77 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 86
FIGURA 78 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 87
FIGURA 79 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM . 87
FIGURA 80 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 88
FIGURA 81 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 89
FIGURA 82 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 89
FIGURA 83 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 90
FIGURA 84 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 91
FIGURA 85 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 91
FIGURA 86 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 92
FIGURA 87 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 93
FIGURA 88 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 93
FIGURA 89 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 94
FIGURA 90 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 95
FIGURA 91 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 95
FIGURA 92 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 96
FIGURA 93 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 97
FIGURA 94 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN2, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 97
FIGURA 95 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 98
FIGURA 96 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 99
FIGURA 97 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 100

FIGURA 98 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 101
FIGURA 99 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM 101
FIGURA 100 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 102
FIGURA 101 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 103
FIGURA 102 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP 103
FIGURA 103 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 104
FIGURA 104 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 105
FIGURA 105 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM 105
FIGURA 106 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 106
FIGURA 107 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 107
FIGURA 108 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM 107
FIGURA 109 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 108
FIGURA 110 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 109
FIGURA 111 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP 109
FIGURA 112 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 110
FIGURA 113 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 111
FIGURA 114 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM 111
FIGURA 115 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 112
FIGURA 116 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 113
FIGURA 117 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM 113
FIGURA 118 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 114

FIGURA 119 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 115
FIGURA 120 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP 116
FIGURA 121 GRÁFICOS DA ACURÁCIA E FUNÇÃO DE PERDA PARA ARQUITETURA CNN3, DURANTE UMA SEÇÃO DE TREINAMENTO, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 116
FIGURA 122 MATRIZ DE CONFUSÃO PARA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 117
FIGURA 123 SEGMENTAÇÃO REALIZADA PELA ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM 118
FIGURA 124 SEGMENTAÇÃO REALIZADA PELA REDE DAG, ONDE A LINHA FECHADA EM PRETO REPRESENTA A BORDA DA IMAGEM PADRÃO OURO 120

LISTA DE TABELAS

TABELA 1 SÍNTESE DA REVISÃO BIBLIOGRÁFICA DE SEGMENTAÇÃO EM IMAGENS MÉDICAS 17
TABELA 2 PARÂMETROS UTILIZADOS NA ETAPA DE TREINAMENTO DA REDE 58
TABELA 3 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 64
TABELA 4 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 64
TABELA 5 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO 66
TABELA 6 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO 66
TABELA 7 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO 68
TABELA 8 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E 68
TABELA 9 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO 70
TABELA 10 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO 70
TABELA 11 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO 72
TABELA 12 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E 72
TABELA 13 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO 74
TABELA 14 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO L2 E 74
TABELA 15 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO 76
TABELA 16 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO 76
TABELA 17 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO ... 78

TABELA 18 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO 78
TABELA 19 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO 80
TABELA 20 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN1, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO 80
TABELA 21 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 82
TABELA 22 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 82
TABELA 23 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 84
TABELA 24 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 84
TABELA 25 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 86
TABELA 26 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 86
TABELA 27 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 88
TABELA 28 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 88
TABELA 29 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 90
TABELA 30 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 90
TABELA 31 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 92
TABELA 32 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 92
TABELA 33 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 94
TABELA 34 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 94
TABELA 35 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO.... 96
TABELA 36 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 96
TABELA 37 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 98
TABELA 38 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN2, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 98

TABELA 39 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 100
TABELA 40 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 100
TABELA 41 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 102
TABELA 42 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 102
TABELA 43 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 104
TABELA 44 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT* E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 104
TABELA 45 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 106
TABELA 46 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 106
TABELA 47 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 108
TABELA 48 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 108
TABELA 49 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 110
TABELA 50 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 110
TABELA 51 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 112
TABELA 52 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+ L2 E OTIMIZAÇÃO SGDM, NA ETAPA DE VALIDAÇÃO. 112
TABELA 53 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 114
TABELA 54 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+ L2 E OTIMIZAÇÃO RMSPROP, NA ETAPA DE VALIDAÇÃO. 115
TABELA 55 MÉTRICAS DE DESEMPENHO OBTIDAS COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 117
TABELA 56 MÉTRICAS DE DESEMPENHO OBTIDAS POR CLASSE COM A ARQUITETURA CNN3, UTILIZANDO REGULARIZAÇÃO *DROPOUT*+ L2 E OTIMIZAÇÃO ADAM, NA ETAPA DE VALIDAÇÃO. 117
TABELA 57 RESULTADOS OBTIDOS COM O TREINAMENTO DE ARQUITETURAS USANDO O CONJUNTO DE TREINAMENTO COM O DESEMPENHO AVALIADO USANDO O CONJUNTO DE VALIDAÇÃO 118
TABELA 58 COMPARAÇÃO DOS RESULTADOS OBTIDOS USANDO VALIDAÇÃO CRUZADA (MÉDIA ± DESVIO PADRÃO) COM *BENCHMARK* 119

LISTA DE ABREVIATURAS E SIGLAS

AAM – *Active Appearance Models*

ADAM – Estimação de Momento Adaptativo

ASM – *Active Shape Model*

CAD – *Computer Aided Diagnostic*

CNN – Redes Neurais Convolutivas

DAC – Diagnóstico Assistido por Computador

DAG – Grafos Acíclicos Direcionados

GPU – Unidade de Processamento Gráfico

HHS – *Hamilton Health Sciences*

INCA – Instituto Nacional de Câncer

IVOCT – *Intravascular Optical Coherence Tomography*

JSRT – *Japanese Society of Radiological Technology*

LIDC – *Lung Image Database Consortium*

OMS – Organização Mundial da Saúde

PC – *Pixel Classification*

PDI – Processamento Digital de Imagens

ReLu – Unidade Retificadora Linear

RMSProp – Propagação de raiz quadrada média

RNA – Redes Neurais Artificiais

ROI – Região de Interesse

SGD – *Stochastic Gradient Descent*

SGDM – *Stochastic Gradient Descent with Momentum*

SLICO – *Simple Linear Iterative Clustering*

SUMÁRIO

1 INTRODUÇÃO 10
1.1 OBJETIVO GERAL 11
1.2 OBJETIVOS ESPECÍFICOS 11

2 REVISÃO BIBLIOGRÁFICA 12
2.1 TÉCNICAS DE SEGMENTAÇÃO EM IMAGENS MÉDICAS 13

3 REFERENCIAL TEÓRICO 19
3.1 REDES NEURAIS ARTIFICIAIS 19
Neurônio biológico e 20
3.1.1 Neurônio artificial 20
3.1.2 Funções de ativação 22
3.1.3 Arquiteturas e processos de treinamento 24
3.1.4 Processo de treinamento da *feedforward* de múltiplas camadas 27
3.2 REDES CONVOLUTIVAS 29
3.2.1 Camada de Convolução 30
3.2.2 Camada *Batch Normalization* 35
3.2.3 Camada de Unidades Lineares Retificadoras 36
3.2.4 Camada de *Pooling* (ou sub-amostragem) 36
3.2.5 Camada de Convolução Transposta (ou sobre-amostragem) 37
3.2.6 Camada de Classificação 38
3.2.7 Camada *Dropout* 39
3.3 PROCESSO DE TREINAMENTO DE UMA REDE CONVOLUTIVA 40
3.3.1 Métodos de Otimização utilizados no treinamento 42
3.3.2 Métodos de Regularização 46
3.4 SEGMENTAÇÃO SEMÂNTICA 48

4 METODOLOGIA 49
4.1 CONFIGURAÇÃO DO AMBIENTE DE TRABALHO 49
4.2 PRÉ-PROCESSAMENTO DA BASE DE DADOS 49
4.3 PROPOSTA DAS ARQUITETURAS DE REDES CONVOLUTIVAS 52
4.4 EXPERIMENTOS 57
4.5 MÉTRICAS DE DESEMPENHO DAS REDES 58

5 RESULTADOS OBTIDOS E DISCUSSÕES 63
5.1 SIMULAÇÕES REALIZADAS COM A REDE CONVOLUTIVA CNN1 63
5.2 SIMULAÇÕES REALIZADAS COM A REDE CONVOLUTIVA CNN2 81
5.3 SIMULAÇÕES REALIZADAS COM A REDE CONVOLUTIVA CNN3 99
5.4 COMPARAÇÃO COM O BENCHMARK 118

6 CONCLUSÕES 121

REFERÊNCIAS BIBLIOGRÁFICAS 122

1 INTRODUÇÃO

De acordo com a Organização Mundial da Saúde ("OMS," 2018) o câncer é a segunda maior causa de mortes no mundo, sendo que, por ano, 8,8 milhões de pacientes morrem. A maioria das mortes ocorrem em famílias de baixa renda e quase todas as famílias são afetadas pelo câncer de alguma maneira. Segundo dados do Instituto Nacional de Câncer (INCA, 2018) , no Brasil, estimam-se 18.740 casos novos de câncer de pulmão entre homens (o segundo tumor mais frequente) e de 12.530 nas mulheres (quarto tumor mais frequente) para cada ano do biênio 2018-2019.

Para o diagnóstico e tratamento das doenças pulmonares, exames como radiografia e tomografia computadorizada auxiliam o médico na hora de uma tomada de decisão.

Para a formação de uma imagem de radiografia (conhecida também como imagem de Raios-x), um tubo emite um feixe de radiação controlado. Ao passar pelas diferentes estruturas do corpo, elas irão absorver ou bloquear a maior parte dessa radiação. A Figura 1 mostra um exemplo de radiografia torácica, utilizada entre outras coisas para o diagnóstico de câncer de pulmão.

Figura 1 Exemplo de radiografia torácica

Atualmente, para auxiliar o profissional no diagnóstico preciso, existem métodos baseados em software, batizados por Diagnóstico Assistido por Computador (DAC), do inglês *Computer Aided Diagnostic (CAD)*. Esses métodos realizam a detecção de anomalias em exames de forma automatizada, suprindo insumos para o médico realizar diagnósticos mais rápidos e precisos.

O processo para o diagnóstico de câncer pulmonar pode passar pela necessidade de segmentação do tecido pulmonar na imagem de radiografia, processo esse que pode consumir

horas do profissional, principalmente em serviços que fazem centenas de exames de radiografia por dia. Dessa forma, a utilização de ferramentas que aplicam técnicas para segmentação automática pode ajudá-los bastante no diagnóstico mais preciso. Nos últimos anos, as Redes Neurais Convolutivas (CNN) vêm ganhando cada vez mais destaque em diversas aplicações relacionadas as imagens na área médica, como segmentação de estruturas anatômicas e detecção de anomalias. Esse trabalho visa contribuir com a avaliação da aplicação de CNN para a tarefa de segmentação da região pulmonar em imagens de radiografia torácica. O estudo foi feito utilizando a base de dados *Japanese Society of Radiological Technology (JSRT)*, que contém um total de 247 imagens de radiografias torácica. A base de dados encontra-se mais bem detalhada na seção 4.2.

1.1 OBJETIVO GERAL

Propor e avaliar diferentes arquiteturas de Redes Neurais Convolutivas para a segmentação automática da região pulmonar em imagens de radiografia torácica.

1.2 OBJETIVOS ESPECÍFICOS

1. Propor e avaliar o desempenho das Redes Neurais Convolutivas com Arquitetura Serial e Grafos Acíclicos Direcionados (DAG) na tarefa de segmentação pulmonar em imagens de radiografia torácica;
2. Avaliar se diferentes métodos de otimização podem contribuir com o desempenho das redes convolutivas utilizando o método de propagação reversa propostas;
3. Avaliar se diferentes métodos de regularização associados aos métodos de otimização podem contribuir com o desempenho das redes convolutivas utilizando o método de propagação reversa propostas;
4. Contribuir para o estado da arte da segmentação de pulmão em imagens de radiografia torácica;
5. Realizar o *benchmark* das arquiteturas propostas com os métodos previamente publicados na literatura com a mesma base de dados.

2 REVISÃO BIBLIOGRÁFICA

O exame de radiografia é um procedimento de imagem para avaliar o corpo humano, que cria uma imagem das estruturas internas do corpo, utilizando uma pequena quantidade de radiação. Dentre outras aplicações, a radiografia torácica pode ser usada para detectar a presença de tumor em algum dos pulmões. É um exame rápido, fácil de ser realizado e mais barato do que outros exames de imagem.

Para a obtenção da imagem, um tubo especial de raios-x emite um feixe de radiação controlado. Os tecidos do corpo absorvem ou bloqueiam a radiação em diferentes graus. O tecido denso como do osso bloqueia a maior parte da radiação, mas os tecidos moles, como a gordura ou músculo, bloqueiam menos radiação. Após atravessar o corpo, o feixe incide sobre uma placa com um filme, onde projeta uma espécie de sombra. Os tecidos que bloqueiam quantidades elevadas de radiação, como ossos, aparecem como áreas brancas. Os tecidos moles, que bloqueiam menos radiação, aparecem em tons de cinza e os órgãos que são principalmente preenchidos com ar, como os pulmões, aparecem normalmente na cor preta. Os tumores são geralmente mais densos do que o tecido circundante, de modo que muitas vezes podem ser vistos como tons mais claros de cinza. A Figura 2 mostra o resultado de imagens de radiografia de várias regiões do corpo.

Figura 2 Exemplos de imagens de radiografia

Em uma imagem de radiografia torácica, para se realizar o diagnóstico de câncer de pulmão, para melhorar a visualização do médico e auxiliá-lo na tomada de decisão, é interessante realizar-se a segmentação da estrutura pulmonar na imagem. Fazer isso de forma manual, no entanto, demanda muito tempo e pode-se incorrer em erros. Desse modo, a segmentação automática da região pulmonar é algo desejável.

Em meio aos trabalhos publicados na literatura com o objetivo de segmentação da região pulmonar em imagens de radiografia do tórax, as mais diversas técnicas são utilizadas para realizar tal tarefa, desde técnicas de baixa complexidade, como aplicação de um limiar e métodos morfológicos, até técnicas mais complexas, como técnicas de *Deep Learning* usando Redes Neurais Convolutivas. A seguir serão mostradas as principais técnicas descritas na literatura referente a segmentação em imagens médicas.

2.1 TÉCNICAS DE SEGMENTAÇÃO EM IMAGENS MÉDICAS

Neste artigo, (Ginneken, Stegmann, & Loog, 2005) apresentam uma abordagem para realizar a segmentação das cinco estruturas anatômicas que aparecem em uma imagem de radiografia pulmonar: pulmão esquerdo e direito, clavícula esquerda e direita e o coração. Para a realização dessa tarefa os autores fazem uso de 3 técnicas. A primeira técnica utilizada é a *Active Shape Model Segmentartion* (ASM), em que um conjunto de objetos em uma imagem de treinamento é descrito por n pontos correspondentes. Esses pontos são postos em um vetor. Após essa etapa, realiza-se um processo de minimização de distância entre os pontos. A segunda técnica utilizada pelos autores é a *Active Appearance Models* (AAM), que se diferencia do método anterior por fazer uso de todos os pixels da imagem. Por fim, os autores fazem uso da técnica de *Pixel Classification* (PC), em que o treinamento é realizado através de um conjunto de características extraídas das imagens de entrada, como por exemplo, níveis de cinza nos arredores, saídas de filtros etc. O método foi aplicado na base de dados *Japanese Society of Radiological Technology (JSRT)*, que contém 247 imagens de radiografias do tórax. A métrica utilizada pelos autores para avaliação das segmentações foi o coeficiente de Jaccard. Para a segmentação do pulmão, os resultados obtidos utilizando cada uma das técnicas foram: ASM: 0,903±0,057, AAM: 0,847±0,095 e PC: 0,938±0,027. Para a segmentação do coração, os resultados foram: ASM: 0,793±0,119, AAM: 0,775±0,135 e PC: 0,811±0,077. Para a segmentação das clavículas: ASM: 0,690±0,143, AAM: 0,505±0,234 e PC: 0,618±0,100.

Para a tarefa de segmentação de um nódulo pulmonar, (Cavalcanti et al., 2016) apresentam dois métodos. Os autores fazem uso de várias fatias (*slices*) obtidas em uma tomografia computadorizada e as trata como se fossem *frames* de um vídeo. Para isso, faz-se uso de técnicas de estimativas de movimento originalmente propostas para vídeos, como o método de Lucas-Kanade e o método SubME. Na metodologia proposta trabalha-se com a Região de Interesse (ROI) da imagem de entrada e com a ROI do *slice* mais próximo que não possui o nódulo, para se fazer estimação de *background* das imagens (O autor considera como

background tudo aquilo que não faz parte do nódulo). Em seguida, aplica-se o limiar de Otsu, obtendo-se assim uma imagem binarizada, porém possivelmente com mais de uma região conectada. Posteriormente, para eliminar as regiões conectadas que não são de interesse, ou seja, não fazem parte do nódulo, operações morfológicas são aplicadas à imagem resultante, eliminando-se objetos pequenos e mantendo apenas aquele mais próximo da região central. Para a realização desse trabalho os autores utilizaram duas bases de dados: *Hamilton Health Sciences (HHS)* e *The Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI)*. Os resultados obtidos foram os seguintes: usando SubME, 93,53% para TPR (*true positive rate*), 0,89% para FPR (*false positive rate*) e 99,11% de acurácia. Usando Lucas-kanade, 95,66% para TPR, 0,98% para FPR e 99,02% de acurácia.

No trabalho de (Chondro, Yao, Ruan, & Chien, 2018), os autores apresentam uma abordagem baseada em técnicas de Processamento Digital de Imagens (PDI) para segmentação da região pulmonar em imagens de radiografia torácica. Primeiramente, uma equalização do histograma é aplicada na imagem, com o objetivo de realçar o contraste entre os pulmões e outras regiões adjacentes. Após isso, é realizada a extração do primeiro plano da imagem, através do método de binarização inteligente baseada em blocos, seguida da técnica de erosão morfológica. Em seguida, o autor faz uso de um filtro Gaussiano seguido de operações morfológicas para se realizar a segmentação baseada em regiões estatísticas. Por fim, é realizado o refinamento dos limites da área desejada. O método foi aplicado utilizando duas bases de dados: JSRT e Montgomery. Os seguintes resultados foram obtidos: 1) base de dados JSRT: Para medir a precisão do método, foram utilizadas as seguintes métricas: Coeficiente de Jaccard 0,963±0,012, Índice Dice 0,983±0,007 e distância média de contorno 0,341±2,154; 2) base de dados Montgomery: Coeficiente de Jaccard 0,966±0,018, Índice Dice 0,978±0,005 e distância média de contorno 0,394±0,623.

Os autores (Uzelaltinbulat & Ugur, 2017) apresentam uma abordagem para a tarefa de segmentação de um tumor pulmonar. A implementação do método compreende diferentes estágios. O primeiro passo consiste em um pré-processamento da imagem, realizado através de erosão morfológica, seguida de filtragem mediana, para suavização e aprimoramento. Após isso, aplica-se o limiar de Otsu, obtendo-se assim uma imagem binária, resultando em uma imagem contendo apenas o tumor e as clavículas. No passo seguinte aplica-se operação morfológica para eliminação de objetos pequenos. Por fim, realiza-se a subtração entre a imagem resultante de Otsu e a imagem resultante do processo morfológico. Como resultado dessa subtração restará apenas o tumor na imagem. O algoritmo foi testado na base de dados

LIDC, que contém 70 imagens, sendo 50 com tumor e 20 sem tumor. Os resultados obtidos foram: Acurácia de 97,14%, Sensibilidade de 96% e Especificidade de 100%.

Utilizando redes neurais convolutivas (CNN), (Hwang & Park, 2017) propuseram um método para realizar a segmentação da região pulmonar em imagens de radiografia torácica. O modelo proposto baseia-se em camadas convolutivas dilatadas (Atrous), de forma a aumentar o campo de visão de forma eficiente. A rede também faz uso do método de interpolação bilinear. A arquitetura proposta para a realização da tarefa foi a *Deep-and-Thin*. Uma das vantagens dessa arquitetura é que possui muito menos parâmetros em comparação com outros modelos de segmentação pulmonar baseados em CNN. O modelo proposto possui 120.672 pesos, enquanto uma rede com arquitetura U-Net possui 3.140.771 pesos. Os autores utilizaram a base de dados JSRT. Esta base contém 247 imagens de radiografia torácica, sendo que 154 imagens possuem 1 nódulo pulmonar, enquanto outras 93 imagens não possuem nódulo algum. Essas imagens são divididas em duas partes, *fold1* e *fold2*, de forma a cada parte ficar com o mesmo número de imagens com e sem nódulo. Como metodologia, o treinamento da rede foi realizado usando os dados do *fold1* e o teste foi realizado usando os dados do *fold2*, e vice-versa. Como forma de testar sua capacidade de generalização em outro tipo de imagens, os autores realizaram também treinamento na base de dados *Montgomery County*. As métricas utilizadas nesse trabalho para avaliar o desempenho da rede foram: Coeficiente de similaridade de Jaccard, 0,961±0,015; Coeficiente de dados, 0,980±0,008; Distância média de contorno, 1,237±0,702 e Distância média de superfície, 0,675±0,122.

Os autores (Yassine, Taylor, & Story, 2018) em seu trabalho se dedicaram a realizar a segmentação da região pulmonar em imagens radiográficas utilizando a técnica dos Superpixels. Essa técnica é similar ao algoritmo k-means. Porém, enquanto o algoritmo k-means leva em consideração apenas a distância entre os pixels para agrupá-los, a metodologia *Simple Linear Iterative Clustering* (SLICO) Superpixels leva em conta também a similaridade entre as cores dos pixels, para que assim se agrupe pixels que, além de estarem próximos, também possuam corem semelhantes. As imagens utilizadas nesse trabalho são 40 imagens de 16 bits (2^{16}níveis de cinza). Para tornar as bordas das imagens mais proeminentes, realizou-se um pré-processamento, reduzindo de 2^{16} para 64 níveis de cinza. Ao realizar esse pré-processamento perdeu-se alguns detalhes das imagens. Essa perda, no entanto, não impactou, de forma significativa, o objetivo de segmentação. Ao realizar a segmentação, comparou-se o resultado com 3 padrões ouro (obtidos manualmente por 2 radiologistas e 1 médico) e as imagens obtidas foram sobrepostas para saber o quanto de sobreposição e não-sobreposição havia. A métrica utilizada foi o Índice Dice. Os resultados quando comparados com cada um

dos 3 padrões ouro foram: Médico x Algoritmo: 0,953; Radiologista 1 x Algoritmo: 0,957 e Radiologista 2 x Algoritmo: 0,952.

Trabalhando com imagens de Tomografia Computadorizada (TC), (Skourt, Hassani, & Majda, 2018) apresentam uma proposta para a segmentação da região pulmonar. Para a realização da tarefa, os autores fazem uso de redes neurais convolutivas, usando a arquitetura U-NET, que consiste em uma etapa de contração para extrair informações de alto nível e uma etapa de expansão responsável pela recuperação das informações necessárias. O método foi aplicado na base de dados LIDC-IDRI, que consiste em imagens de tomografia computadorizada para diagnóstico e rastreamento de câncer de pulmão. A base é formada por 1018 casos. A métrica utilizada para mensurar a precisão do método foi a acurácia e o resultado obtido foi de 0,9502.

Os autores (Islam & Zhang, 2018) propõem um modelo robusto para realizar a segmentação da região pulmonar em imagens de radiografia torácica. O modelo proposto aprende a ignorar as regiões irrelevantes em uma radiografia de tórax de entrada, enquanto destaca as regiões úteis para a segmentação pulmonar. O trabalho foi desenvolvido utilizando uma estrutura automatizada para a segmentação pulmonar usando uma rede convolutiva baseada na arquitetura U-Net, que possui uma etapa de contração (conhecida também como etapa de convolução), seguida de outra etapa de expansão (também conhecida como etapa de deconvolução). Duas bases de dados foram utilizadas pelos autores: *Montgomery e Shenzhen*. A base de dados *Montgomery* consiste em 138 imagens, onde 80 são normais e 58 apresentam uma gama de anormalidades. A base de dados *Shenzhen* é composta por 367 imagens, sendo 340 normais e 27 apresentam manifestações de tuberculose. Na metodologia utilizada pelos autores, 80% das imagens foi usada para treinamento da rede e 20% para teste. Além do mais, usou-se 10% do conjunto de treinamento como dados de validação. Os autores fazem uso de várias técnicas de aumento de dados, como *zoom*, corte etc. A métrica utilizada pelos autores para avaliação do método foi o Índice Dice = 0,986.

Procurando investigar a influência de estruturas ósseas na tarefa de segmentação, (Gordienko et al., 2019) estudam o potencial das redes convolutivas em realizar segmentação pulmonar para dois tipos de imagens de radiografia: primeiramente, com a imagem de radiografia original e, posteriormente, com as mesmas imagens de radiografia, porém após sofrerem uma etapa de pré-processamento, em que se retirou as estruturas ósseas, ou seja, costelas e clavículas. A arquitetura utilizada foi a rede convolutiva U-NET. Uma rede convolutiva com 7 camadas 2D foi utilizada para treinamento utilizando o conjunto de dados originais e os dados com eliminação óssea. O autor fez uso das bases *Japanese Society of*

Radiological Technology (JSRT) e *Bone Shadow Exclusion-JSRT (BSE-JSRT)*. Analisando-se os resultados obtidos para a acurácia e para função de perda durante a fase de treinamento e validação da rede, os autores concluem que a segmentação é obtida de melhor forma quando se realiza um pré-processamento para a retirada das estruturas ósseas da imagem de radiografia.

Os autores (Novikov et al., 2018) têm como objetivo em seu trabalho a segmentação das estruturas anatômicas de radiografia torácica. Para isso, os autores fazem uso de redes convolutivas com arquitetura U-Net. Porém, como novidade, os autores propõem modificações na arquitetura original como forma de driblar alguns problemas, como por exemplo, um número reduzido de dados para treinamento da rede, ou para tentar evitar o *overfiting*. Para o estudo, usou-se o banco de dados JSRT e o Índice Dice como métrica de avaliação, obtendo-se os seguintes resultados: 0,950 para segmentação do pulmão; 0,868 para a clavícula e 0,882 para o coração.

A Tabela 1 sintetiza todos os trabalhos analisados que realizam a tarefa de segmentação e utilizam as mais diversas bases de dados disponíveis.

Tabela 1 Síntese da revisão bibliográfica de segmentação em imagens médicas

Autores	Objetivo	Base de dados	Metodologia	Métricas	Resultados
Ginneken *et al.* (2005)	Segmentação de estruturas anatômicas de radiografias do peito	*Japanese Society Radiological Technology (JSRT)*	*Active shape model segmentation, Active appearance models* e *Pixel classification*	Coeficiente de Jaccard	Pulmão: ASM:0,903±0,057 AAM: 0,847±0,095 PC: 0,938±0,027 Coração: ASM:0,793±0,119 AAM: 0,775±0,135 PC: 0,811±0,077 Clavícula: ASM:0,690±0,143 AAM: 0,505±0,234 PC: 0,618±0,100
Cavalcanti *et al.* (2016)	Segmentação de nódulo pulmonar em imagens CT	*Hamilton Health Sciences (HHS) e Lung Image Database Consortium - Image Database Resource Initiative (LIDC-IDRI)*	Método de Lucas-Kanade e o método SubME. Usa vários *slices* de CT como se fossem frames de um vídeo.	*True positive rate* (TPR) *False positive rate* (FPR) Acurácia	Lucas-Kanade: TPR = 95,66% FPR = 0,98% Acurácia = 99,02% SubME: TPR = 93,53% FPR = 0,89% Acurácia = 99,11%
Chondro *et al.* (2017)	Segmentação do pulmão em imagens de radiografia	JSRT e *Montgomery*	Equalização de histograma, binarização, filtro Gaussiano, operações morfológicas e refinamento dos limites da área desejada.	Coeficiente de Jaccard Índice Dice Distância média de contorno	Dataset JRST: 0,963±0,012 0,983±0,007 0,341±2,154 Dataset Montgomery: 0,966±0,018 0,978±0,005 0,394±0,623
Uzelaltinbulat e Ugur (2017)	Segmentação de um tumor pulmonar	70 imagens, 50 com tumor e 20 sem tumor - *Lung Image Database Consortium (LIDC)*	Erosão, filtro mediano, limiar de Otsu seguido de operações morfológicas.	Acurácia Sensibilidade Especificidade	97,14% 96% 100%

Hwang e Park (2017)	Segmentação pulmonar em imagens de radiografia	JSRT e *Montgomery*	Rede convolucional: arquitetura *deep-and-thin*	Coeficiente de similaridade de Jaccard Coeficiente de dados Distância média de contorno Distância média da superfície	0,961 ± 0,015 0,980 ± 0,008 1,237 ± 0,702 0,675 ± 0,122
Yassine *et al.* (2018)	Segmentação da região pulmonar em imagens de radiografia	40 imagens de 16 bits	Pré-processamento para redução dos níveis de cinza seguida da técnica de Superpixels. Comparação com padrão ouro feito manualmente por especialistas	Índice Dice	Médico x Algor: 0,953 Radiol1 x Algor: 0,957 Radiol2 x Algor: 0,952
Skourt *et al.* (2018)	Segmentação em imagens de pulmão	*Lung Image Database Consortium - Image Database Resource Initiative (LIDC-IDRI)*	Rede convolucional, com etapa de convolução e deconvolução	Acurácia	0,9502
Islam e Zhang (2018)	Segmentação pulmonar em imagens de radiografia	*Montgomery* e *Shenzhen*	Pré-processamento para redução do tamanho da imagem e aos isso passando por uma rede convolutiva: arquitetura U-net	Índice Dice	0,986
Gordienko *et al.* (2018)	Segmentação pulmonar em imagens de radiografia, com e sem estrutura óssea	JSRT e *Bone Shadow Exclusion (BSE-JSRT)*	Rede convolucional: arquitetura U-net, usando imagens com e sem estruturas ósseas	Acurácia e função de perda no treinamento e validação	Segmentação obtida de melhor forma quando realiza nas imagens sem estrutura óssea
Novikov et al. (2018)	Segmentação de estruturas anatômicas de radiografias do peito	JSRT	Redes convolutivas: arquitetura U-Net com modificações	Índice Dice	Pulmão: 0,950 Clavícula: 0,868 Coração: 0,882

Entre os trabalhos analisados destacam-se os trabalhos de *Ginneken et al.*, *Chondro et al.* e *Novikov et al.*, pois utilizam a mesma base de dados para realizar a mesma tarefa de segmentação, porém com diferentes métodos. Os resultados obtidos por esses autores foram utilizados como *benchmark*.

3 REFERENCIAL TEÓRICO

Neste capítulo será apresentado a teoria necessária para o entendimento do trabalho de segmentação. Começaremos descrevendo as Redes Neurais Artificiais para introduzirmos conceitos necessários, após isso será apresentado o funcionamento das Redes Neurais Convolutivas, elemento central desse trabalho.

3.1 REDES NEURAIS ARTIFICIAIS

Redes Neurais Artificiais (RNA) são modelos computacionais inspirados no sistema nervoso de seres vivos. Possuem capacidade de aquisição e manutenção do conhecimento (baseado em informações) e podem ser definidas como um conjunto de unidades de processamento, caracterizadas por neurônios artificiais, que são interligados por muitas interconexões (sinapses artificiais) (Silva, Spatti, & Flauzino, 2010).

As características mais relevantes envolvidas com aplicações de redes neurais artificiais são:

1. Adaptação por experiência: as adaptações dos parâmetros internos da rede, tipicamente seus pesos sinápticos, são ajustados a partir da apresentação sucessiva de exemplos (padrões, amostras, medidas) relacionados ao comportamento do processo, possibilitando a aquisição do conhecimento por experimentação;
2. Capacidade de aprendizado: por intermédio de aplicação de um método de treinamento, a rede consegue extrair o relacionamento existente entre as diversas variáveis que compõem a aplicação;
3. Habilidade de generalização: após o processo de treinamento, a rede é capaz de generalizar o conhecimento adquirido, possibilitando estimar soluções que eram até então desconhecidas;

As RNA podem ser empregadas em diversos problemas relacionados às engenharias e ciências. As potenciais áreas de aplicabilidade podem ser enquadradas conforme se segue:

1. Aproximador de funções: o objetivo consiste em mapear o relacionamento funcional entre as variáveis (tipicamente reais) de um sistema a partir de um conjunto conhecido de seus valores representativos. As aplicações são as mais

diversas possíveis, sendo que envolvem normalmente mapeamento de processos cuja modelagem por técnicas convencionais seja de difícil obtenção;

2. Controle de processos: o objetivo consiste em identificar ações de controle que permitam o alcance dos requisitos de qualidade, eficiência e segurança do processo. Entre as várias aplicações disponíveis destacam-se os controles empregados em robótica, aeronaves, satélites etc.;
3. Reconhecimento/classificação de padrões: o objetivo deste tipo de aplicação consiste em associar um padrão de entrada (amostra) para uma das classes previamente definidas, como acontece em reconhecimento de imagens, voz, escrita, etc. Neste caso, o problema a ser tratado possui um conjunto de amostras associadas às suas correspondentes classes;
4. Sistemas de previsão: o objetivo consiste em estimar valores futuros de um processo levando-se em consideração diversas medidas prévias observadas em seu domínio. Entre as aplicações disponíveis enquadram-se a previsão de séries temporais, previsões de mercados financeiros, previsões climáticas etc.;
5. Otimização de sistemas: o alvo consiste em minimizar ou maximizar uma função de custo (objetivo) obedecendo também eventuais restrições que são impostas para o correto mapeamento do problema.

3.1.1 Neurônio biológico e Neurônio artificial

O processamento de informações no cérebro humano é regido por elementos processadores biológicos que operam em paralelo, tendo como objetivo a produção de ações apropriadas para cada uma de suas funcionalidades, tais como o pensar e o memorizar.

A célula elementar do sistema nervoso cerebral é o neurônio, cujo papel é o de conduzir impulsos sob determinadas condições. Tal elemento biológico pode ser dividido em três partes principais: dendritos, corpo celular (ou soma) e axônio. Os dendritos são constituídos por vários finos prolongamentos, cuja função é captar, de forma contínua, os estímulos vindos de diversos outros neurônios ou do próprio meio externo. O corpo celular é incumbido de processar todas as informações advindas dos dendritos, e é nele também que se encontram as principais organelas citoplasmáticas (núcleo, mitocôndria, centríolo, lisossomo etc.) do neurônio. O axônio é constituído por um único prolongamento, cuja missão é conduzir os impulsos elétricos para outros neurônios conectores ou para aqueles se conectam diretamente com o tecido muscular.

A estrutura das RNA foi desenvolvida a partir desses modelos conhecidos de sistemas nervosos biológicos e do próprio cérebro humano. Os elementos computacionais ou unidades processadoras, denominadas neurônios artificiais, são modelos bem simplificados dos neurônios biológicos. A Figura 3 mostra a representação de um neurônio biológico, bem como a representação de um neurônio artificial.

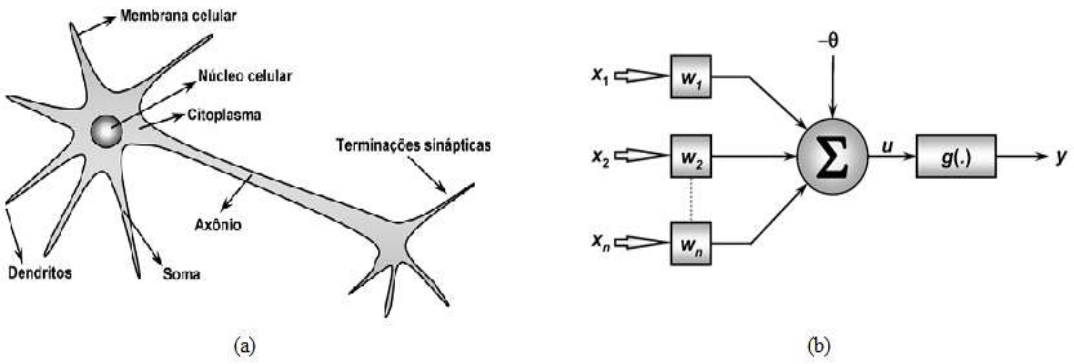


Figura 3 (a) Representação do neurônio biológico. (b) Representação do neurônio artificial
Fonte: (Silva, 2020)

De acordo com a Figura 3(b), um neurônio artificial é composto por sete elementos básicos:

1. Sinais de entrada $\{x_1, x_2, \dots, x_n\}$: São os sinais advindos do meio externo e que representam os valores assumidos pelas variáveis de uma aplicação específica.
2. Pesos sinápticos $\{w_1, w_2, \dots, w_n\}$: São os valores que servirão para ponderar cada uma das variáveis de entrada da rede, permitindo-se quantificar suas relevâncias em relação a funcionalidade do respectivo neurônio.
3. Combinador Linear $\{\Sigma\}$: Tem a função de agregar todos os sinais de entrada que foram ponderados pelos respectivos pesos sinápticos.
4. Limiar de ativação $\{\theta\}$: É uma variável que especifica qual será o patamar apropriado para que o resultado produzido pelo combinador linear possa gerar um valor de disparo em direção à saída do neurônio.
5. Potencial de ativação $\{u\}$: É o resultado produzido pela diferença do valor produzido entre o combinador linear e o limiar de ativação. Se tal valor é positivo, ou seja, $u \geq 0$ então o neurônio produz um potencial excitatório; caso contrário, o potencial será inibitório.

6. Função de ativação $\{g\}$: Seu objetivo é limitar a saída do neurônio dentro de um intervalo de valores associados ao *output* do problema.
7. Sinal de saída $\{y\}$: Consiste no valor final produzido pelo neurônio em relação a um determinado conjunto de sinais de entrada, podendo ser também utilizado por outros neurônios que estão sequencialmente interligados.

O neurônio artificial ilustrado na Figura 3(b) pode ser expresso através de duas equações, são elas:

$$u = \sum_{i=1}^{n} w_i x_i - \theta \tag{1}$$

$$y = g(u) \tag{2}$$

3.1.2 Funções de ativação

As principais funções de ativação utilizadas em RNA são:

1. Função degrau: Essa função é definida matematicamente pela expressão abaixo e seu gráfico está mostrado na Figura 4.

$$g(u) = \begin{cases} 1, se\ u \geq 0 \\ 0, se\ u < 0 \end{cases} \tag{3}$$

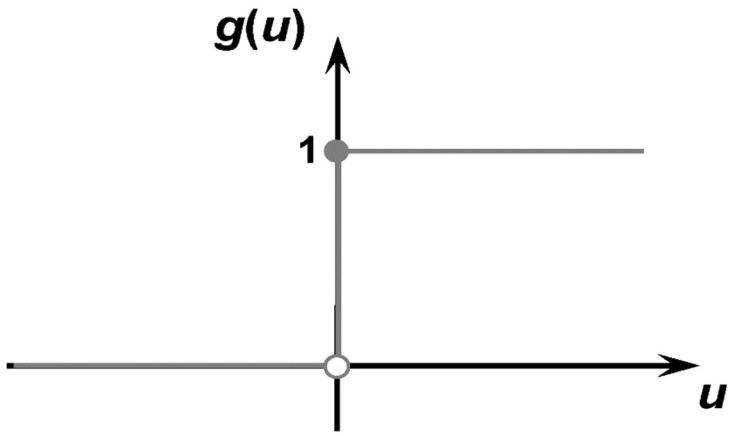


Figura 4 Função de ativação degrau

Fonte: (Silva, 2020)

2. Função degrau bipolar: Em representação matemática, tem-se a expressão:

$$g(u) = \begin{cases} 1, se\ u > 0 \\ 0, se\ u = 0 \\ -1, se\ u < 0 \end{cases} \tag{4}$$

A representação gráfica desta função está ilustrada na Figura 5.

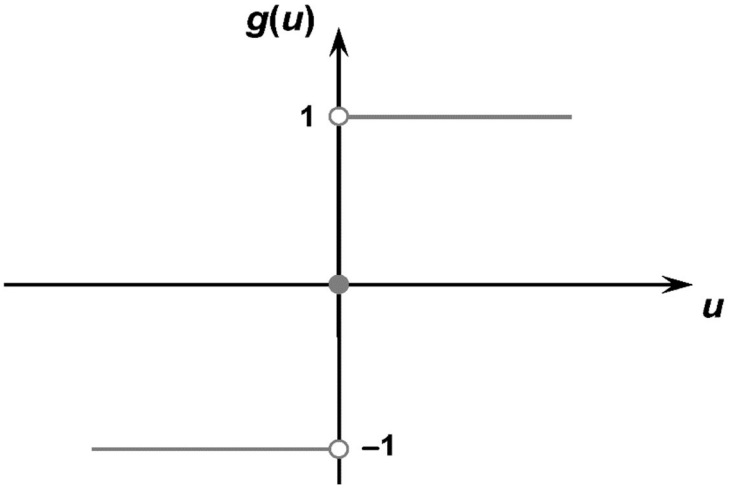


Figura 5 Função de ativação degrau bipolar

Fonte: (Silva, 2020)

3. Função logística (ou sigmoide): A saída produzida assumirá sempre valores reais entre zero e um, tendo sua expressão matemática mostrada a seguir:

$$g(u) = \frac{1}{1 + e^{-\beta u}} \tag{5}$$

A ilustração gráfica desta função é mostrada na Figura 6.

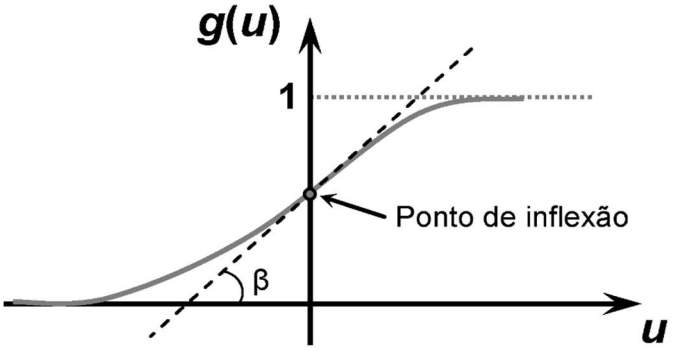


Figura 6 Função de ativação logística

Fonte: (Silva, 2020)

4. Função tangente hiperbólica: O resultado de saída, diferentemente da função logística, assumirá valores reais entre -1 e 1, cuja expressão é dada por:

$$g(u) = \frac{1 - e^{-\beta u}}{1 + e^{-\beta u}} \tag{6}$$

A representação gráfica da função está ilustrada na Figura 7.

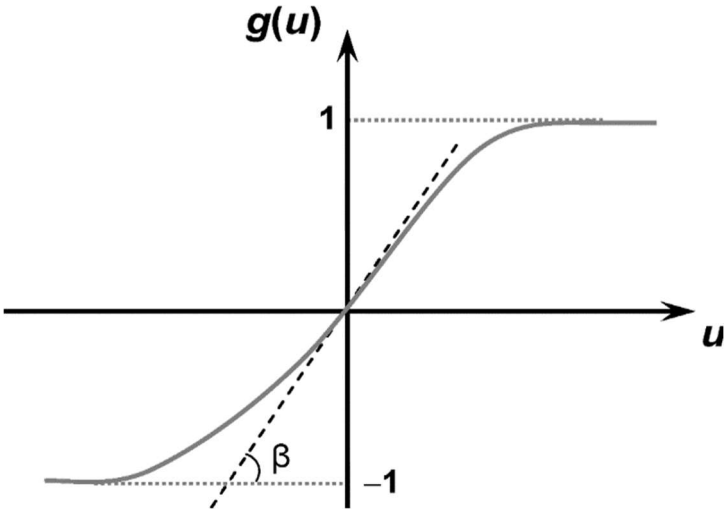


Figura 7 Função de ativação tangente hiperbólica

Fonte: (Silva, 2020)

3.1.3 Arquiteturas e processos de treinamento

Entre as principais arquiteturas de uma RNA podemos destacar a arquitetura *feedforward* de camada simples, arquitetura *feedforward* de camadas múltiplas e arquitetura recorrente. A Figura 8 mostra um exemplo de RNA *feedforward* de camadas múltiplas formada por uma camada de entrada composta por n sinais, duas camadas neurais escondidas constituídas respectivamente de n_1 e n_2 neurônios e, finalmente, uma camada neural de saída composta por m neurônios representando os possíveis valores de saída da aplicação.

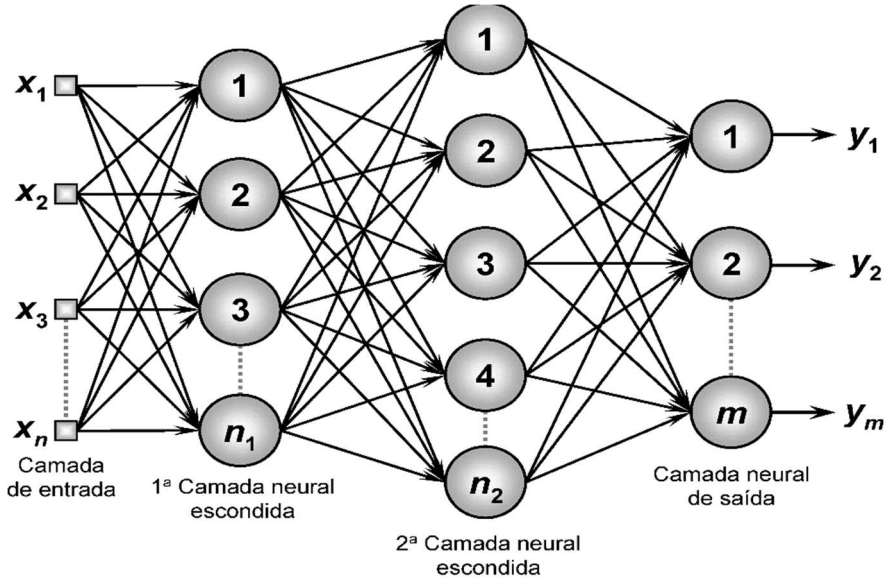


Figura 8 Exemplo de rede *feedforward* de camadas múltiplas

Um dos destaques mais relevantes das RNA está na capacidade de aprender a partir da apresentação de amostras (padrões) representativas do comportamento dos sistemas. Após a rede ter aprendido o relacionamento entre as entradas e saídas, esta é capaz de generalizar soluções. A rede será então capaz de produzir uma saída próxima daquela esperada (desejada) a partir de quaisquer sinais inseridos em suas entradas.

Portanto, o processo de treinamento de uma rede RNA consiste na aplicação de passos ordenados que sejam necessários para sintonização dos pesos sinápticos de seus neurônios, tendo-se como objetivo final a generalização de soluções a serem produzidas pelas suas saídas, cujas respostas são representativas do sistema físico em que estas estão mapeando.

Normalmente, o conjunto total de amostras disponíveis sobre o comportamento do sistema é dividido em dois subconjuntos, os quais são denominados de subconjunto de treinamento e subconjunto de teste. O subconjunto de treinamento, composto aleatoriamente com cerca de 60 a 90% das amostras do conjunto total, será usado essencialmente no processo de treinamento da rede. Já o subconjunto de teste, cuja composição está entre 10 e 40% do conjunto total de amostras, será utilizado para verificar se os aspectos referentes à generalização de soluções por parte da rede já estão em patamares aceitáveis, possibilitando assim a validação da topologia assumida.

Em relação ao treinamento de uma rede, existem algumas estratégias de treinamento, aos quais se destacam o treinamento supervisionado e o treinamento não-supervisionado.

A estratégia de treinamento supervisionado consiste em se ter disponível, considerando cada amostra dos sinais de entrada, as respectivas saídas desejadas, ou seja, cada

amostra de treinamento é composta pelos sinais de entrada e suas correspondentes saídas. Desta forma, há a necessidade de se disponibilizar uma tabela de dados (entradas/saídas) representativa do processo, também conhecida por tabela de atributos/valores, e que comtemple inclusive o seu comportamento, pois é a partir de tais informações que as estruturas neurais formularão as hipóteses sobre aquilo a ser aprendido.

Os pesos sinápticos são continuamente ajustados mediantes a aplicação de ações comparativas, executadas pelo próprio algoritmo de aprendizagem, que supervisionam a defasagem entre as respostas produzidas pela rede em relação àquelas desejadas, sendo esta diferença usada no procedimento de ajuste. A rede será considerada treinada quando tal defasagem estiver dentro de valores aceitáveis, levando-se em consideração os propósitos de generalização de soluções.

Diferentemente do treinamento supervisionado, durante a aplicação de um algoritmo de aprendizado não-supervisionado inexistem as respectivas saídas desejadas.

Desta forma, no treinamento não-supervisionado a própria rede se auto organiza em relação às particularidades existentes entre os elementos componentes do conjunto total de amostras, identificando subconjuntos (*clusters*) que contenham similaridades. Os pesos sinápticos dos neurônios da rede são então ajustados pelo algoritmo de aprendizado de forma a refletir esta representação internamente dentro da própria rede.

Ao se realizar a apresentação de amostras para a rede, isso pode ser feito de duas formas: usando lote de amostras (*off-line*) e usando amostra-por-amostra (*online*).

Na aprendizagem usando lote de amostras, também denominada de aprendizagem *offline* ou *batch*, os ajustes efetuados nos vetores de pesos das redes só são efetivados após a apresentação de todo o conjunto de treinamento, pois cada passo de ajuste leva em consideração o total de desvios observados nas amostras de treinamento frente aos respectivos valores desejados para as suas saídas.

Para a aprendizagem usando amostra-por-amostra (*online*), ao contrário daquela por lote, os ajustes nos pesos das redes são efetuados após a apresentação de cada amostra de treinamento. Portanto, após a execução do passo de ajuste, a respectiva amostra pode ser descartada. Esse tipo de aprendizagem é normalmente utilizado quando o comportamento do sistema a ser mapeado varia de forma bastante rápida, sendo quase impraticável a adoção do aprendizado *off-line*, pois as amostras utilizadas em determinado instante podem não mais representar o comportamento do processo nos instantes posteriores.

3.1.4 Processo de treinamento da *feedforward* de múltiplas camadas

Para entender como funciona o processo de treinamento e atualização dos seus pesos sinápticos, vamos considerar a RNA mostrada na Figura 9, onde será aplicado o algoritmo de aprendizagem conhecido como *Backpropagation*

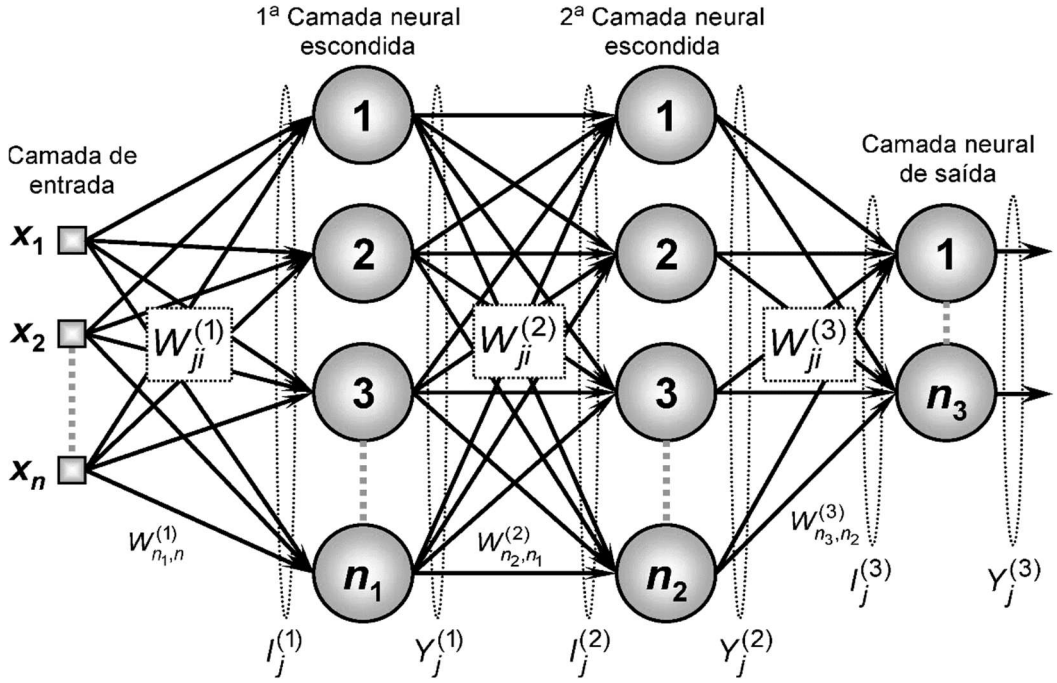


Figura 9 Notação das variáveis para elaboração do algoritmo *Backpropagation*

A função representativa do erro de aproximação, cuja incumbência será medir o desvio entre as respostas produzidas pelos neurônios de saída da rede em relação aos respectivos valores desejados, é dada por:

$$E(k) = \frac{1}{2}\sum_{j=1}^{n_3}\left(d_j(k) - Y_j^3(k)\right)^2 \tag{7}$$

onde $Y_j^3(k)$ é o valor produzido pelo j-ésimo neurônio de saída da rede considerando-se a k-ésima amostra de treinamento, enquanto $d_j(k)$ é o seu respectivo valor desejado.

Assumindo-se um conjunto de treinamento composto por p amostras, a medição da evolução do desempenho global do algoritmo pode ser efetuada por meio da avaliação do erro médio quadrático:

$$E_M = \frac{1}{p}\sum_{1}^{p} E(k) \tag{8}$$

O objetivo do processo de treinamento para a camada neural de saída consiste em ajustar os pesos sinápticos de saída a fim de minimizar o erro produzido entre a saída produzida pela rede em relação à respectiva saída desejada. Empregando a definição de gradiente e explorando a regra de diferenciação em cadeia:

$$\nabla E^{(3)} = \frac{\partial E}{\partial W_{ji}^3} = \frac{\partial E}{\partial Y_j^3}\frac{\partial Y_j^3}{\partial l_j^3}\frac{\partial l_j^3}{\partial W_{ji}^3} \tag{9}$$

$$\frac{\partial E}{\partial W_{ji}^3} = -\left(d_j - Y_j^3\right).g'\left(l_j^3\right).Y_i^2 \tag{10}$$

O ajuste dos pesos sinápticos deve ser feito em direção oposta ao gradiente a fim de minimizar o erro, ou seja:

$$\Delta W_{ji}^3 = -\eta\frac{\partial E}{\partial W_{ji}^3} \tag{11}$$

$$W_{ji}^3(t+1) = W_{ji}^3(t) + \eta.\delta_j^3.Y_i^2 \tag{12}$$

onde η é a taxa de aprendizagem do algoritmo e

$$\delta_j^3 = \left(d_j - Y_j^3\right).g'\left(l_j^3\right) \tag{13}$$

é definido como o gradiente local em relação ao j-ésimo neurônio da camada de saída.

Usando pensamento análogo para as outras camadas da rede, podemos deduzir que os pesos sinápticos são atualizados segundo as equações abaixo:

$$W_{ji}^2(t+1) = W_{ji}^2(t) + \eta.\delta_j^2.Y_i^1 \tag{14}$$

$$W_{ji}^1(t+1) = W_{ji}^1(t) + \eta.\delta_j^1.x_i \tag{15}$$

O critério de parada do processo fica estipulado em função do erro quadrático médio, levando-se em conta todas as amostras de treinamento disponíveis. O algoritmo converge quando o erro quadrático médio entre duas épocas sucessivas for suficientemente pequeno, ou seja:

$$\left|E_M^{atual} - E_M^{anterior}\right| \leq \tau \tag{16}$$

onde τ é a precisão requerida para o processo de convergência e seu valor deve ser previamente estabelecido. O valor da precisão varia e depende de fatores como, por exemplo, tipo de dados e a tarefa para qual a RNA está sendo treinada.

3.2 REDES CONVOLUTIVAS

Desde 1943, quando Warren McCulloch e Walter Pitts (McCulloch & Pitts, 1943) criaram o primeiro modelo computacional para redes neurais baseadas em matemática e algoritmos, até os dias de hoje, ocorreram inúmeros avanços na área de aprendizado de máquina, mas isso só foi possível pelo avanço que também ocorreu na área da computação, surgindo sempre processadores mais eficazes e velozes.

Graças a esse fato surgiu o campo de estudo conhecido como *Deep Learning*, termo traduzido como Aprendizado Profundo. Os métodos de aprendizagem profunda são baseados em representações distribuídas em camadas. Estas camadas correspondem às abstrações de características, onde cada camada de característica de um nível mais alto de hierarquia utiliza abstrações de camadas de características de nível mais baixo, simulando o córtex visual humano (Goodfellow, Bengio, & Courville, 2016).

A ideia de Rede Neurais Convolutivas (CNN) surgiu em 1998, com LeCun, em seu trabalho sobre reconhecimento de dígitos *Object Recognition with Gradient-based Learning* (LeCun, Haffner, Bottou, & Bengio, 1998). As CNN têm esse nome por fazerem uso da operação matemática de convolução em uma de suas camadas e são adequadas para trabalharem com grandes dados, como imagens ou sinais amostrados. As CNN combinam três ideias em suas arquiteturas para garantir um grau de invariância de deslocamento, escala e distorção: campo receptivo local, pesos compartilhados e sub-amostragem (LeCun et al., 1998). As principais camadas desse tipo de redes são: Camada de Convolução, Camada de *Pooling*, Camada de Unidades Lineares Retificadoras, Camada de Convolução Transposta, Camada de Classificação, Camada *Dropout* e Camada *Batch Normalization*. A Figura 10 mostra uma típica

arquitetura de CNN, chamada de LeNet-5 e utilizada para reconhecer formas. Nas próximas seções será descrito o funcionamento de cada uma de suas camadas.

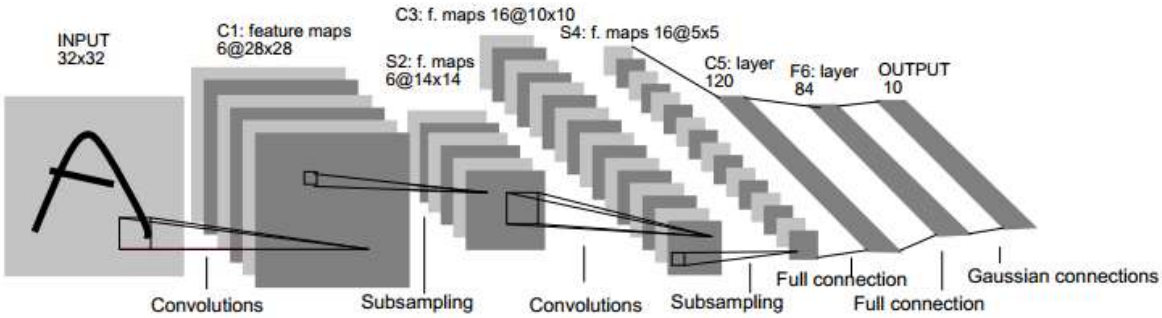


Figura 10 Arquitetura da LeNet-5, uma Rede Neural Convolutiva usada para reconhecimento de dígitos

Fonte: (Adaptado de LeCun, 1998)

3.2.1 Camada de Convolução

A camada de convolução é a primeira camada de uma CNN logo após a entrada da rede e ela é responsável por criar mapas de ativação a partir dos dados de entrada.

Na forma geral, a operação de convolução realizada entre duas funções $f(t)$ e $g(t)$ é definida como:

$$f(t) * g(t) = \int_{-\infty}^{\infty} f(t)g(t-\tau)\, d\tau \tag{17}$$

Sua versão discreta é definida como:

$$f[n] * g[n] = \sum_{m=0}^{n} f[m]g[n-m] \tag{18}$$

Para aplicações em CNN, a operação de convolução é geralmente aplicada em mais de um eixo, pois geralmente os dados de entrada da rede estão organizados em grade. Para uma imagem I e um kernel K, temos no pixel (i, j) da imagem resultante,

$$(I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{19}$$

Como a convolução é uma operação comutativa, podemos escrever a equação acima como:

$$(K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \tag{20}$$

Em aplicações envolvendo CNN, não é usual empregar esta propriedade devido a fatores inerentes ao processo de aprendizado da rede. Ao invés disso, é implementada a função correlação cruzada:

$$(K * I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \tag{21}$$

Para exemplificar a dinâmica do processo de convolução, vamos considerar o sinal em uma dimensão da Figura 11(a) (que pode ser entendido como uma linha de uma imagem em escala de cinza), bem como o *kernel* da Figura 11(b), e realizar a convolução entre eles:

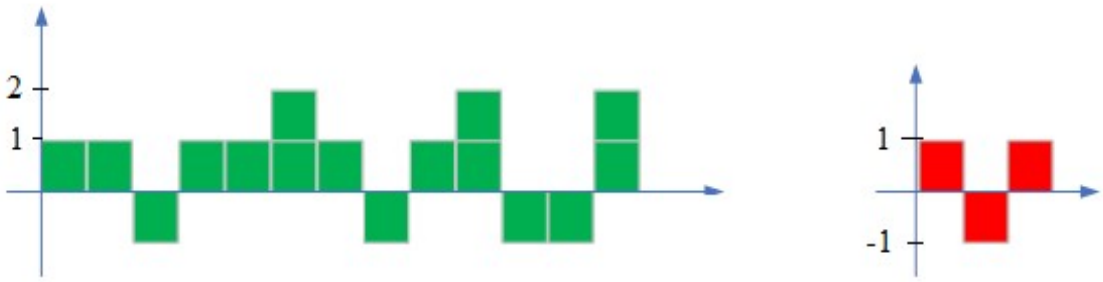


Figura 11 (a) Sinal a ser convoluído com o *kernel*. (b) *Kernel*

A Figura 12 ilustra o primeiro e o segundo passo da convolução.

Passo 1: $(1).(1) + (1).(-1) + (-1).(1) = -1$

Passo 2: $(1).(1) + (-1).(-1) + (1).(1) = 3$

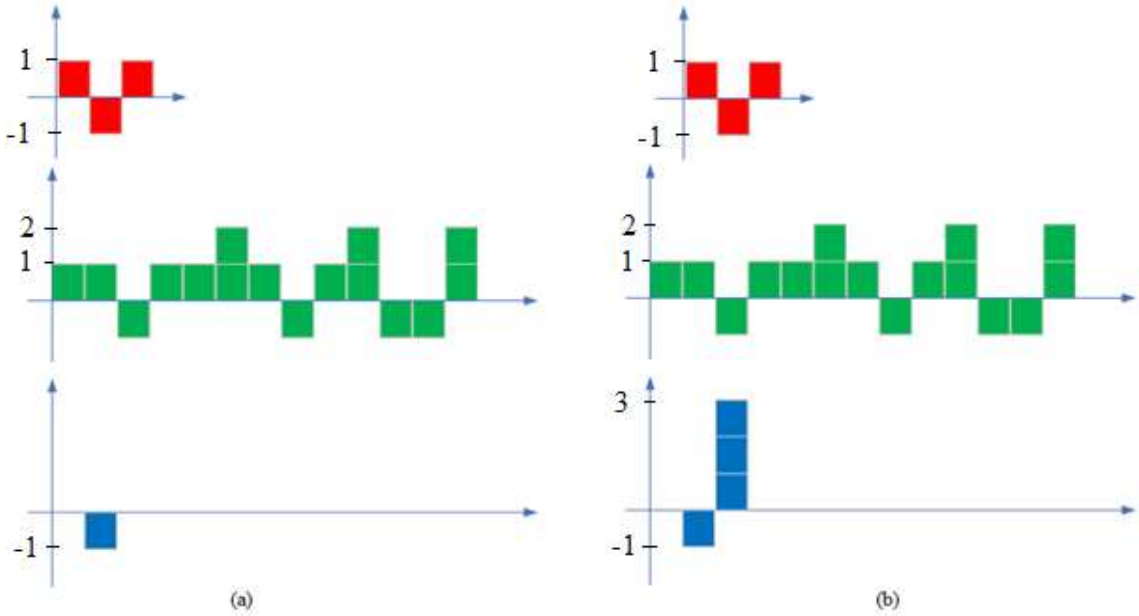


Figura 12 (a) Primeiro passo da convolução. (b) Segundo passo da convolução

A Figura 13 mostra o resultado da convolução, após o *kernel* percorrer todo o sinal.

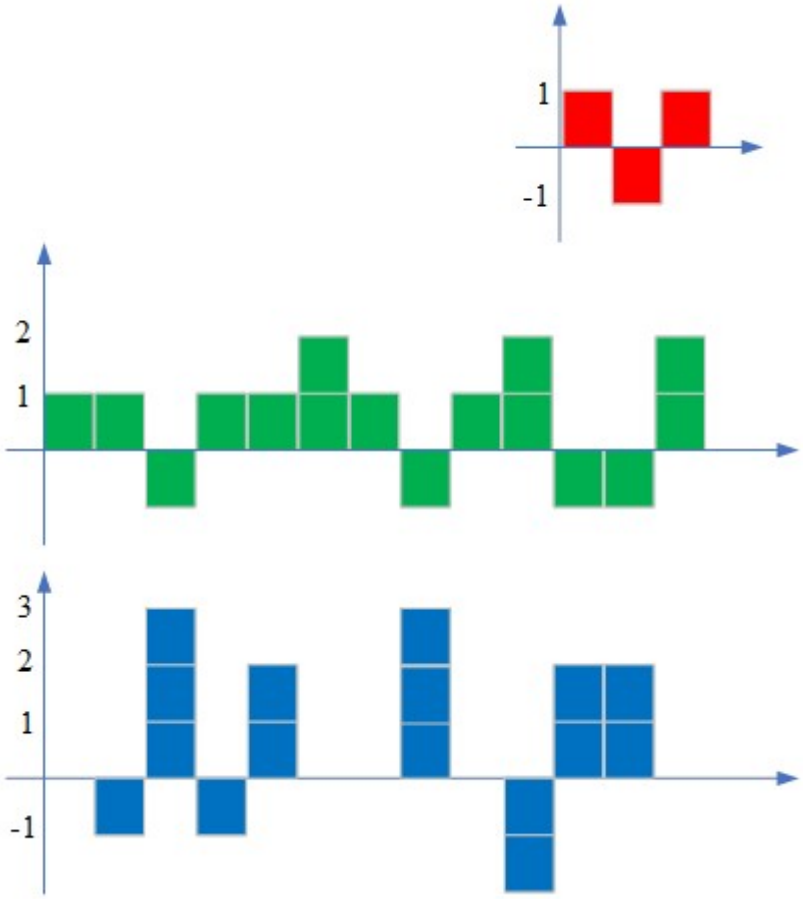


Figura 13 Resultado da convolução

Podemos notar no sinal resultante obtido pela convolução que seu valor máximo é 3 e, esse valor ocorre exatamente quando o padrão representado pelo kernel aparece da mesma forma no sinal original, ou seja, esse *kernel* busca reconhecer o padrão $[1, -1, 1]$. É exatamente isso que ocorre na camada de convolução de uma CNN, onde cada mapa de ativação da camada será responsável por buscar/reconhecer determinado padrão da imagem, como, por exemplo, borda horizontal, borda vertical, círculos, diagonais etc. Esse reconhecimento se dá através da atualização dos coeficientes do *kernel*.

A Figura 14 mostra o procedimento quando trabalhamos com funções 2-D, como por exemplo, imagens:

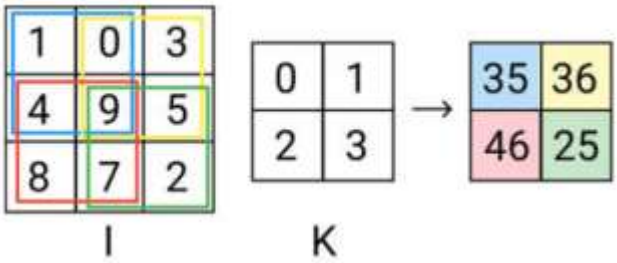


Figura 14 Convolução utilizando funções 2-D

Aqui nesse ponto podemos notar uma das vantagens das CNN frente as RNA: para sabermos o valor que determinado neurônio irá assumir, não precisamos dos valores de todos os neurônios da camada anterior, mas apenas dos neurônios envoltos pelo *kernel*. Isso faz com que diminua o número de conexões entre as camadas. A Figura 15 ilustra esse fato.

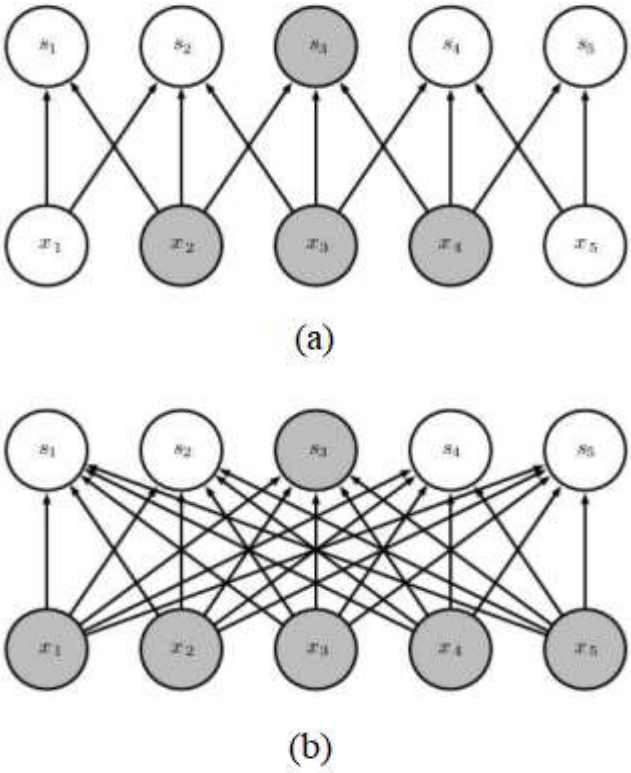


Figura 15 (a) Estrutura de uma CNN com conectividade esparsa. (b) Estrutura de uma RNN tradicional completamente conectada

Podemos expressar a convolução também como uma multiplicação de matrizes, onde, por exemplo, expressamos a convolução da Figura 14 como mostrada na Figura 16.

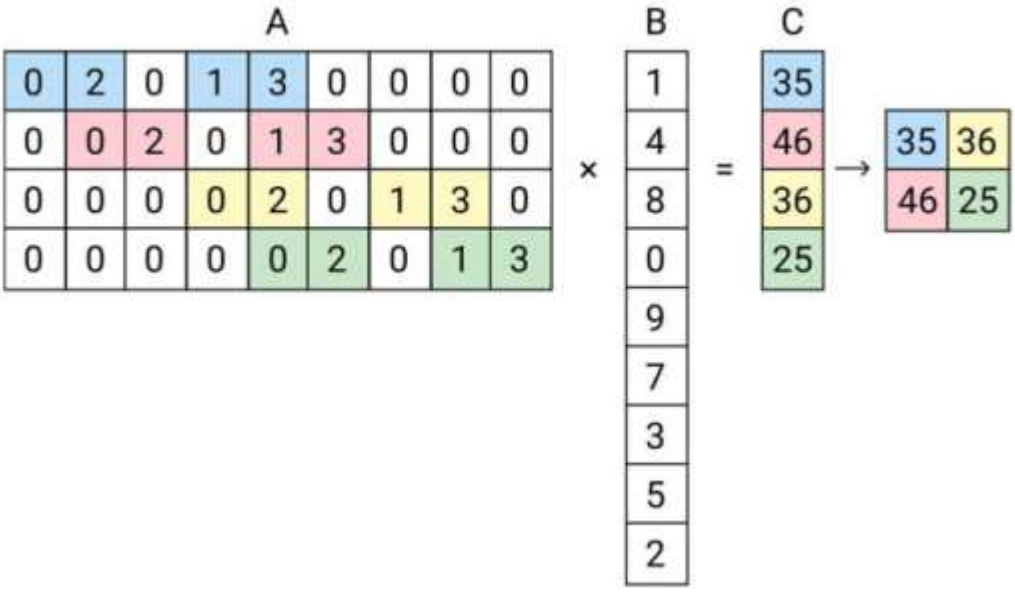


Figura 16 Convolução mostrada como multiplicação de matrizes

3.2.2 Camada *Batch Normalization*

A camada *Batch Normalization* soluciona um problema onde, durante o treinamento da rede, as mudanças dos parâmetros da rede causam mudanças na distribuição das ativações da rede nas camadas ocultas da rede. Este problema, definido como "Mudança de covariância interna", é solucionado através de uma etapa de normalização que corrige a média e a variância da saída da camada anterior. O *Batch Normalization* permite taxas de aprendizado maiores e contribui na normalização do modelo, sendo que em redes que utilizam essa camada, a camada de *Dropout* pode deixar de ser utilizada ou ter sua influência reduzida.

Para centralizar e normalizar as entradas, o algoritmo necessita estimar a média e o desvio padrão de cada uma delas, para isso, são utilizadas informações advindas de um subconjunto amostral das entradas, conhecido como *mini-batch*, por isso o nome *Batch Normalization*. O passo a passo do algoritmo pode ser descrito pelas equações:

$$\mu_B = \frac{1}{m_B}\sum_{i=1}^{m_B} x^{(i)} \tag{22}$$

$$\sigma_B^2 = \frac{1}{m_B}\sum_{i=1}^{m_B} (x^{(i)} - \mu_B)^2 \tag{23}$$

$$x_N^{(i)} = \frac{x^{(i)} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{24}$$

$$z(i) = \gamma x_N^{(i)} + \beta \tag{25}$$

onde, μ_B é a média empírica avaliada sobre o *mini-batch* B inteiro, σ_B é o desvio padrão empírico também avaliado sobre todo o *mini-batch*, m_B é o número de instâncias do *mini-batch*, $x_N^{(i)}$ representa a imagem centralizada e normalizada, γ é o parâmetro de escala da camada, β é o parâmetro de deslocamento da camada, ϵ representa uma constante pequena para evitar divisão por zero e $z(i)$ representa a saída da operação *batch normalization*.

3.2.3 Camada de Unidades Lineares Retificadoras

Em RNA tradicional é comum o uso da função de ativação sigmoide. Em CNN, porém, é mais usual utilizar a função *ReLu*, definida por:

$$f(x) = \max(0, x) \tag{26}$$

pois funções como a sigmoide e a tangente hiperbólica tornam o processo de treinamento da rede mais lento, uma vez que seus gradientes são fracos após determinados valores de potencial de ativação, u. O gráfico da *ReLu* e sua derivada estão mostrados na Figura 17.

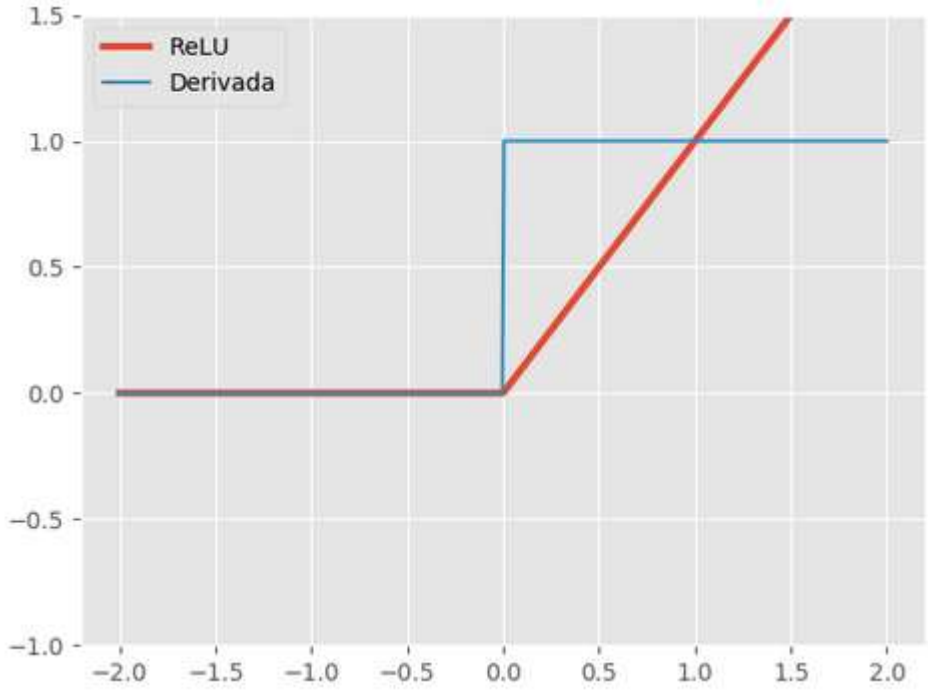


Figura 17 Função de ativação *ReLu* e sua derivada

3.2.4 Camada de *Pooling* (ou sub-amostragem)

A camada de *Pooling*, também chamada de camada de sub-amostragem, tem como objetivo diminuir a sensibilidade da rede a pequenas alterações da imagem, como uma pequena rotação, ou translação.

Um procedimento comum para o *Pooling* é conhecido como *pool* máximo (ou *Max-Pooling*). No *Max-Pooling*, uma unidade de *Pooling* simplesmente gera a ativação máxima na região de entrada 2x2, conforme ilustrado nas Figura 18 e Figura 19.

Podemos notar que a operação de *Pooling* reduz a dimensão na sua saída. A dimensão da saída é controlada pelo tamanho da máscara e pelo passo (*stride*) dado por essa máscara.

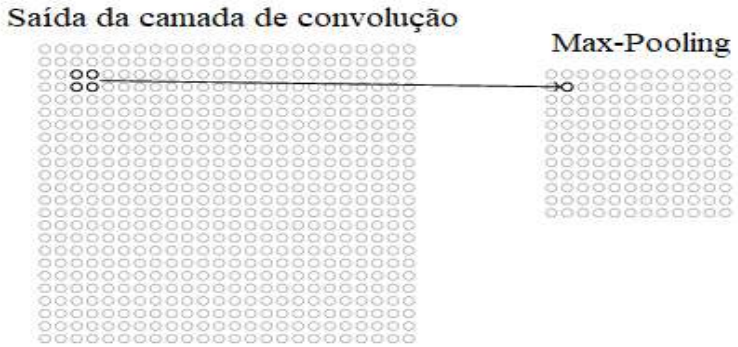


Figura 18 Operação *Max-Pooling*

A Figura 19 mostra a operação de *Max-Pooling* utilizando exemplo numérico:

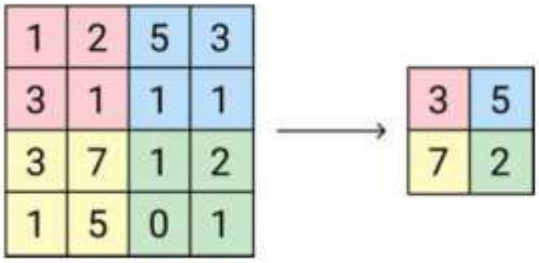


Figura 19 Operação *Max-Pooling* utilizando exemplo numérico

3.2.5 Camada de Convolução Transposta (ou sobre-amostragem)

A convolução transposta é uma operação matemática cuja finalidade é realizar a sobre-amostragem da rede. Seu uso é necessário para que as imagens possam retornar para seu tamanho original, recuperando informações perdidas, uma vez que foram reduzidas na etapa de sub-amostragem.

Tomando como exemplo a operação realizada na Figura 16, a convolução é uma multiplicação de matrizes que resultou em um vetor $(4x1)$:

$$A_{(4x9)}xB_{(9x1)} = C_{(4x1)} \tag{27}$$

A convolução transposta é usada para, a partir de uma matriz $(4x1)$ e um *kernel* $(4x9)$ retornar para uma matriz $(9x1)$.

$$(A^T)_{(9x4)}.\,vec(I)_{(4x1)} = E_{(9x1)} \tag{28}$$

A **Figura** 20 ilustra a operação realizada durante a deconvolução:

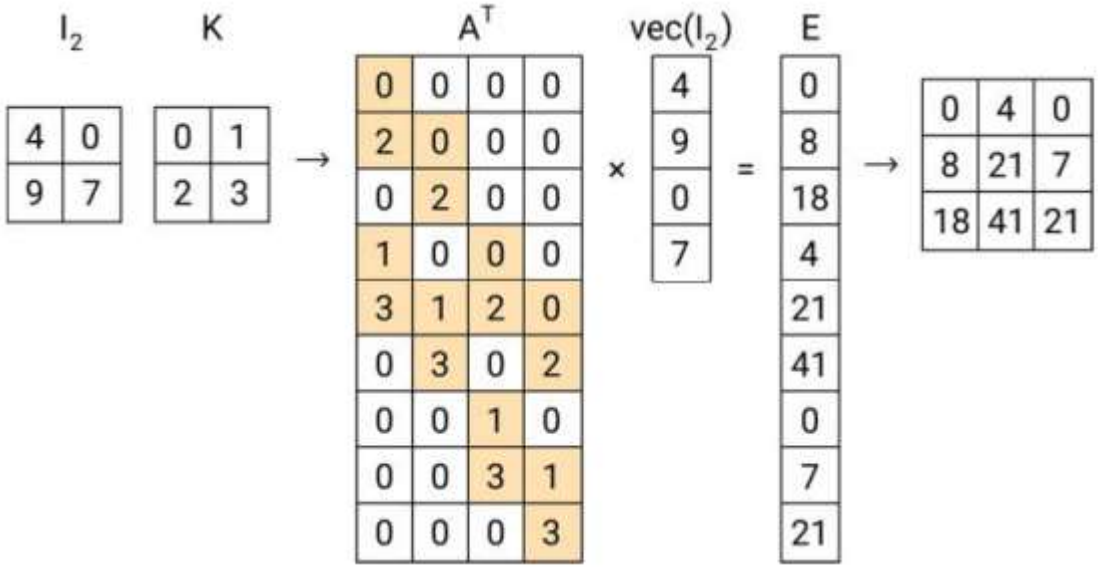


Figura 20 Operação de deconvolução ou sobre-amostragem

3.2.6 Camada de Classificação

Ao final de uma CNN é utilizada uma camada de classificação, que tem por objetivo estimar uma saída para a rede e, em fase de treinamento, calcular a diferença entre a saída estimada e o valor desejado para que os parâmetros de aprendizado sejam ajustados de forma que esta diferença seja minimizada. Normalmente utiliza-se a função *Softmax* por ser a mais apropriada quando a tarefa de classificação envolve múltiplas classes.

A função *Softmax* também é um tipo de função Sigmoide, mas é útil quando tentamos lidar com problemas de classificação. A função Sigmoide como vimos anteriormente é capaz de lidar com apenas duas classes. A função *Softmax* transforma as saídas para cada classe para valores entre 0 e 1 e divide pela soma das saídas. Isso essencialmente dá a probabilidade de a entrada estar em uma determinada classe.

Seja x a ativação dos nós, W o vetor de pesos e b o bias da camada conectada à camada *Sofmax*, como mostrado na Figura 21. O valor de ativação a da entrada da camada *Softmax* é dado pela equação:

$$a_i = \sum_j x_i W_{i,j} + b_i \tag{29}$$

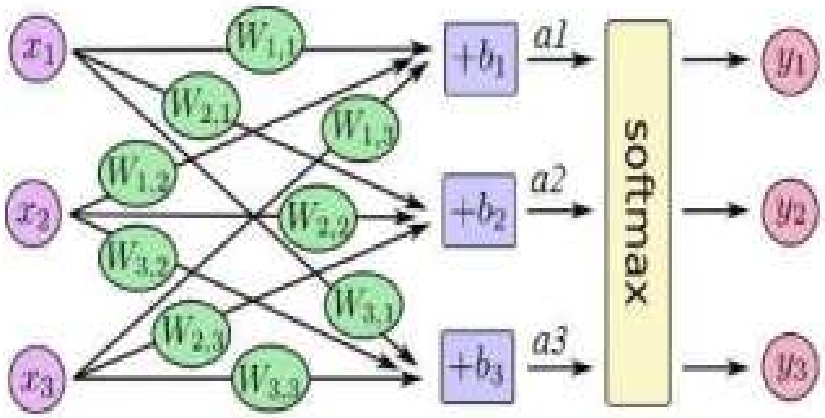


Figura 21 Função *Softmax*

A probabilidade p de a é dada por:

$$p_i = \frac{e^{a_i}}{\sum_j^n e^{a_j}} = y_i \tag{30}$$

A partir desses resultados, a classe estimada î será aquela com maior probabilidade.

$$\hat{\imath} = \arg\max\,(p_i) \tag{31}$$

3.2.7 Camada *Dropout*

Uma forma de evitar o *overfiting* da rede é utilizando uma camada de *Dropout*. Durante o treinamento, as unidades possuem uma probabilidade p de não serem desativadas. As unidades desativadas não participam da rede, como mostra na Figura 22.

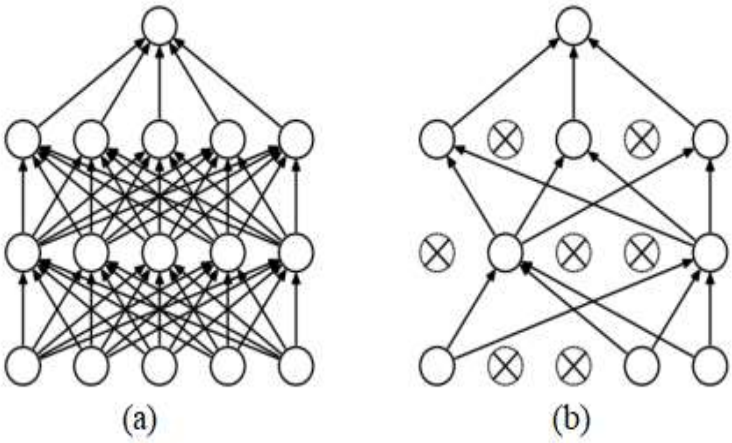


Figura 22 Ilustração da ação da camada *Dropout*. (a) Rede e suas conexões. (b) Resultado da aplicação do *Dropout*

A desativação destes neurônios é temporária e aleatória, feita somente durante a fase de treinamento da CNN. Ao final os neurônios que foram desligados têm seus parâmetros reajustados, multiplicando seus pesos w pela probabilidade p.

3.3 PROCESSO DE TREINAMENTO DE UMA REDE CONVOLUTIVA

Na Seção 3.1.3 foi mostrado que existem 2 tipos de treinamentos para redes neurais, o treinamento supervisionado e o treinamento não-supervisionado. Em CNN é utilizado o treinamento supervisionado, ou seja, para cada padrão de entrada, existe uma saída desejada. Durante o processo de treinamento da rede a saída produzida por ela é comparada com a saída desejada e, havendo diferença entre as duas, um erro é produzido. Após a detecção desse erro os parâmetros da rede serão ajustados de forma a minimizar esse erro. O algoritmo responsável por minimizar esses erros foi mencionado na Secção 3.1.4, algoritmo *Backpropagation*.

O algoritmo *Backpropagation* opera da camada de saída da rede em direção à camada de entrada, propagando o erro durante seu caminho. Como as funções de ativação de cada camada são deriváveis, o algoritmo utiliza-se da regra da cadeia para computar a derivada do erro em relação aos parâmetros (pesos sinápticos) da rede tendo apenas o valor do erro na camada de saída da rede. Considerando uma rede com n camadas, a atualização dos pesos sinápticos é feita através das equações abaixo:

$$\frac{\partial E}{\partial W_{ji}^{n}} = -\left(d_j - Y_j^n\right).\, g'\left(l_j^n\right).\, Y_i^{n-1}$$

$$W_{ji}^{n}(t+1) = W_{ji}^{n}(t) + \eta.\delta_{j}^{n}.Y_{i}^{n-1}$$

$$W_{ji}^{n-1}(t+1) = W_{ji}^{n-1}(t) + \eta.\delta_{j}^{n-1}.Y_{i}^{n-2}$$

.

.

.

$$W_{ji}^{1}(t+1) = W_{ji}^{1}(t) + \eta.\delta_{j}^{1}.x_{i}$$

Vale salientar nesse ponto que, para melhorarmos o desempenho de uma rede neural não basta acrescentarmos mais camadas em sua arquitetura, pois o seu processo de aprendizagem pode se tornar extremamente lento e, consequentemente, não satisfatório. Isso acontece devido ao fato de precisarmos da derivada da função de ativação e, quando colocamos várias camadas, a derivada da função resultante será extremamente pequena (quase nula). Esse fato pode ser observado na Figura 23, onde o gráfico em azul representa a função sigmoide, o gráfico em verde representa a função resultante quando colocamos duas camadas, o gráfico em vermelho representa a função resultante quando colocamos três camadas e, por fim, o gráfico em ciano é o resultado de quatro camadas. Podemos perceber pelo gráfico que ao trabalharmos com mais de duas camadas a derivada da função resultante é muito pequena e, consequentemente, a atualização de pesos sinápticos ocorrerá de forma muito lenta.

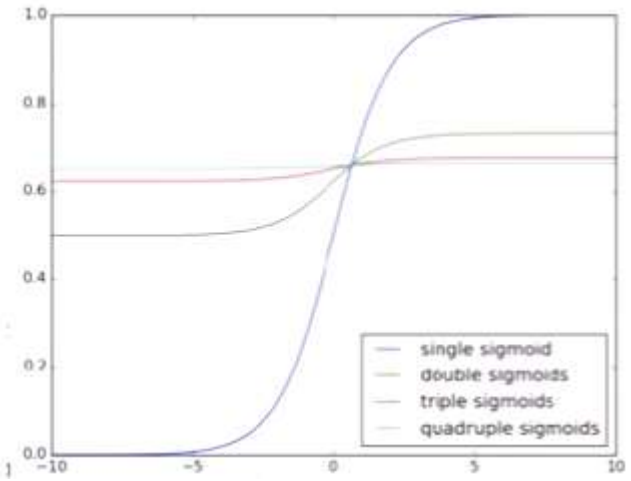


Figura 23 Comportamento da função Sigmoide em redes com uma, duas, três e quatro camadas

Para melhorarmos o treinamento de uma CNN outros métodos são utilizados, como, por exemplo, utilizar algoritmos que possuam taxa de aprendizagem adaptativa. Esses métodos são apresentados a seguir.

3.3.1 Métodos de Otimização utilizados no treinamento

Nesse tópico serão apresentados os métodos de otimização utilizados nesse trabalho, são eles: Gradiente Descendente Estocástico com *Momentum* (SGDM), Propagação de raiz quadrada média (RMSProp) e ADAM (termo derivado da estimação de momento adaptativa).

Quando treinamos uma CNN, essencialmente o que fazemos é procurar parâmetros que minimize uma determinada função de perda (*Loss function*). O valor dessa função nos dá uma medida do quão perfeito é o desempenho de nossa CNN para um determinado conjunto de padrões.

Considerando uma rede que possua apenas dois parâmetros (na prática esse número gira em torno de milhões ou bilhões), uma possível função de perda pode se parecer como a mostrada na Figura 24.

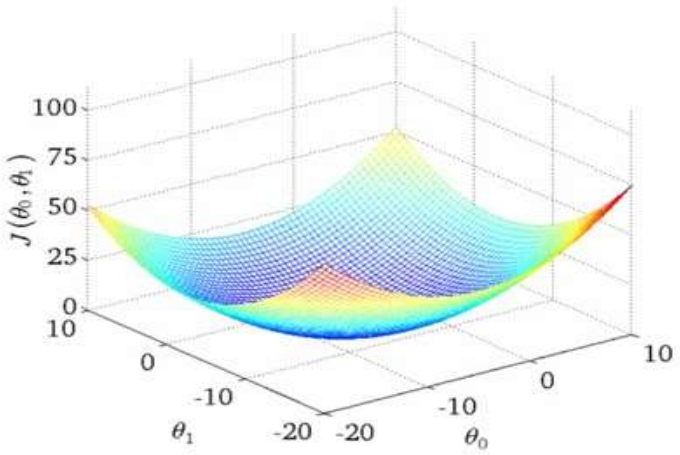


Figura 24 Função de perda (*Loss function*) de uma rede

Ao realizar o treinamento de uma rede, o objetivo é encontrar os valores de θ_0 e θ_1, no qual a função de perda seja mínima. Como os pesos sinápticos de uma rede são inicializados de forma aleatória, um possível caminho seria o mostrado na Figura 25 a seguir, onde os pesos da rede são iniciados aleatoriamente no ponto A e segue, através do gradiente descendente (ou seja, sempre no sentido oposto ao sentido de crescimento máximo), até o ponto B (ponto de mínimo).

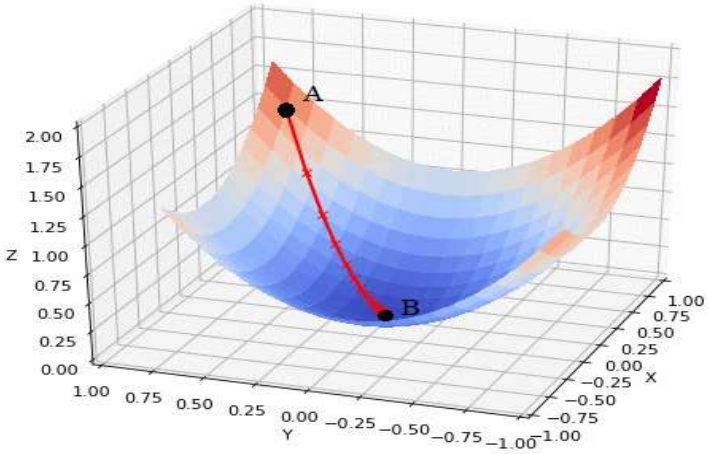


Figura 25 Trajetória percorrida pelos pesos sinápticos desde sua inicialização aleatória até alcançar o mínimo da função de perda

Através do gradiente da função nós decidimos para qual sentido devemos caminhar com a atualização dos pesos sinápticos da rede, falta decidir o tamanho do "passo" que será dado entre duas atualizações. Esse "passo" é determinado pela taxa de aprendizagem da rede. A escolha da taxa de aprendizagem é uma etapa muito importante em treinamento de CNN, pois se escolhermos um valor muito alto podemos ultrapassar o mínimo da função de perda e continuar saltando através das adjacências, como mostrado na Figura 26 , sem nunca chegarmos no mínimo. Por outro lado, se escolhermos valores de taxa de aprendizagem muito pequenas podemos tornar o processo de aprendizagem extremamente lento.

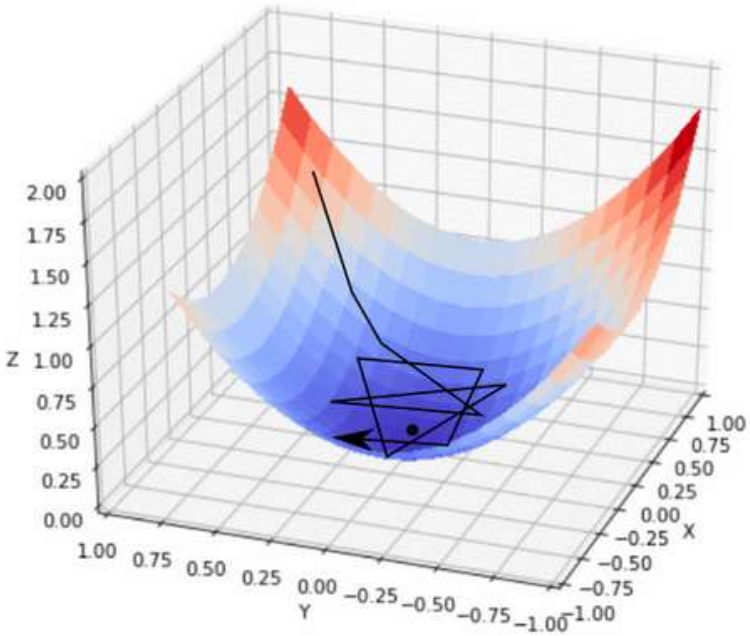


Figura 26 Trajetória percorrida pelos pesos sinápticos utilizando uma taxa de aprendizado grande

Dois problemas surgem quando usamos o método do gradiente descendente. Podemos ficar preso em um mínimo local ou podemos ficar preso em um ponto de sela (*Saddle point*), pois nesses dois pontos o gradiente é nulo, ou seja, não temos atualização dos pesos sinápticos da rede e consequentemente a rede entende que encontrou os valores ótimos de pesos. Um ponto de mínimo local e um ponto de sela pode ser visto na Figura 27.

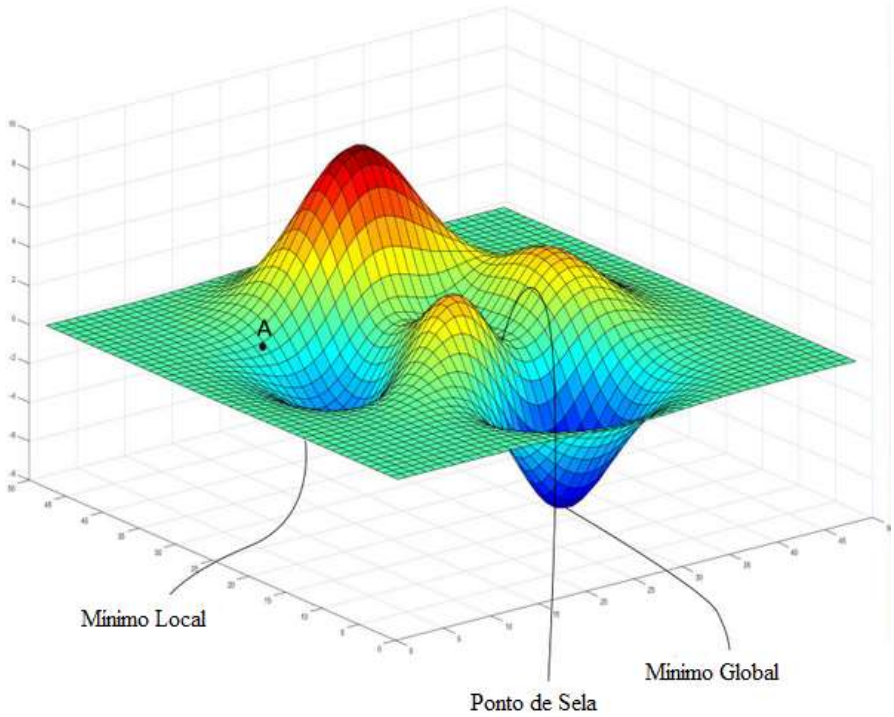


Figura 27 Ilustração de um ponto de mínimo local, global e um ponto de sela

Uma maneira de ajudar o gradiente escapar desses pontos é utilizando o Gradiente Descendente Estocástico (*Stochastic Gradient Descent – SGD*).

No gradiente descendente tradicional todos os padrões são processados em um único lote para formar a função de perda, já no SGD lotes menores (*mini-batch*) são estocasticamente escolhidos para achar o gradiente da função de perda apenas daquele lote.

Para otimizar o treinamento da rede podemos utilizar algoritmos com taxa de aprendizagem adaptativa. Serão descritos os três algoritmos que foram utilizados nesse trabalho: Gradiente descendente Estocástico com *Momentum* (*Stochastic Gradient Descent with Momentum– SGDM)*, Propagação de Raiz Quadrada Média (RMSProp) e Estimação de Momento Adaptativo (ADAM).

O SGDM adiciona um novo parâmetro que permite controlar a velocidade das mudanças nos pesos sinápticos da rede. Nesse método, em vez de usar apenas o gradiente da

etapa atual para saber a direção de atualização, o *Momentum* também acumula o gradiente de etapas anteriores para determinar a direção a seguir. Formalmente, tem-se:

$$W(t+1) = W(t) + \alpha\big(W(t) - W(t-1)\big) + \eta\nabla J(W(t)) \tag{36}$$

onde α é definida como taxa de *momentum*, $J(W)$ é a perda calculada para um determinado mini-batch e η é a taxa de aprendizagem da rede.

O algoritmo SGDM usa uma única taxa de aprendizado para todos os parâmetros treináveis da CNN. Outros algoritmos de otimização tentam melhorar o treinamento da rede usando taxas de aprendizado que são diferentes para diferentes parâmetros e que podem se adaptar automaticamente à função de perda que está sendo otimizada. O algoritmo RMSProp é um desses algoritmos. Ele mantém uma média móvel dos quadrados elementares dos gradientes de parâmetro,

$$\boldsymbol{v}(\boldsymbol{t}) = \beta_2.\boldsymbol{v}(\boldsymbol{t}-\mathbf{1}) + (1-\beta_2).[\nabla J(W(t))]^2 \tag{37}$$

onde, β_2 é a taxa de decaimento da média móvel e $\boldsymbol{v}(\boldsymbol{t})$ é o vetor das médias dos quadrados dos gradientes no tempo t. Valores comuns da taxa de decaimento são $0{,}9; 0{,}99\ e\ 0{,}999$. Os comprimentos de média móvel correspondentes dos gradientes quadrados são iguais a

$$\frac{1}{1-\beta_2} \tag{38}$$

isto é, $10, 100\ e\ 1000$ atualizações de parâmetros, respectivamente. A atualização dos pesos da rede segue a fórmula:

$$W(t+1) = W(t) - \frac{\eta}{\sqrt{\boldsymbol{v}(\boldsymbol{t})} + \varepsilon}\nabla J(W(t)) \tag{39}$$

O algoritmo de otimização ADAM utiliza uma atualização de parâmetros que é semelhante a RMSProp, mas com um termo de *momentum* adicionado. Esse algoritmo mantém uma média móvel de elementos dos gradientes dos parâmetros e seus valores quadrados,

$$\boldsymbol{m}(\boldsymbol{t}) = \beta_1 . \boldsymbol{m}(\boldsymbol{t}-\mathbf{1}) + (1-\beta_1)\nabla J(W(t)) \tag{40}$$

$$\boldsymbol{v}(\boldsymbol{t}) = \beta_2 . \boldsymbol{v}(\boldsymbol{t}-\mathbf{1}) + (1-\beta_2).[\nabla J(W(t))]^2 \tag{41}$$

onde: $\boldsymbol{m}(\boldsymbol{t})$ é o vetor das médias dos gradientes no tempo t, $\boldsymbol{v}(\boldsymbol{t})$ é o vetor das médias dos quadrados dos gradientes no tempo t, $\nabla J(W(t))$ é o gradiente da função de custo, β_1 é o peso atribuído à média dos gradientes (seu valor sugerido é 0,9) e β_2 é o peso atribuído à média dos quadrados dos gradientes (seu valor sugerido é 0,999).

No entanto, esses valores de $\boldsymbol{m}(\boldsymbol{t})$ e $\boldsymbol{v}(\boldsymbol{t})$ são enviesados e tendem a mover a solução em direção ao zero, principalmente no início e quando os valores de β_1 e β_2 são próximos de 1. A razão para isso é que os passos iniciais tendem a direcionar $\boldsymbol{m}(\boldsymbol{t})$ e $\boldsymbol{v}(\boldsymbol{t})$ para $\boldsymbol{m}(\boldsymbol{t}-\mathbf{1})$ e $\boldsymbol{v}(\boldsymbol{t}-\mathbf{1})$ (inicialmente nulos) devido aos altos valores de β_1 e β_2. Isso torna o crescimento de $\boldsymbol{m}(\boldsymbol{t})$ e $\boldsymbol{v}(\boldsymbol{t})$ lento. Para evitar isso, esses valores são corrigidos de acordo com as equações:

$$\boldsymbol{m}(\boldsymbol{t})_{\boldsymbol{c}} = \frac{\boldsymbol{m}(\boldsymbol{t})}{1-\beta_1^t} \tag{42}$$

$$\boldsymbol{v}(\boldsymbol{t})_{\boldsymbol{c}} = \frac{\boldsymbol{v}(\boldsymbol{t})}{1-\beta_2^t} \tag{43}$$

onde: $\boldsymbol{m}(\boldsymbol{t})_{\boldsymbol{c}}$ e $\boldsymbol{v}(\boldsymbol{t})_{\boldsymbol{c}}$ são o vetor corrigido das médias dos gradientes e o vetor corrigido das médias dos quadrados dos gradientes, respectivamente, e β_1^t, β_2^t significa o peso elevado ao número de épocas. A atualização dos pesos da rede segue a fórmula:

$$W(t+1) = W(t) - \frac{\eta}{\sqrt{\boldsymbol{v}(\boldsymbol{t})_{\boldsymbol{c}}} + \varepsilon}\boldsymbol{m}(\boldsymbol{t})_{\boldsymbol{c}} \tag{44}$$

3.3.2 Métodos de Regularização

Nesse tópico serão apresentados os métodos de regularização utilizados nesse trabalho a fim de evitar o *overfiting,* são eles: *Dropout* e regularização L2.

Uma CNN pode possuir uma quantidade de parâmetros ajustáveis (pesos sinápticos) na ordem de milhões, isso faz com que uma CNN seja capaz de descrever quase qualquer conjunto de dados de tamanho determinado. No entanto, mesmo que o modelo obtido pela rede

esteja de acordo com os dados disponíveis, isso não o torno necessariamente um bom modelo. Isso pode significar apenas que a rede funciona bem para os dados existentes, mas não conseguirá generalizar para novas situações. O verdadeiro teste para saber se um modelo de rede realmente generalizou é observar o desempenho da rede frente a situações ao qual ela nunca foi apresentada.

A situação descrita no parágrafo anterior chamamos de *Overfiting*, que é quando a rede se adapta muito bem aos dados com os quais foi treinada, porém não generaliza bem para novos dados. Uma das causas para o *overfiting* é quando temos poucos padrões para realizar o treinamento da rede, sendo assim a rede "decora" os padrões ao invés de aprendê-los. No presente trabalho o conjunto de dados contém 247 imagens, sendo 50% (123 imagens) utilizadas para realizar o treinamento. Devido ao baixo número de imagens para treinamento certamente o problema de *overfiting* se faz presente.

Para evitarmos o *overfiting* é realizado a chamada Regularização, e dentre os métodos existentes utilizamos *Dropout* e Regularização L2, bem como os dois atuando conjuntamente, que aqui chamaremos de *Dropout*+L2.

Conforme visto na Seção 3.2.6, *Dropout* consiste em uma camada acrescentada à rede, onde durante o seu processo de treinamento existe uma probabilidade p de um neurônio não ser desativado.

Por outro lado, a técnica de Regularização L2 (também conhecida como decaimento de peso) consiste em adicionar um termo extra à função de custo que deve ser minimizada da rede como forma de penalização do termo. A função de custo para a rede neural da Figura 9 foi apresentada na Seção 3.1.4:

$$E(k) = \frac{1}{2}\sum_{j=1}^{n_3}\left(d_j(k) - Y_j^3(k)\right)^2 \tag{45}$$

Ao adicionar o termo extra a função fica:

$$E(k) = \frac{1}{2}\sum_{j=1}^{n_3}\left(d_j(k) - Y_j^3(k)\right)^2 + \frac{\lambda}{2p}\sum_{w} w^2 \tag{46}$$

onde λ é o parâmetro de regularização da rede neural e p é o número de padrões disponíveis para o treinamento da rede.

3.4 SEGMENTAÇÃO SEMÂNTICA

A segmentação é essencial para tarefas de análise de imagens. A segmentação semântica descreve o processo de associar cada pixel de uma imagem a um rótulo de classe (como ‘flor’, ‘pessoa’, ‘estrada’, ‘céu’, oceano’ ou ‘carro’). A Figura 28 mostra um exemplo de uma rede utilizada para realizar a segmentação semântica.

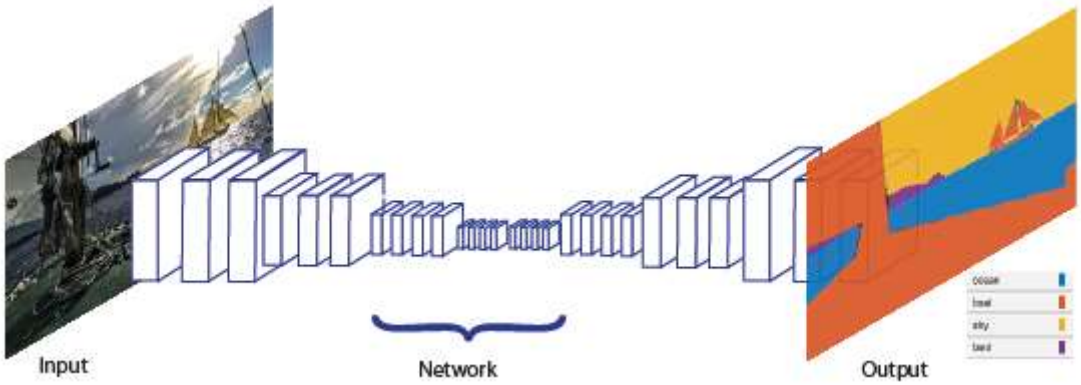


Figura 28 Exemplo de uma CNN utilizada para realizar segmentação semântica

Conforme pode ser observado esse tipo de arquitetura faz uso tanto da etapa de sub-amostragem como da etapa de sobre-amostragem, sendo elas, nesse caso mostrado, espelhamento uma da outra.

Diversas são as aplicações para segmentação semântica, as quais podemos destacar:

1. Inspeção industrial;
2. Classificação de terreno visível em imagens de satélites;
3. Análise de imagens médicas.

Nesse trabalho três propostas de rede semânticas foram investigadas, todas com o objetivo de realizar a segmentação da região pulmonar, ou seja, estaremos interessados em rotular os pixels da imagem em duas classes: ‘*lung*’ ou ‘*background*’.

4 METODOLOGIA

Na Figura 29 mostra-se um diagrama em blocos da metodologia utilizada para a realização da pesquisa proposta, tendo em vista a segmentação da região do pulmão.

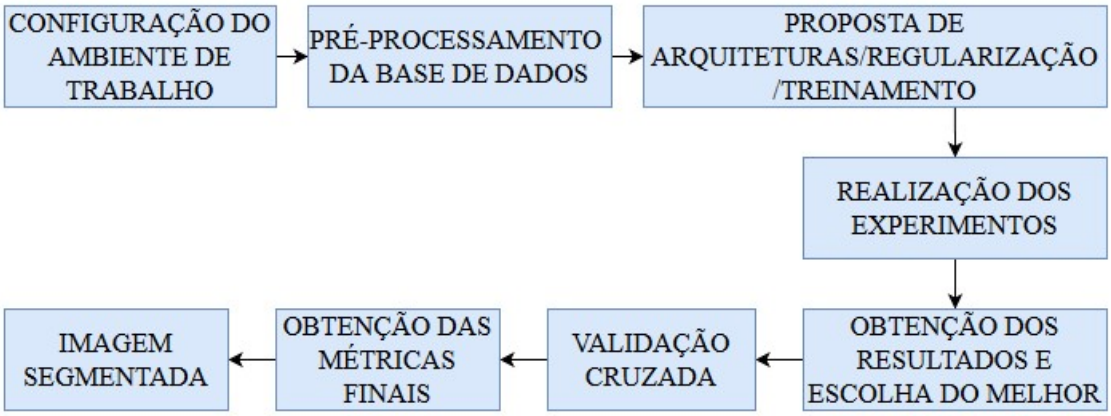


Figura 29 Diagrama de blocos da metodologia utilizada para realização da tarefa de segmentação da região do pulmão

4.1 CONFIGURAÇÃO DO AMBIENTE DE TRABALHO

Para este trabalho, foi utilizado um computador com sistema operacional Windows 10, um processador Intel Core i7-8700 CPU @ 3.20GHz 3.19GHz, 16GB de memória RAM e uma GPU NVIDIA GeForce GTX 1070 com 8GB de memória dedicada.

Para o desenvolvimento da pesquisa, pré-processamento da base de dados e simulação das redes convolutivas, foi utilizado o software MATLAB R2018b.

4.2 PRÉ-PROCESSAMENTO DA BASE DE DADOS

A base de dados utilizada nesse trabalho é a *JSRT - Japanese Society of Radiological Technology* (Shiraishi et al., 1999). Esta base contém 247 imagens de radiografia torácica, com seus respectivos padrões ouros de segmentação da região pulmonar, sendo que 154 imagens possuem um único nódulo e as outras 93 imagens não apresentam nódulo algum. Todas as imagens da base de dados estão em nível de cinza e possuem tamanho de 2048x2048 pixels. A Figura 30 mostra dois exemplos de imagens da base de dados, sendo uma com nódulo e a outra sem nódulo.

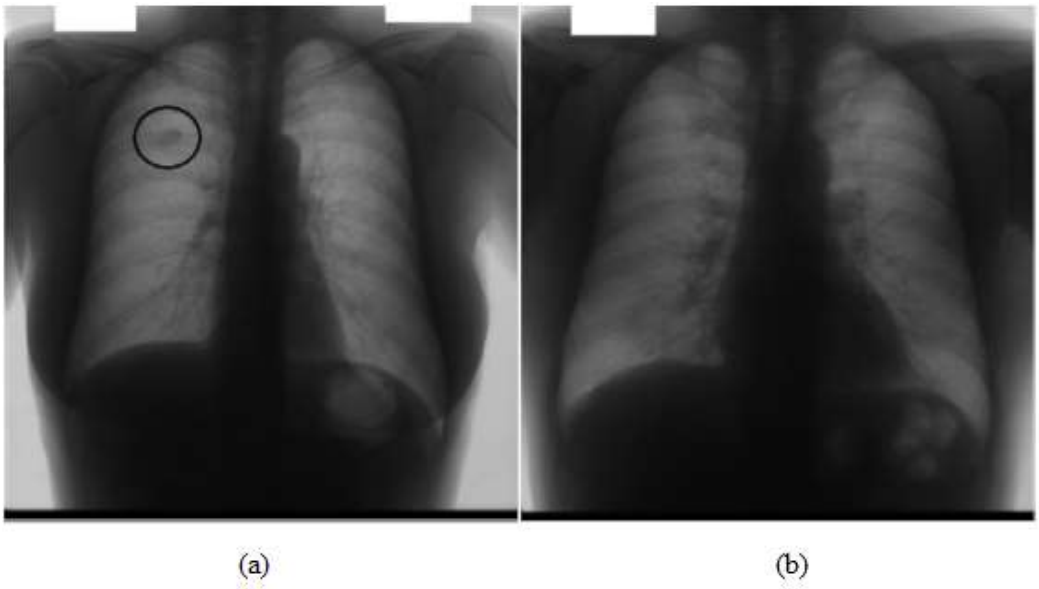


Figura 30 Exemplos de radiografias torácicas da base de dados. (a) Imagem com um nódulo (b) Imagem sem nódulo

O padrão ouro das imagens de radiografia apresenta separadamente os pulmões esquerdo e direito, de tal forma que, para realizar o treinamento da rede convolutiva, foi necessário realizar a soma das imagens contendo pulmão esquerdo com seu correspondente pulmão direito. A Figura 31 mostra os respectivos padrões ouros das imagens acima, após a realização da soma das imagens contendo o pulmão esquerdo e direito.

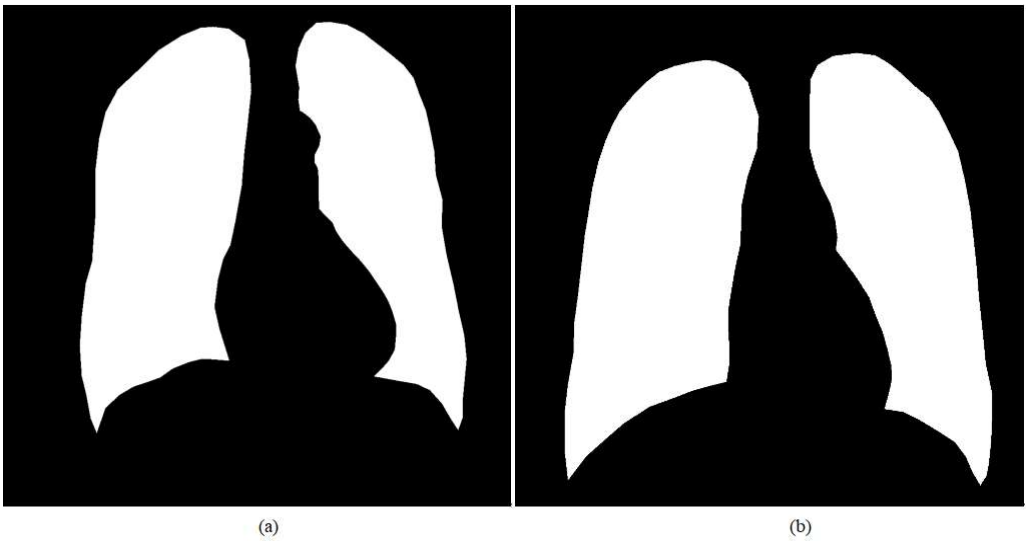


Figura 31 Exemplo de padrão ouro das imagens da figura anterior. (a) Padrão ouro da Figura 30(a). (b) Padrão ouro da Figura 30(b)

Por questões de limitação de memória do hardware, foi necessário o redimensionamento de todas as imagens, bem como de seus respectivos padrões ouro, para

512x512 pixels. O redimensionamento das imagens foi realizado através da função *imresize()* do MATLAB.

A Figura 32 mostra o procedimento descrito acima, referente ao redimensionamento das imagens de entradas. A Figura 33 ilustra o mesmo procedimento para as imagens do padrão ouro.

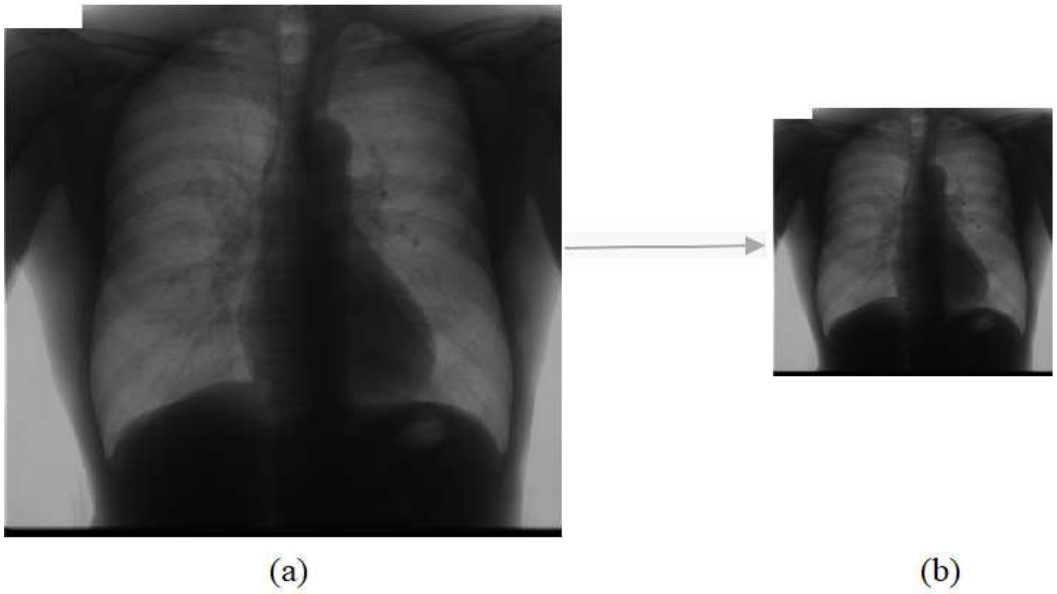


Figura 32 Ilustração do procedimento de redimensionamento das imagens de entrada da rede. (a) Imagem 2048x2048. (b) Imagem após o redimensionamento, em tamanho 512x512

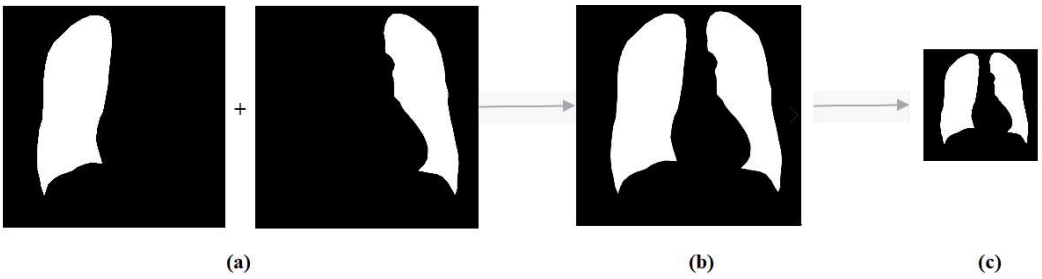


Figura 33 Ilustração do procedimento de soma e redimensionamento das imagens padrão ouro. (a) Imagem contendo o pulmão esquerdo e o direito, separadamente. (b) Imagem resultante da soma das duas imagens, em tamanho 2048x2048. (c) Imagem resultante do redimensionamento, em tamanho 512x512

Depois de redimensionadas, o próximo passo foi a divisão das imagens em 3 conjuntos: treinamento, validação e teste, na seguinte proporção: 50%, 25% e 25%, conforme a Figura 34.

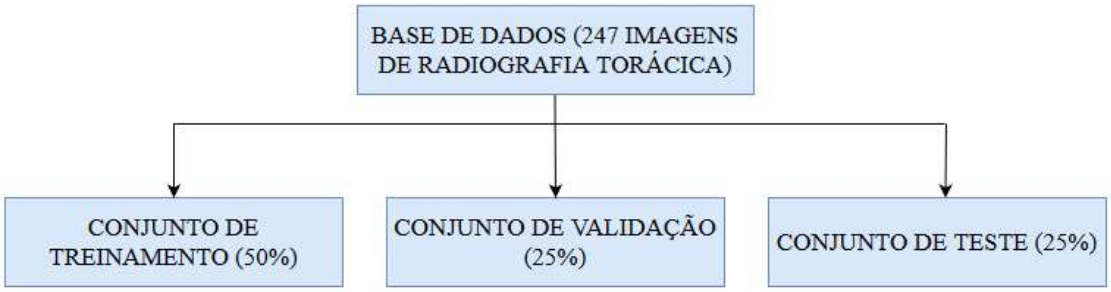


Figura 34 Divisão da base de dados em treinamento, validação e teste

Na tarefa de divisão das imagens em treinamento, validação e teste foram consideradas tanto imagens sem nódulo como imagens com nódulo em cada um dos conjuntos, mantendo-se a proporção, de forma a se manter a representatividade de ambos os tipos de imagens. Por exemplo, para a determinação do conjunto de treinamento foram consideradas 50% das imagens sem nódulo mais 50% das imagens com nódulo. Na Figura 35 está mostrado a divisão adotada para no procedimento.

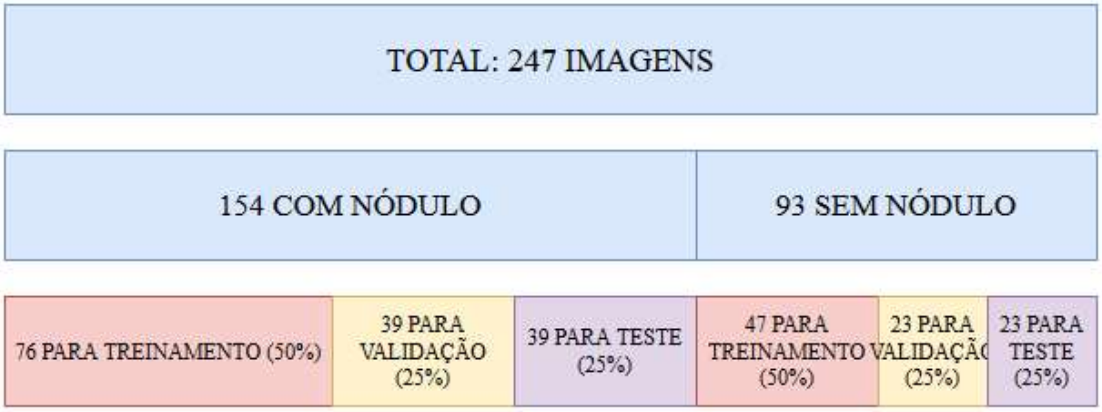


Figura 35 Procedimento para divisão da base de dados em treinamento, validação e teste

4.3 PROPOSTA DAS ARQUITETURAS DE REDES CONVOLUTIVAS

Para a tarefa de segmentação da região pulmonar de uma imagem torácica, são propostas 3 arquiteturas de redes convolutivas. Cada arquitetura foi investigada com os métodos de regularização L2 e *Dropout*, bem como com os dois métodos atuando simultaneamente. Também foi investigado a aplicação de três métodos distintos de otimização: Gradiente descendente estocástico, RMSPROP e ADAM. No total, foram realizadas 27 simulações para se chegar na melhor CNN para a tarefa. A Figura 36 ilustra o fato descrito.

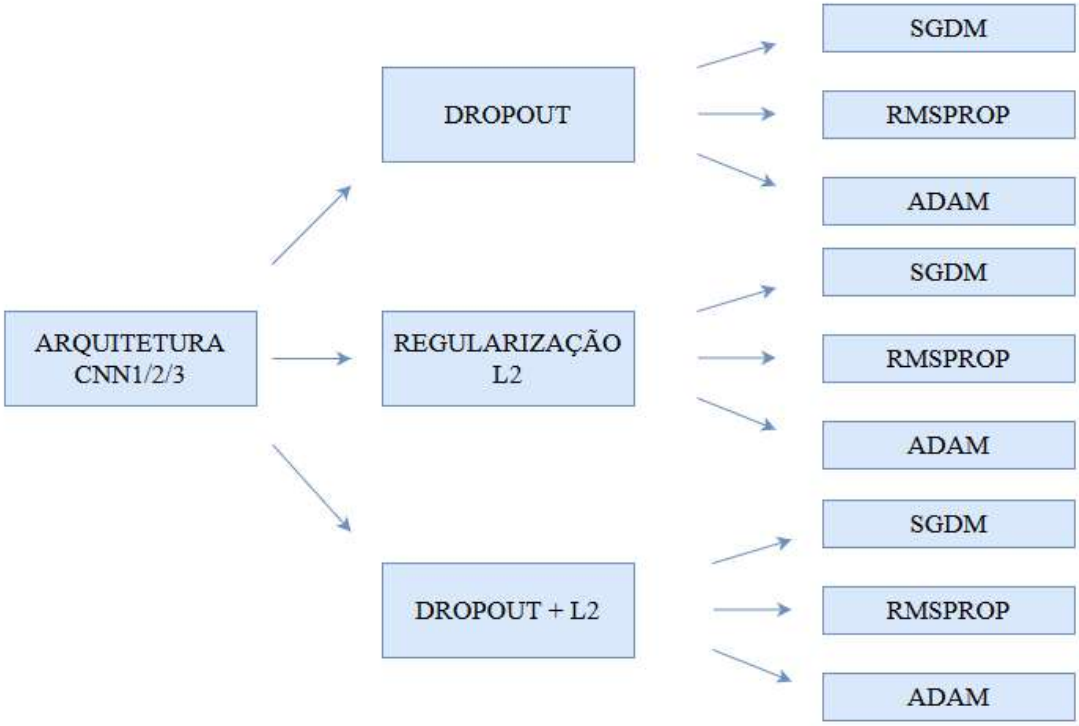


Figura 36 Metodologia para a escolha da melhor arquitetura, melhor método de regularização e melhor técnica de treinamento para realização da tarefa de segmentação pulmonar. A escolha será feita através do desempenho no conjunto de validação.

A associação entre configurações de redes, método de regularização e método de otimização será avaliada através do desempenho no conjunto de validação. Após ser obtida a combinação que resultar na melhor performance, o desempenho da mesma será avaliado através de validação cruzada de 5 pastas, conforme mostrado na Figura 37, utilizando o conjunto de treinamento e de teste. A partir dessa avaliação serão obtidos valores médios e valores de desvios padrões, para as 5 pastas e para cada métrica a ser utilizada. Na primeira etapa da validação cruzada 80% das imagens são usadas para realização do treinamento da rede (Pasta 1 – Pasta 4) e 20% das imagens usadas para realizar o teste da rede (Pasta 5). Esse procedimento se segue em mais 4 etapas (segunda etapa – quinta etapa), de tal forma que todas as imagens passem pelo conjunto de teste em apenas uma das etapas.

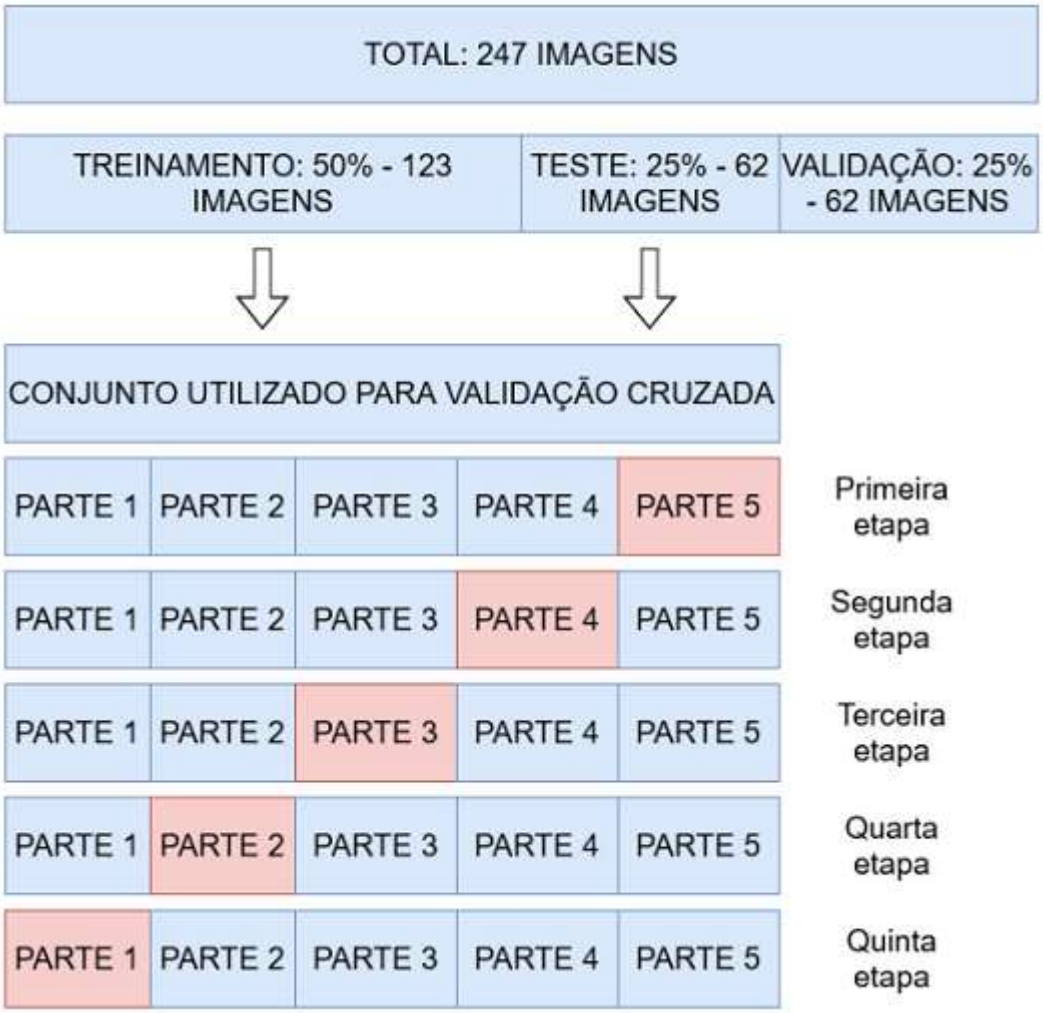


Figura 37 Processo de validação cruzada da rede

A primeira arquitetura (CNN-1) proposta é uma rede semântica, cujo objetivo é obter uma imagem de saída com dimensões idênticas à imagem de entrada, porém contendo a segmentação das regiões correspondentes aos pulmões esquerdo e direito. Essa rede é composta por 48 camadas. Como toda rede semântica, é composta de camadas de sub-amostragem seguidas de camadas de sobre-amostragem. Inicialmente, a rede contém 4 módulos de sub-amostragem. Cada módulo contém uma 2x (operação de convolução, mais uma operação *ReLu*, mais uma operação de *batch normalization*), seguidas da operação de *maxpooling*. Na primeira camada de sub-amostragem foram utilizados 32 mapas de características e nas demais camadas 64 mapas, sendo esse número de mapas obtidos através de testes para se chegar na melhor quantidade. Todos os filtros da camada de convolução são de tamanho 3x3 e utilizou-se preenchimento com zeros nas bordas. Todos os filtros utilizados na subamostragem são de tamanho 2x2 com passos horizontal e vertical igual a 2. Sendo assim, após passar pela camada

de convolução a imagem continuará com a dimensão de entrada e após passar pela camada de sub-amostragem seu tamanho será reduzido por um fator de 2 tanto na horizontal quanto na vertical. Em seguida seguem 4 camadas de sobre-amostragem, para que assim a imagem recupere informações perdidas nas etapas anteriores e possa retornar ao seu tamanho original. Em todas as etapas de sobre-amostragem serão utilizados filtros 4x4, passos horizontais e verticais igual a 2. Nas três primeiras camadas de sobre-amostragem usou-se 64 mapas e na quarta camada usou-se 32 mapas de características. Posteriormente, a imagem passará por uma convolução 1x1, onde seu volume passará a valer 2 (*lung* e *backgroung*). Por fim, a camada de classificação, através da função *Softmax*, classificará cada pixel da imagem como pertencendo ao pulmão (*lung*) ou ao plano de fundo da imagem (*background*). A CNN-1 está mostrada na Figura 38.

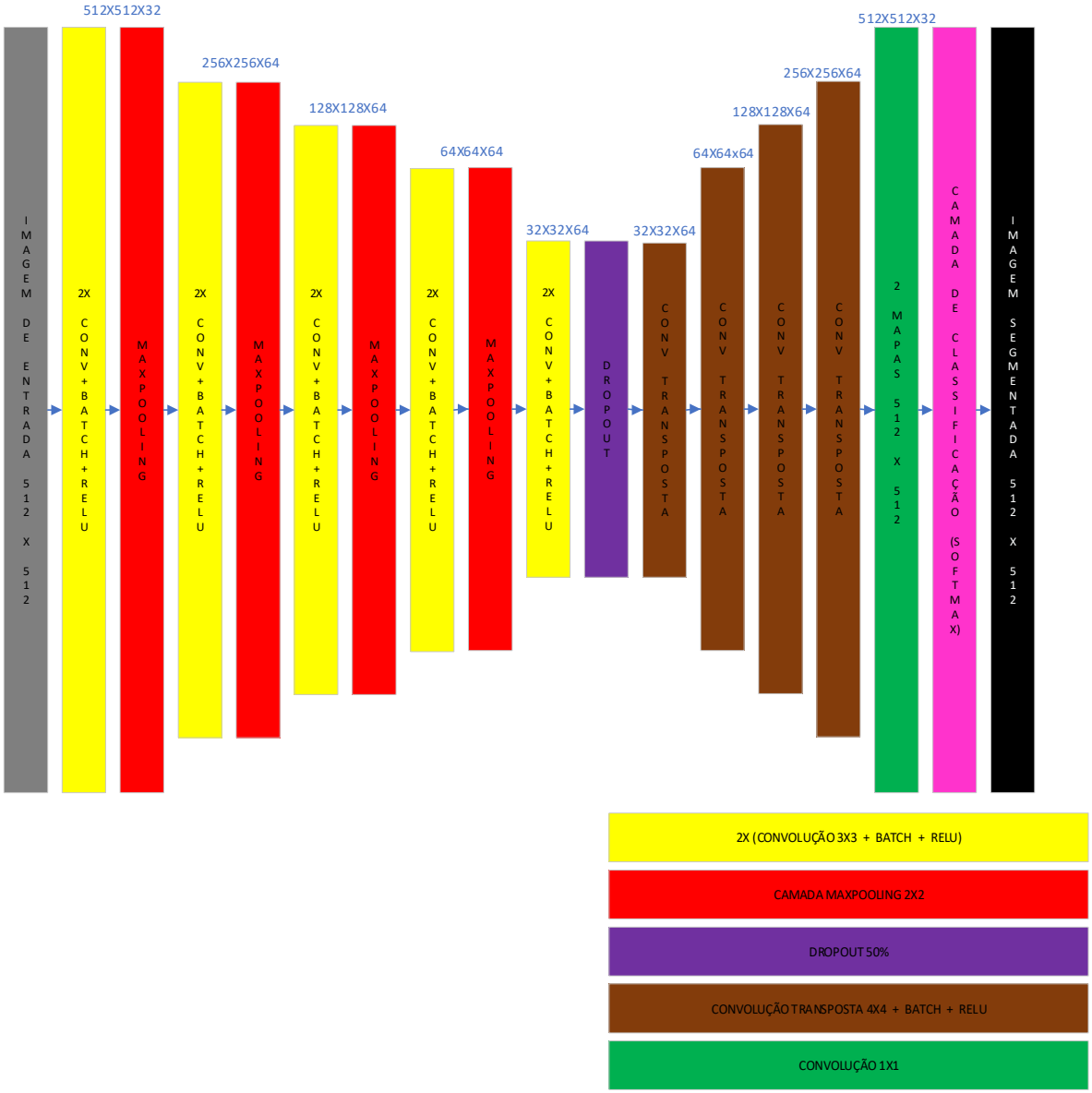


Figura 38 Primeira arquitetura (CNN-1) proposta – Rede Direta

A segunda arquitetura (CNN-2) também é uma rede direta, porém possui camadas de convolução, *Batch* e *ReLu* entre suas camadas de convolução transposta, totalizando 72 camadas em sua estrutura. As camadas *ReLu* adicionadas entre as etapas de convolução transposta devem conferir maior não-linearidade e capacidade de abstração durante cada etapa de sobre-amostragem. A CNN-2 está mostrada na Figura 39.

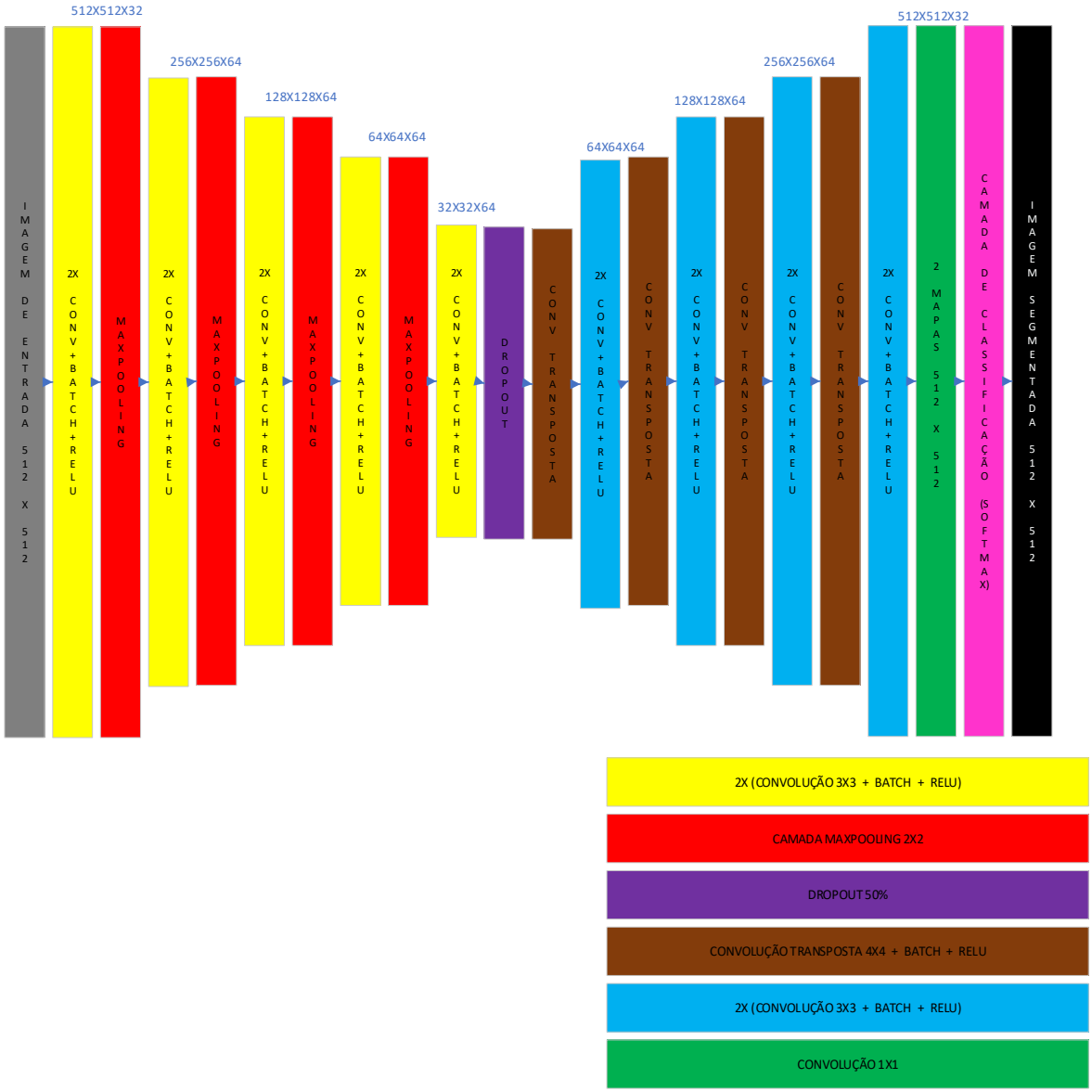


Figura 39 Segunda arquitetura (CNN-2) proposta – Rede Direta

A terceira arquitetura (CNN-3) é uma rede aplicando grafos acíclicos direcionados (DAG). Nesse tipo de arquitetura as informações das etapas iniciais, de sub-amostragem, são passadas mais adiante para as etapas finais, de sobre-amostragem. (Ronneberger, Fischer, & Brox, 2015) utilizam essa técnica para realizar segmentação em imagens médicas, enquanto

(Miyagawa, Costa, Gutierrez, Costa, & Filho, 2019) a utilizam para realizar detecção de bifurcação vascular em imagens IVOCT. As informações que são passadas das camadas iniciais são ligadas às informações contidas nas camadas finais de sobre-amostragem e, como as mesmas possuem as mesmas dimensionalidades, poderão ser conectadas sem que nenhum ajuste de tamanho se faça necessário. A arquitetura da CNN-3 está mostrada na Figura 40.

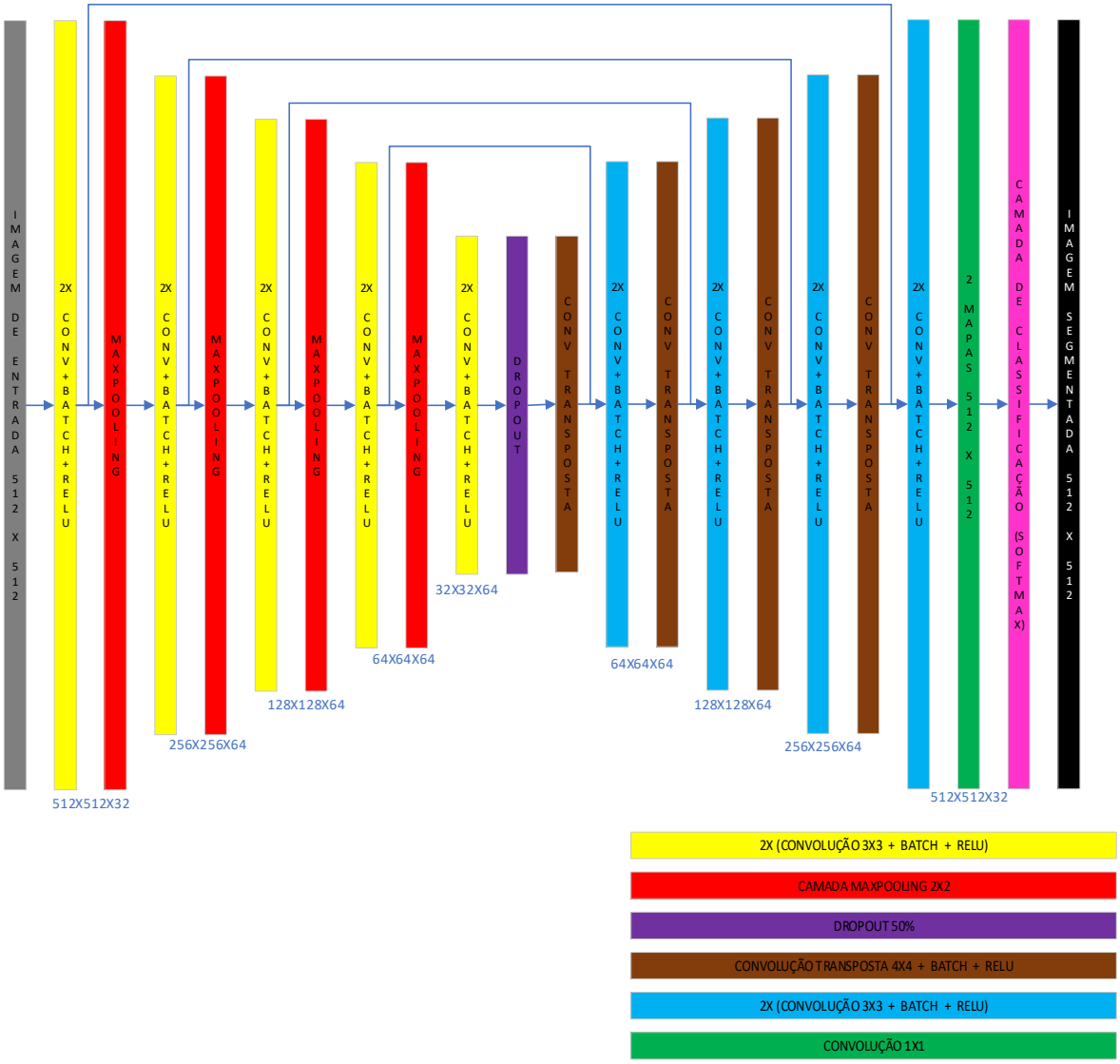


Figura 40 Terceira arquitetura (CNN-3) proposta – Grafo Acíclico Direto

4.4 EXPERIMENTOS

As três arquiteturas foram investigadas com três tipos de regularização e três tipos de algoritmos de otimização, utilizando sempre o mesmo conjunto de treinamento e validação. Os

valores dos parâmetros utilizados para o treinamento da rede foram ajustados experimentalmente. A Tabela 2 mostra os valores utilizados nos experimentos.

Tabela 2 Parâmetros utilizados na etapa de treinamento da rede

InitialLearnRate ($\boldsymbol{\eta}$)	10^{-3}
LearnRateDropFactor	0,1
LearnRateDropPeriod	30
MiniBatchSize	1
MaxEpochs	50
SquaredGradientDecayFactor (β_2)	0,999
GradientDecayFactor (β_1)	0,9
Momentum (α)	0,9
Epsilon (ε)	10^{-8}
L2Regularization (λ)	10^{-4}

Todos os experimentos serão realizados com imagens de entrada 512x512.

4.5 MÉTRICAS DE DESEMPENHO DAS REDES

A seguir definiremos as métricas que foram utilizadas para avaliação do desempenho das redes convolutivas. Essas métricas permitirão, não apenas comparar o desempenho das arquiteturas propostas nesse trabalho, como também a comparação do desempenho dessas arquiteturas com outros resultados obtidos na literatura.

A Figura 41 mostra o diagrama de Venn, onde o círculo em azul representa o padrão ouro e o círculo em amarelo representa a predição da CNN. Estabelecendo que os valores positivos dizem respeito ao pulmão e os negativos dizem respeito ao *background*, podemos definir o seguinte:

1. FN (Falso negativo) - são os pixels que fazem parte da região pulmonar, mas que foram erroneamente classificados pela CNN como *background*;
2. TP (Verdadeiro Positivo) – são os pixels que fazem parte da região pulmonar, e foram corretamente classificados pela CNN como pertencendo à região pulmonar;

3. FP (Falso Positivo) – são os pixels que fazem parte do *background*, mas que foram erroneamente classificados pela CNN como pertencendo à região pulmonar;
4. TN (Verdadeiro Negativo) – são os pixels que fazem parte do *background*, e foram corretamente classificados pela CNN como pertencendo ao background.

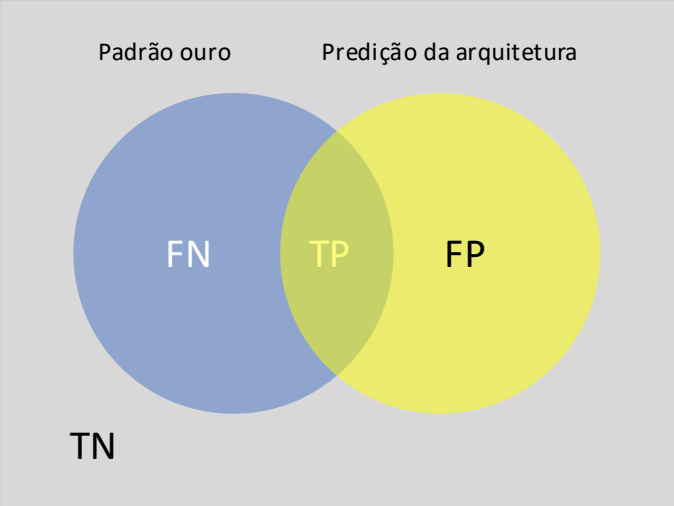


Figura 41 Diagrama de Venn

De acordo com as definições acima, espera-se que TP e TN possuam valores altos e FP e FN possuam valores baixos.

Sendo assim, as métricas utilizadas podem ser definidas como:

A acurácia indicará a proporção de pixels classificados corretamente em cada uma das classes do problema:

$$acurácia = \frac{\left(\frac{TP}{TP + FN}\right) + \left(\frac{TN}{TN + FP}\right)}{2} \tag{47}$$

A acurácia global indicará a proporção de pixels classificados corretamente, independentemente da classe pertencente:

$$acurácia\ global = \frac{TP + TN}{TN + FN + TP + FP} \tag{48}$$

A métrica IOU (*intersection over union*), ou coeficiente de Jaccard, penaliza a classificação incorreta dos pixels ou como pulmão (FP) ou como *background* (FN), sendo dada por:

$$pulmão = \frac{TP}{TP + FN + FP} \tag{49}$$

$$background = \frac{TN}{TN + FN + FP} \tag{50}$$

$$Jaccard\ (ou\ IOU) = \frac{pulmão + background}{2} \tag{51}$$

Pode-se utilizar também a IOU ponderada quando houver desproporcionalidade entre os tamanhos de classes da imagem.

$$peso^{pulmão} = \frac{número\ de\ pixels\ pertecentes\ à\ região\ pulmonar}{número\ total\ de\ pixels\ da\ imagem} \tag{52}$$

$$peso^{background} = \frac{número\ de\ pixels\ pertecentes\ ao\ background}{número\ total\ de\ pixels\ da\ imagem} \tag{53}$$

$$IOU\ ponderada = peso^{pulmão}.pulmão + peso^{background}.background \tag{54}$$

O Coeficiente Dice mede a proporção de pixels corretamente classificados como pertencentes à região pulmonar, penalizando a classificação incorreta:

$$coeficiente\ Dice = \frac{2TP}{2TP + FP + FN} \tag{55}$$

A métrica Score F1 (BF) indicará o quão bem as bordas de cada classe alinham-se com as respectivas bordas do padrão ouro:

$$BF = \frac{2.sensibilidade.especificidade}{sensibilidade + especificidade} \tag{56}$$

onde,

$$sensibilidade = \frac{TP}{TP + FN} \tag{57}$$

$$especificidade = \frac{TN}{TN + FP} \tag{58}$$

A matriz de confusão também é uma ferramenta importante referente ao desempenho de redes, pois ela permite visualizar o desempenho para cada classe no problema separadamente. Nela constam os quatro valores (FN, TP, FP e TN) discutidos anteriormente. Para o presente trabalho, constituído de duas classes (*lung* e *background*), um exemplo de uma matriz de confusão é apresentado na Figura 42:

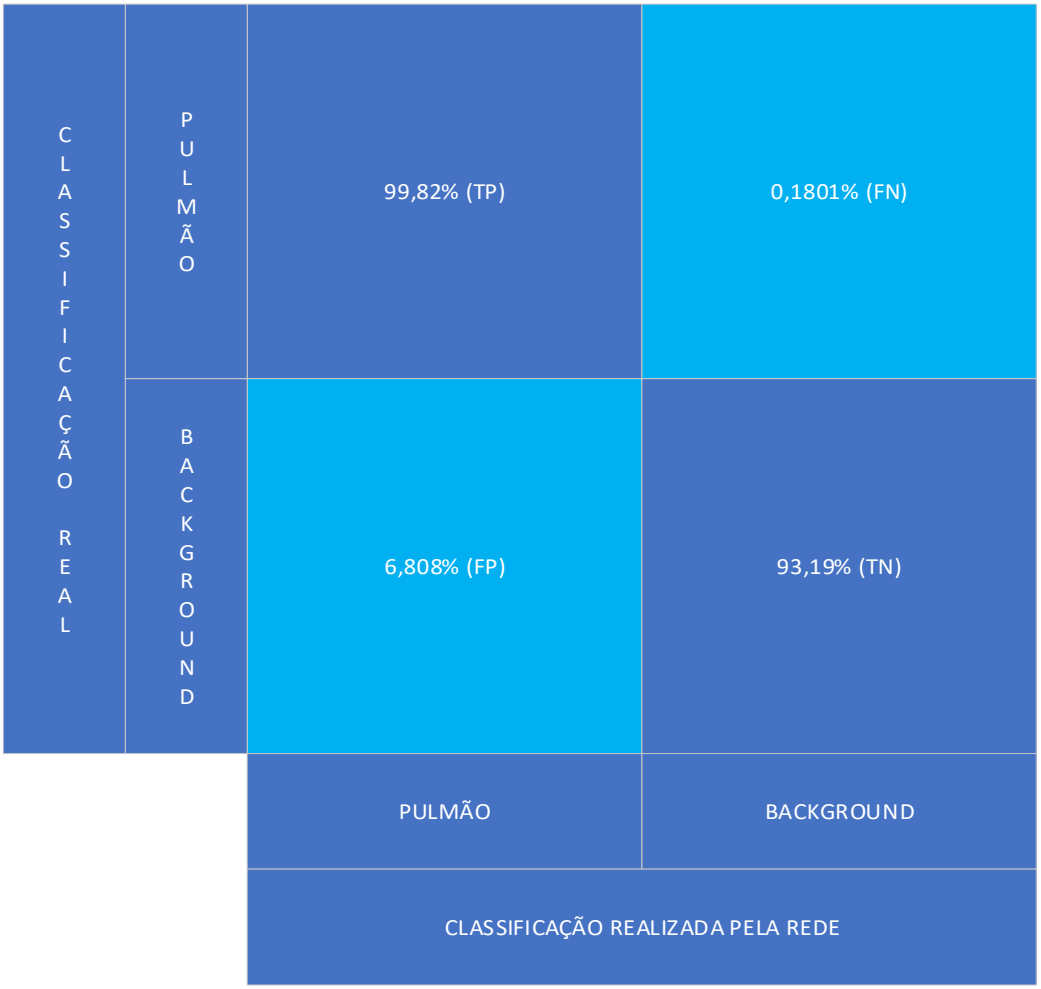


Figura 42 Exemplo de uma Matriz de confusão

As informações contidas nessa tabela são interpretadas da seguinte forma:

1. 99,82% dos pixels da região pulmonar (classe *lung*) foram corretamente classificados como sendo dessa classe (TP);

2. 0,1801% dos pixels da região pulmonar (classe *lung*) foram erroneamente classificados como sendo da classe *background* (FN);

3. 93,19% dos pixels da classe *background* foram corretamente classificados como sendo dessa classe (TN);

4. 6,808% dos pixels da classe *background* foram erroneamente classificados como sendo da classe *lung* (FP).

5 RESULTADOS OBTIDOS E DISCUSSÕES

Neste capítulo serão apresentados resultados obtidos para as métricas, bem como a discussão para a escolha da melhor arquitetura e sua comparação com o *benchmark*.

5.1 SIMULAÇÕES REALIZADAS COM A REDE CONVOLUTIVA CNN1

A primeira simulação foi realizada na arquitetura CNN1, utilizando o método de regularização *Dropout* e otimização SGDM. A Figura 43 mostra o comportamento da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

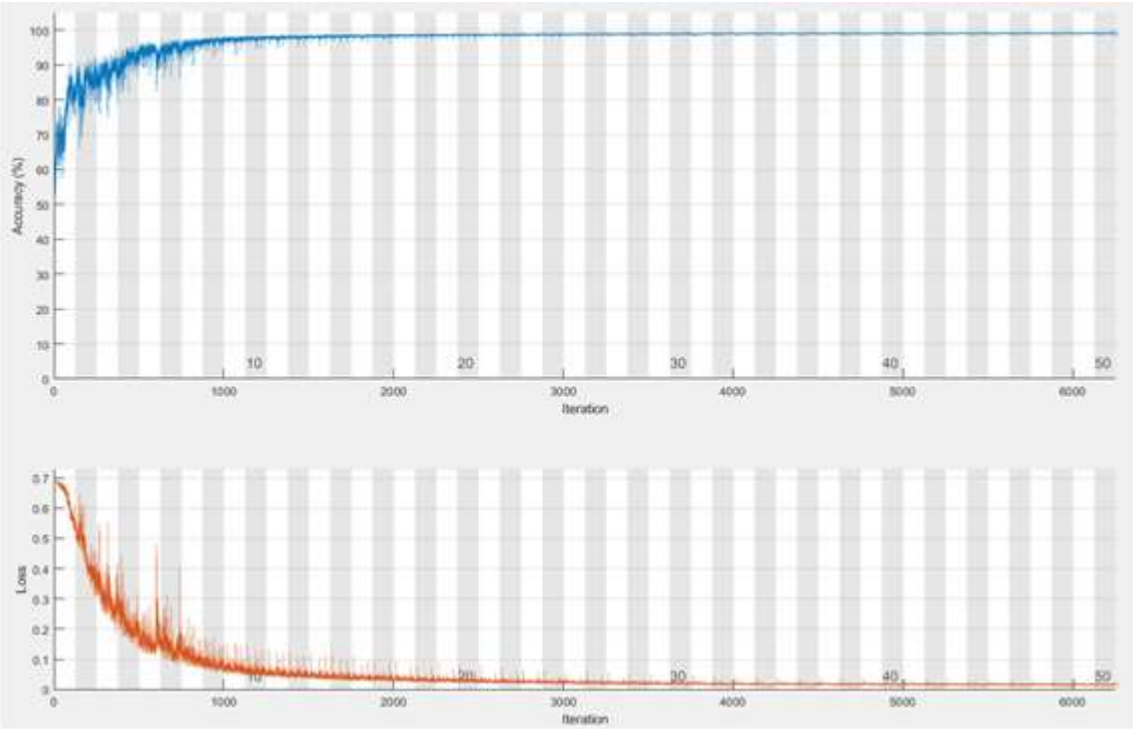


Figura 43 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização SGDM

Nota-se pelo gráfico da Acurácia que à medida que as iterações estão ocorrendo, seu valor tende a se aproximar de 100%, que seria a condição ideal. Pode-se perceber também a tendência da função de perda de se aproximar de zero à medida que as iterações acontecem, que seria também a condição ideal. Esse comportamento é observado também em todas as outras simulações.

A Tabela 3 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 43, durante a fase de validação, enquanto que a Tabela 4 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 3 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,93935	0,95539	0,86247	0,88047	0,52924	0,96375

Tabela 4 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,99861	0,81398	0,37867
Background	0,91217	0,91097	0,67981

Na Figura 44 está a matriz de confusão obtida pela rede.

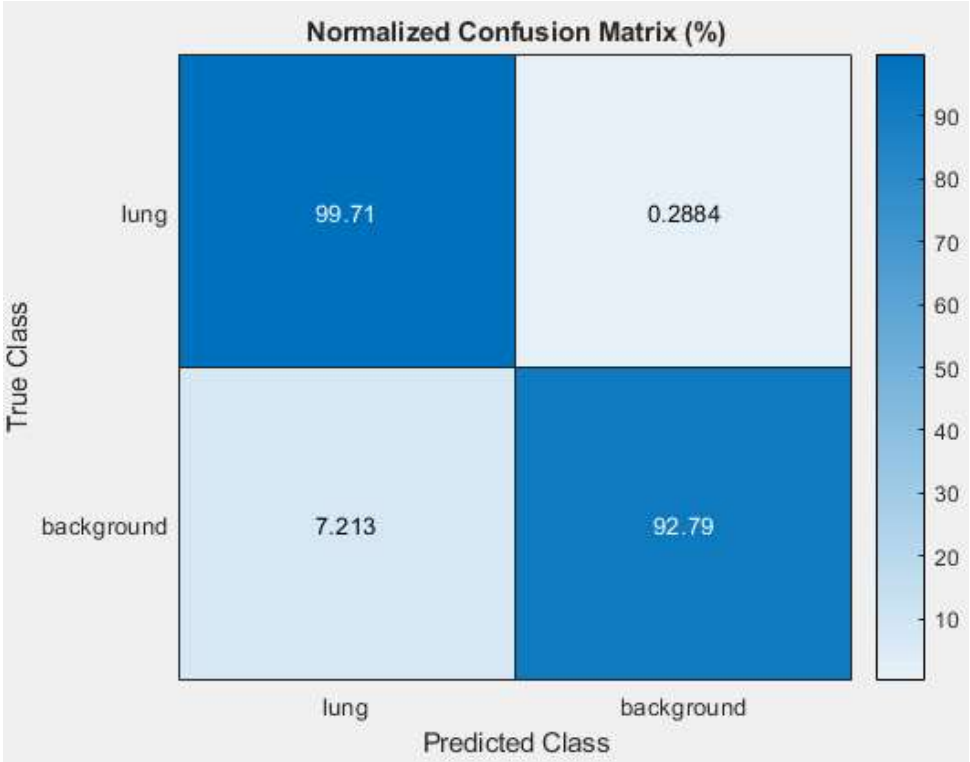


Figura 44 Matriz de confusão para arquitetura CNN1, utilizando regularização *Dropout* e otimização SGDM

A Figura 45 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 43.

Figura 45 Segmentação realizada pela arquitetura CNN1, utilizando regularização *Dropout* e otimização SGDM

O baixo valor da métrica Score F1 na Tabela 3 indica que a borda do padrão ouro não está bem alinhada com a borda da imagem obtida pela CNN. Na Tabela 4 o valor baixo para a acurácia do *Background* indica que muitos pixels pertencentes à região do *Background* foram erroneamente classificados como pertencentes à região do pulmão. Esse fato pode ser observado na Figura 45 pelos pontos mais escuros separados da região principal do pulmão.

A segunda simulação foi realizada na arquitetura CNN1, utilizando o método de regularização *Dropout* e otimização RMSProp. A Figura 46 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

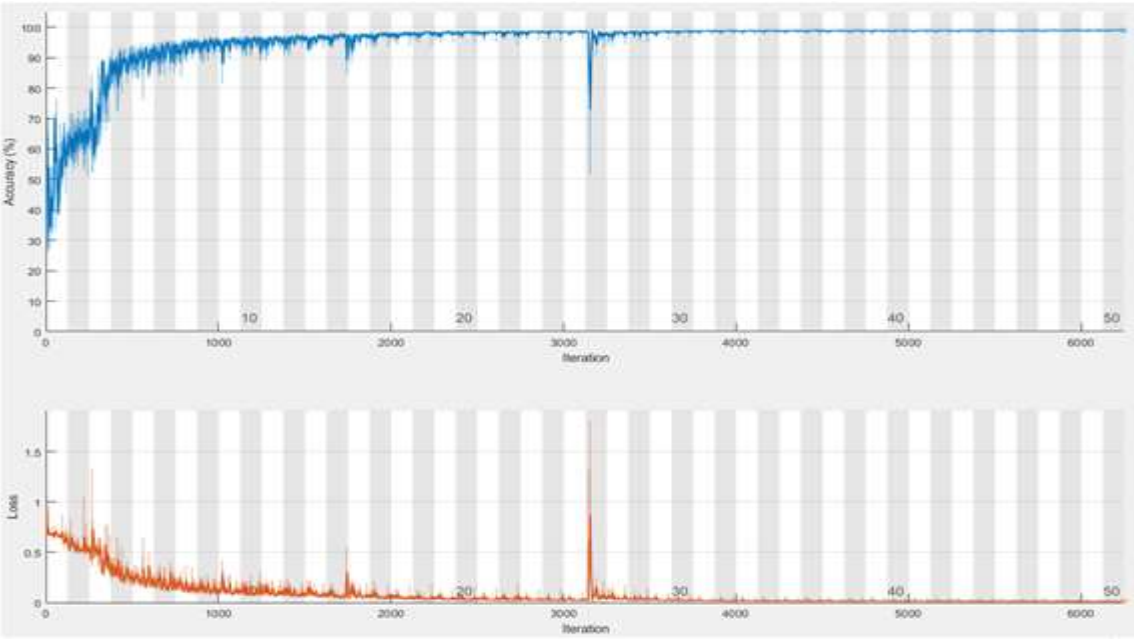


Figura 46 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização RMSProp

A Tabela 5 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 46 durante a fase de validação, enquanto que a Tabela 6 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 5 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização *Dropout* e otimização RMSProp, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98884	0,98704	0,97450	0,97797	0,96889	0,98698

Tabela 6 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98218	0,96513	0,95721
Background	0,99190	0,98386	0,98058

Na Figura 47 está a matriz de confusão obtida pela rede.

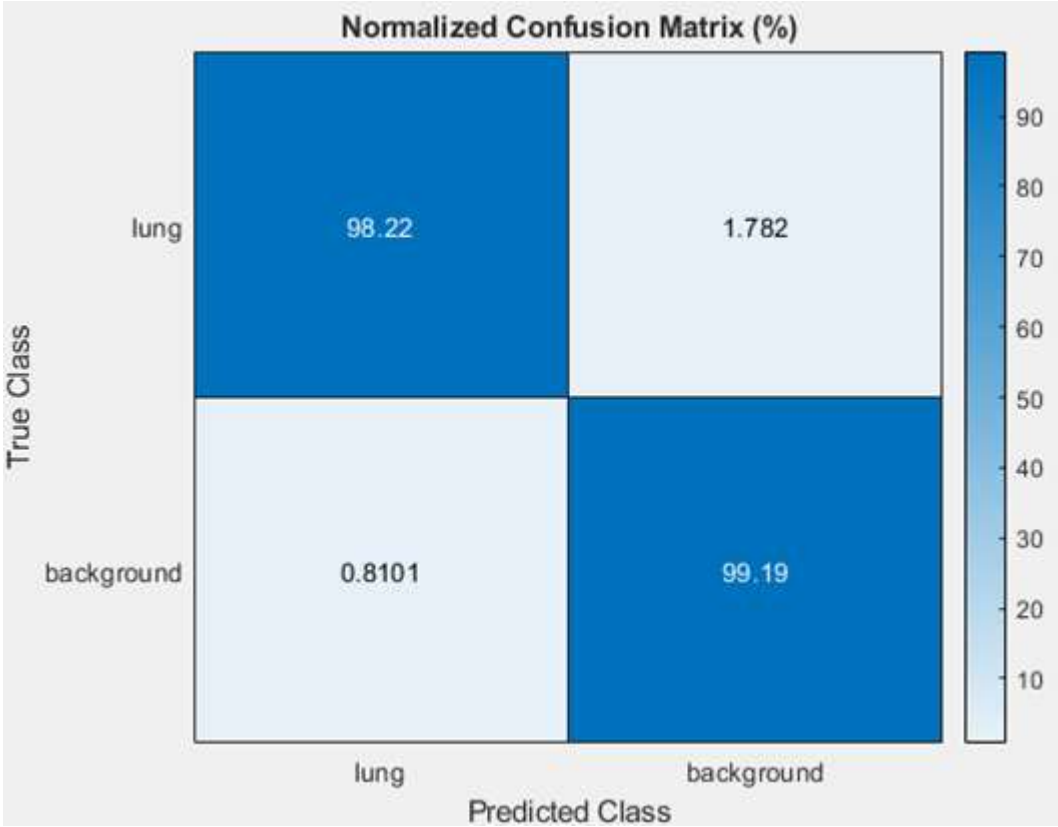


Figura 47 Matriz de confusão para arquitetura CNN1, utilizando regularização *Dropout* e otimização RMSProp

A Figura 48 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 46.

Figura 48 Segmentação realizada pela arquitetura CNN1, utilizando regularização *Dropout* e otimização RMSProp

Pode-se notar que mudando o método de otimização para RMSPROP, as métricas melhoraram bastante, bem como a imagem segmentada pela rede.

A terceira simulação foi realizada na arquitetura CNN1, utilizando o método de regularização *Dropout* e otimização ADAM. A Figura 49 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

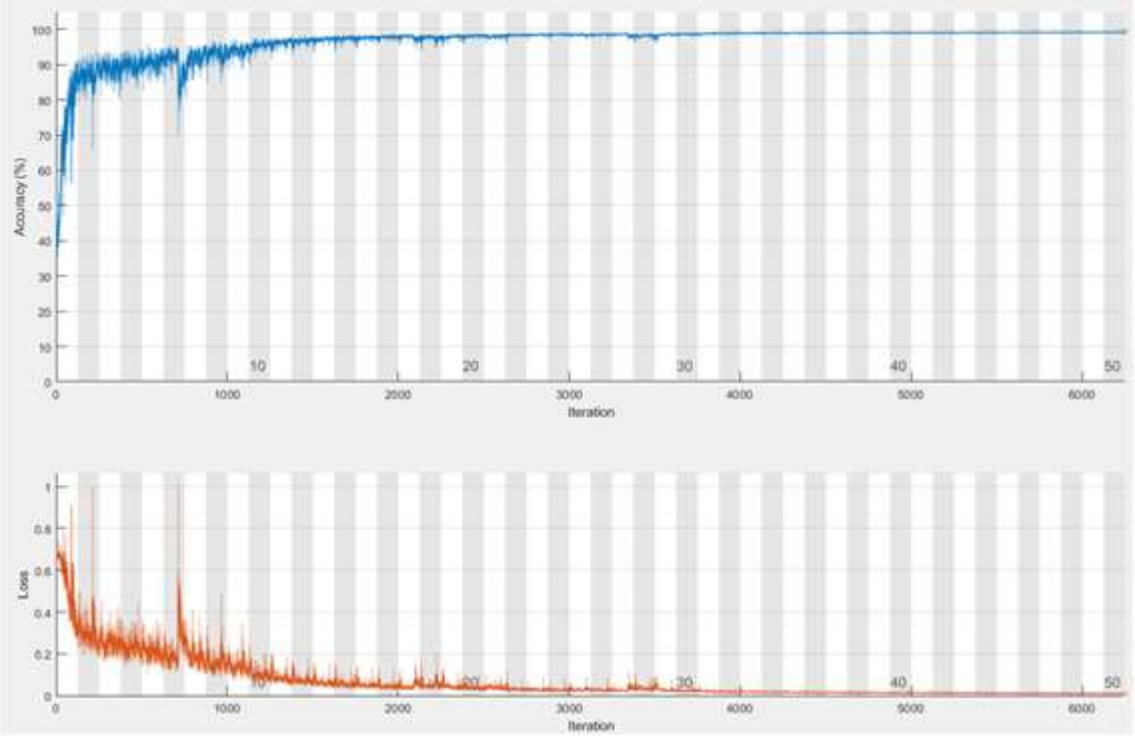


Figura 49 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização ADAM

A Tabela 7 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 49 durante a fase de validação, enquanto que a

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98763	0,98684	0,95836	0,96477	0,94156	0,98681

Tabela 8 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 7 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização *Dropout* e otimização ADAM, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98763	0,98684	0,95836	0,96477	0,94156	0,98681

Tabela 8 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização *Dropout* eotimização ADAM, na etapa de validação

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98470	0,94109	0,91736
Background	0,98897	0,97563	0,96576

Na Figura 50 está a matriz de confusão obtida pela rede.

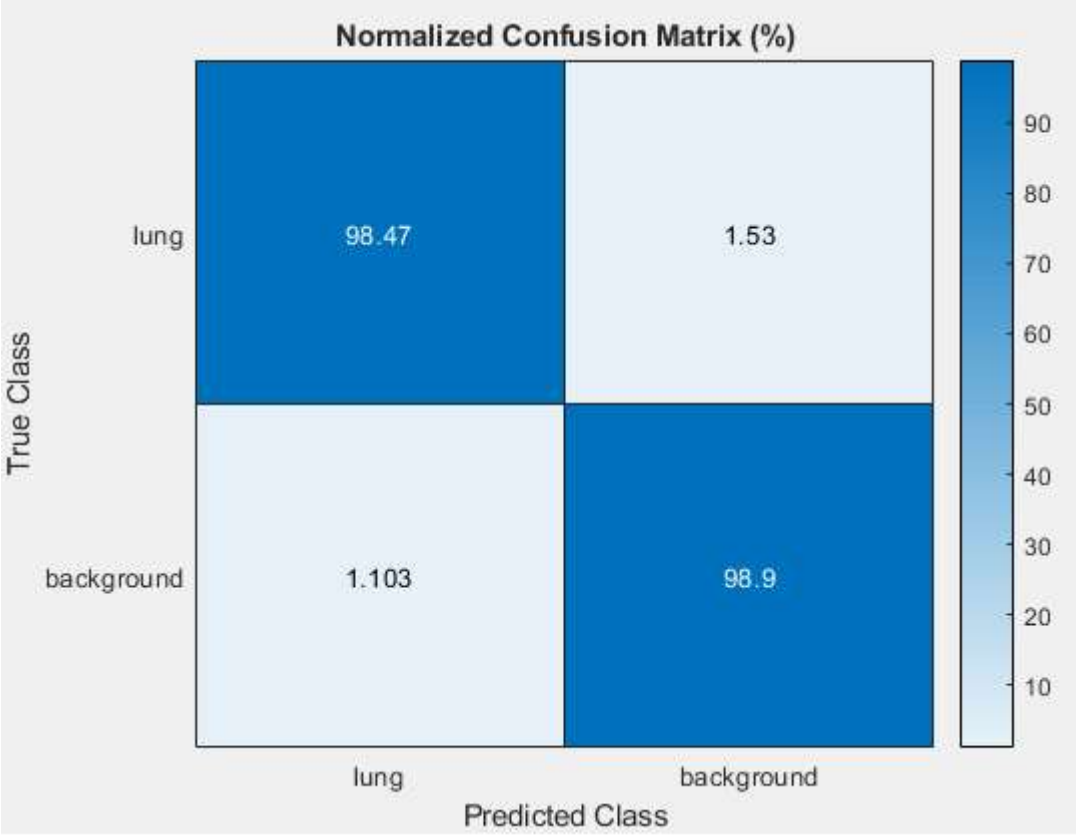


Figura 50 Matriz de confusão para arquitetura CNN1, utilizando regularização *Dropout* e otimização ADAM

A Figura 51 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostra na Figura 49.

Figura 51 Segmentação realizada pela arquitetura CNN1, utilizando regularização *Dropout* e otimização ADAM

A quarta simulação foi realizada na arquitetura CNN1, utilizando o método de regularização L2 e otimização SGDM. A Figura 52 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

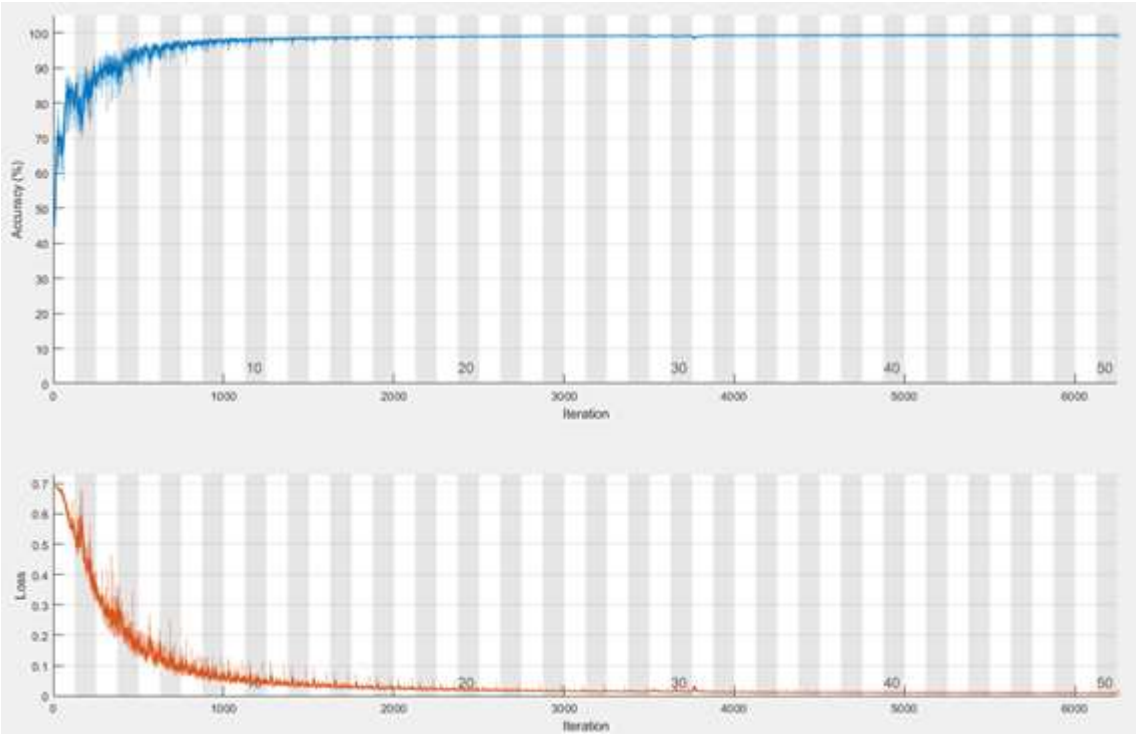


Figura 52 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização L2 e otimização SGDM

A Tabela 9 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 52 durante a fase de validação, enquanto que a Tabela 10 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 9 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização L2 e otimização SGDM, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98591	0,98426	0,95520	0,96143	0,91755	0,98419

Tabela 10 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização L2 e otimização SGDM, na etapa de validação

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,97983	0,93842	0,88707
Background	0,98869	0,97198	0,94803

Na Figura 53 está a matriz de confusão obtida pela rede.

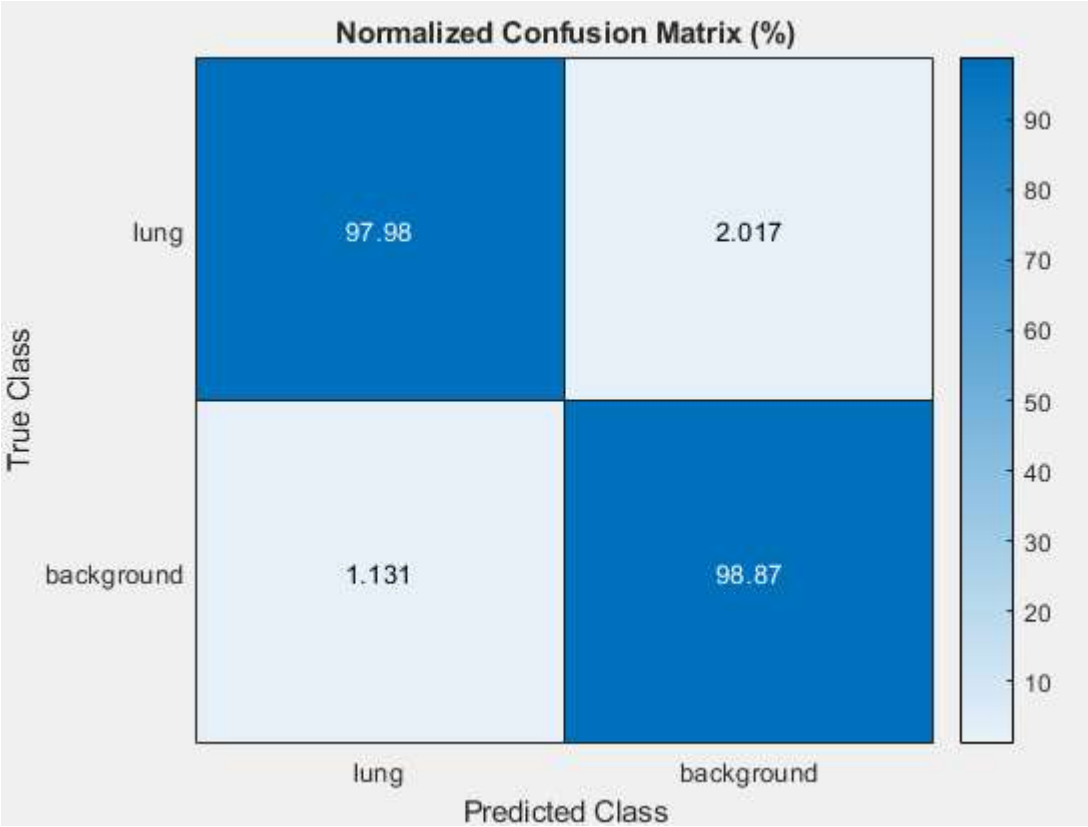


Figura 53 Matriz de confusão perda para arquitetura CNN1, utilizando regularização L2 e otimização SGDM

A Figura 54 apresenta um exemplo de segmentação realizado pela mesma CNN após cujo treinamento foi mostrado na Figura 52.

Figura 54 Segmentação realizada pela arquitetura CNN1, utilizando regularização L2 e otimização SGDM

Na Tabela 10, a métrica Score F1 possui um valor baixo para a classe Pulmão, e isso se refletiu na segmentação da rede, conforme visto na Figura 54, onde uma região de *Background* adjacente a região do pulmão foi equivocadamente classificada como pertencente a classe Pulmão.

A quinta simulação foi realizada na arquitetura CNN1, utilizando o método de regularização L2 e otimização RMSProp. A Figura 55 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

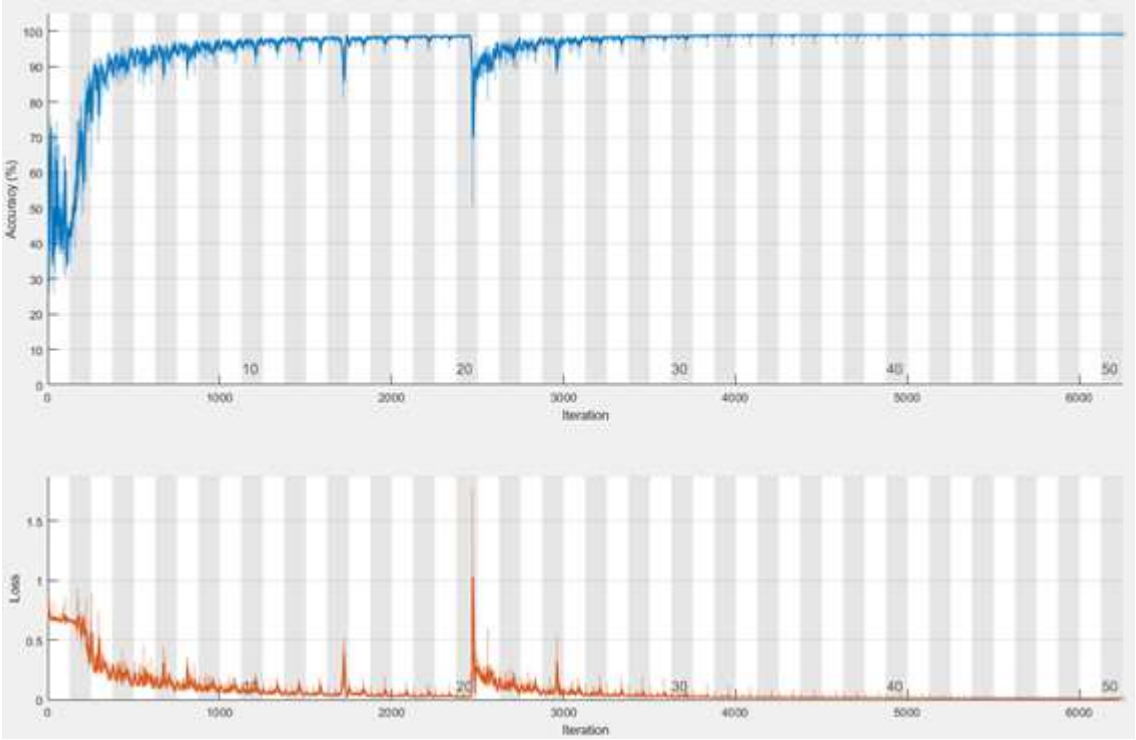


Figura 55 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização L2 e otimização RMSProp

A Tabela 11 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 55 durante a fase de validação, enquanto que a Tabela 12 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 11 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização L2 e otimização RMSProp, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98900	0,98732	0,97486	0,97828	0,96758	0,98726

Tabela 12 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização L2 eotimização RMSProp, na etapa de validação

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98287	0,96563	0,95634
Background	0,99186	0,98409	0,97881

Na Figura 56 está a matriz de confusão obtida pela rede.

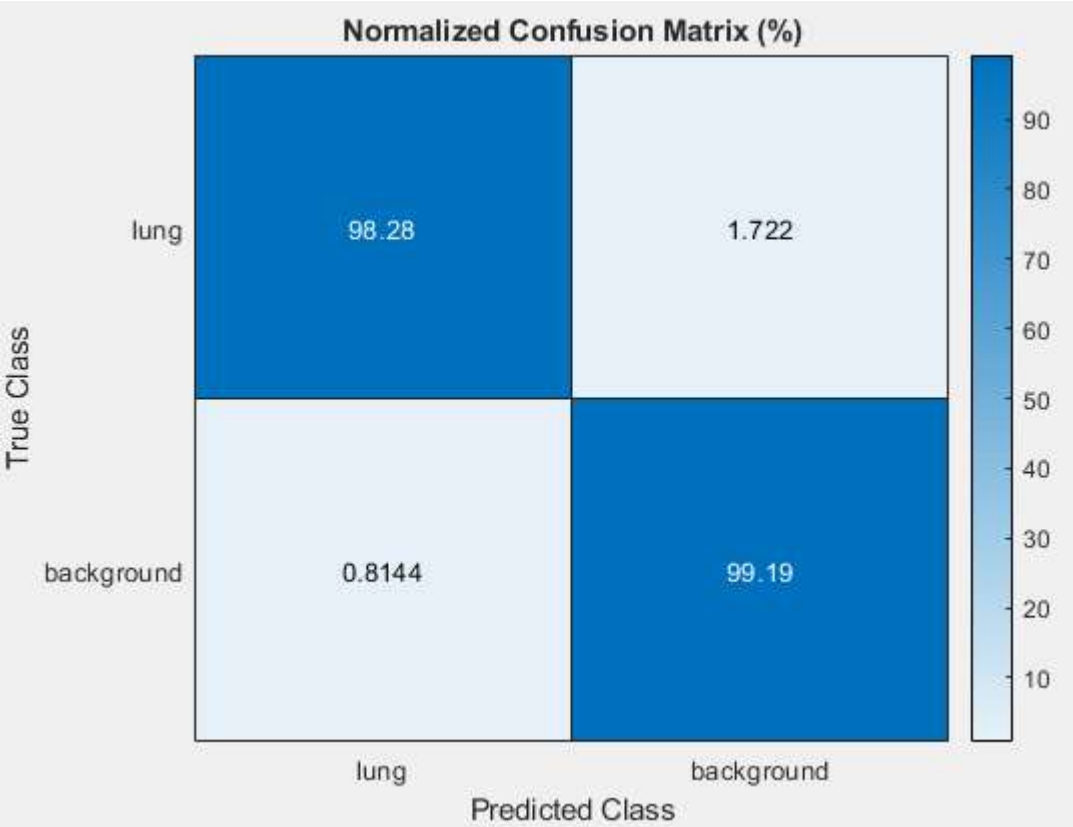


Figura 56 Matriz de confusão para arquitetura CNN1, utilizando regularização L2 e otimização RMSProp

A Figura 57 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 55.

Figura 57 Segmentação realizada pela arquitetura CNN1, utilizando regularização L2 e otimização RMSProp

A sexta simulação foi realizada na arquitetura CNN1, utilizando o método de regularização L2 e otimização ADAM. A Figura 58 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

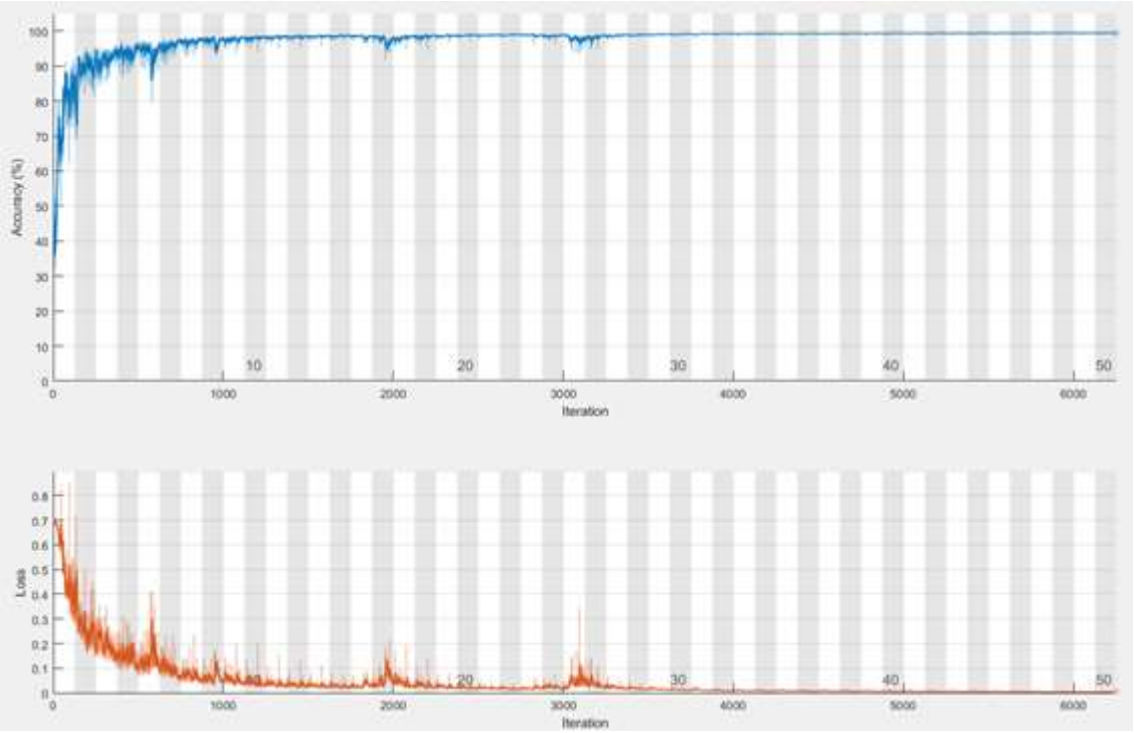


Figura 58 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização L2 e otimização ADAM

A Tabela 13 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 58 durante a fase de validação, enquanto que a Tabela 14 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 13 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização L2 e otimização ADAM, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98820	0,98645	0,96000	0,96578	0,94189	0,98638

Tabela 14 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização L2 eotimização ADAM, na etapa de validação

	Acurácia	Jaccard (IoU)	Score F1
Pulmão	0,98173	0,94444	0,92125
Background	0,99116	0,97557	0,96253

Na Figura 59 está a matriz de confusão obtida pela rede.

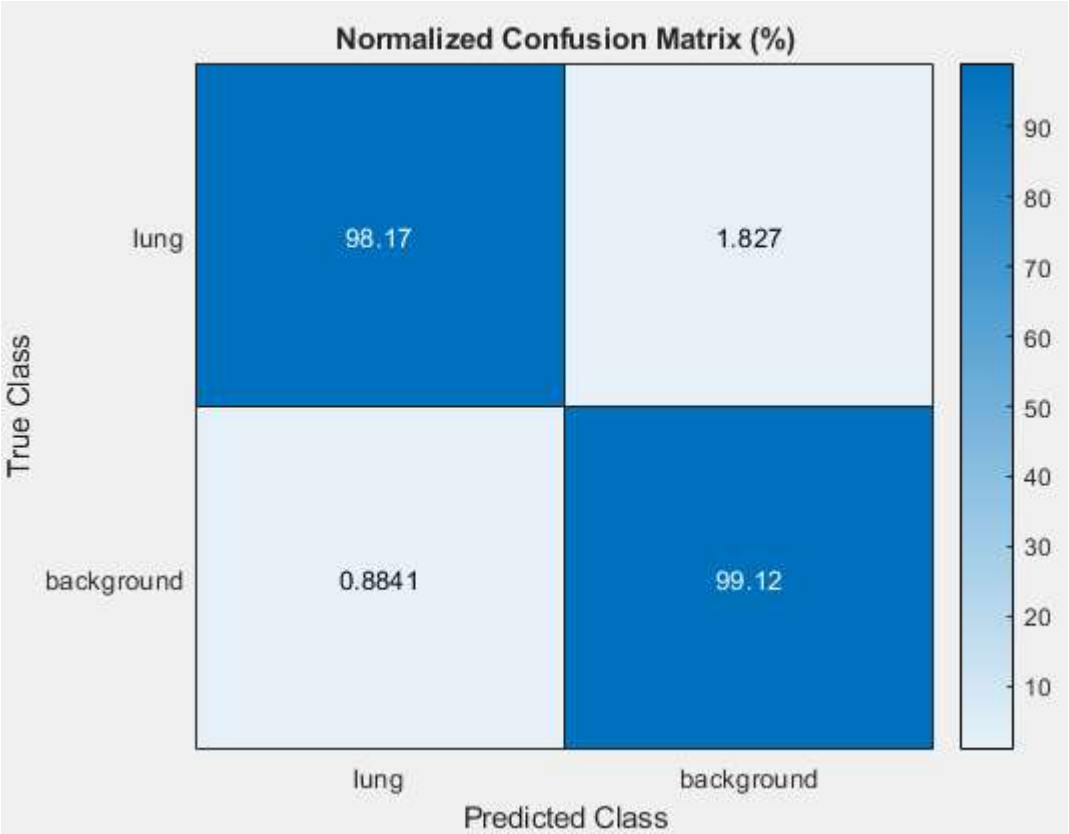


Figura 59 Matriz de confusão para arquitetura CNN1, utilizando regularização L2 e otimização ADAM

A Figura 60 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 58.

Figura 60 Segmentação realizada pela arquitetura CNN1, utilizando regularização L2 e otimização ADAM

A sétima simulação foi realizada na arquitetura CNN1, utilizando o método de regularização *Dropout*+L2 e otimização SGDM. A Figura 61 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

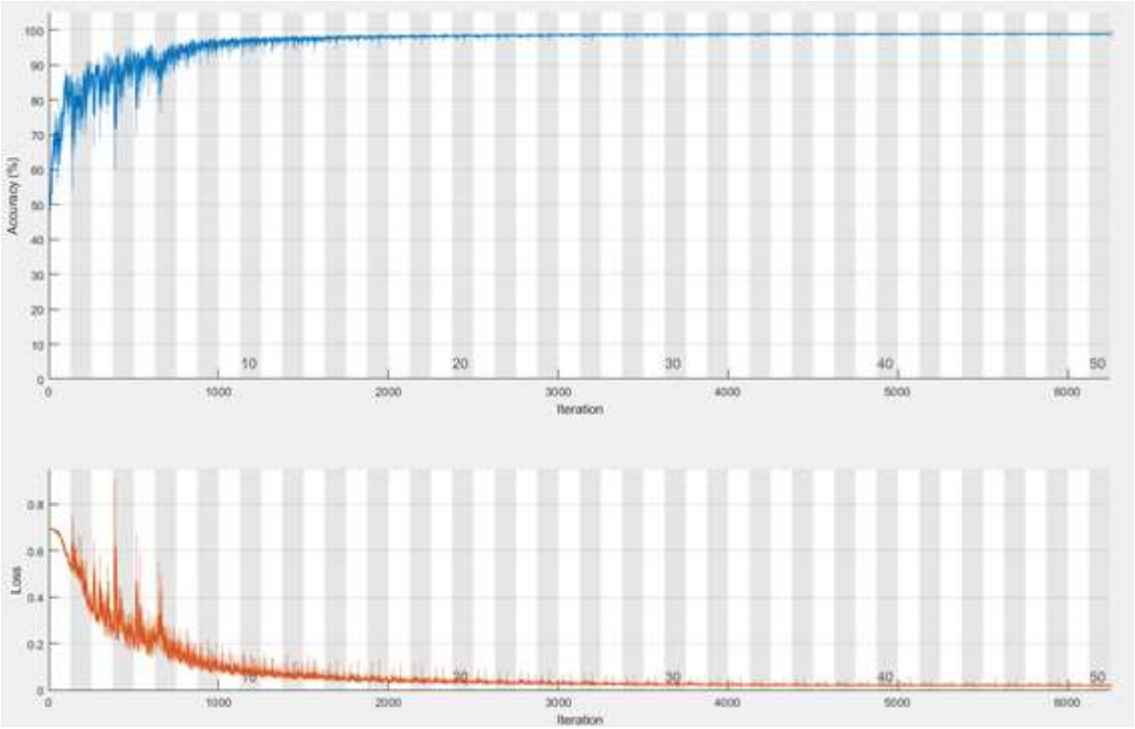


Figura 61 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização SGDM

A Tabela 15 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 61 durante a fase de validação, enquanto que a Tabela 16 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 15 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização SGDM, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,93560	0,95270	0,85582	0,87428	0,51559	0,95479

Tabela 16 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização SGDM, na etapa de validação

	Acurácia	Jaccard (IoU)	Score F1
Pulmão	0,99877	0,80608	0,35548
Background	0,90664	0,90556	0,67571

Na **Figura** 62 está a matriz de confusão obtida pela rede.

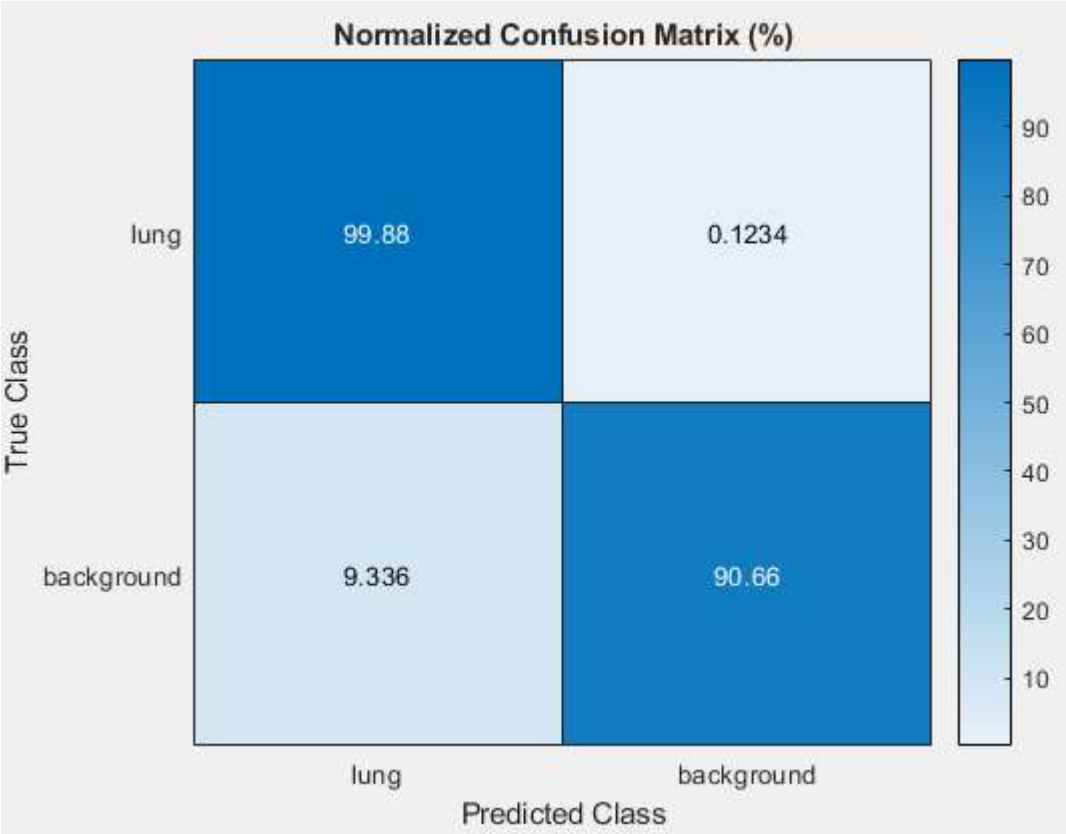


Figura 62 Matriz de confusão para arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização SGDM

A Figura 63 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 61.

Figura 63 Segmentação realizada pela arquitetura CNN1, utilizando regularização L2 e otimização SGDM

Os baixos valores para as métricas foram refletidos na segmentação obtida pela rede. Observando a matriz de confusão, nota-se um alto valor de FP (9,336%), indicando que vários pixels pertencentes a classe *Background* foram erroneamente classificados como pertencentes a classe Pulmão.

A oitava simulação foi realizada na arquitetura CNN1, utilizando o método de regularização *Dropout*+L2 e otimização RMSPROP. A Figura 64 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

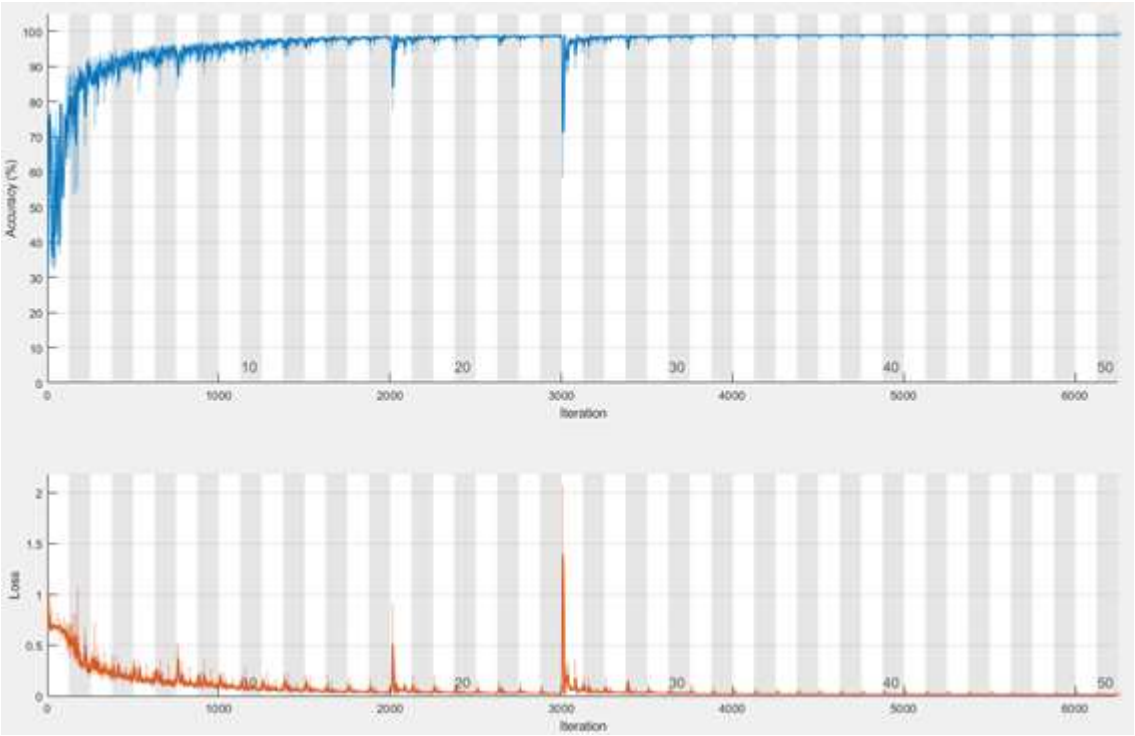


Figura 64 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A Tabela 17 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 64 durante a fase de validação, enquanto que a Tabela 18 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 17 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização RMSPROP, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98931	0,98860	0,97560	0,97891	0,97327	0,98858

Tabela 18 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização RMSPROP, na etapa de validação

	Acurácia	Jaccard (IoU)	Score F1
Pulmão	0,98669	0,96669	0,96501
Background	0,99052	0,98451	0,98154

Na **Figura** 65 está a matriz de confusão obtida pela rede.

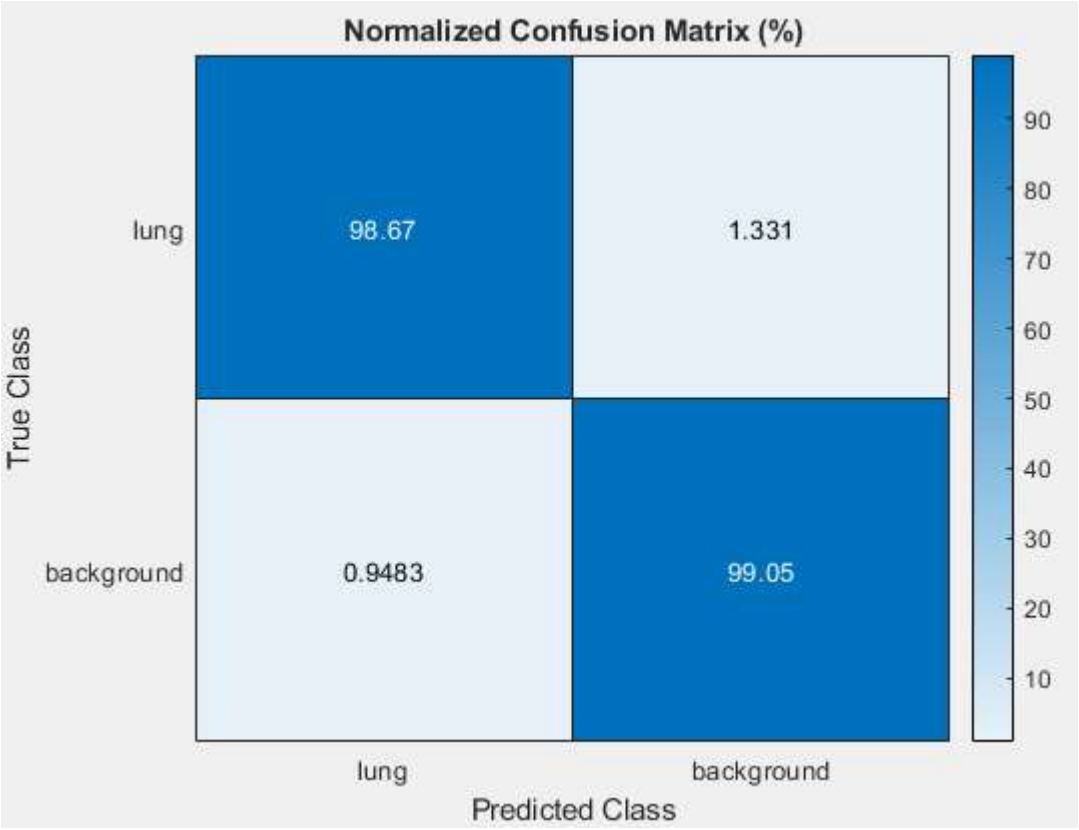


Figura 65 Matriz de confusão para arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A **Figura** 66 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 64.

Figura 66 Segmentação realizada pela arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização RMSPROP

As métricas obtidas nessa etapa foram boas, tendo se refletido na boa segmentação obtida pela rede, conforme observado na **Figura** 66.

A nona simulação foi realizada na arquitetura CNN1, utilizando o método de regularização *Dropout*+L2 e otimização ADAM. A Figura 67 mostra o comportamento gráfico da acurácia da rede CNN1 durante uma seção de treinamento, bem como o gráfico da função de perda.

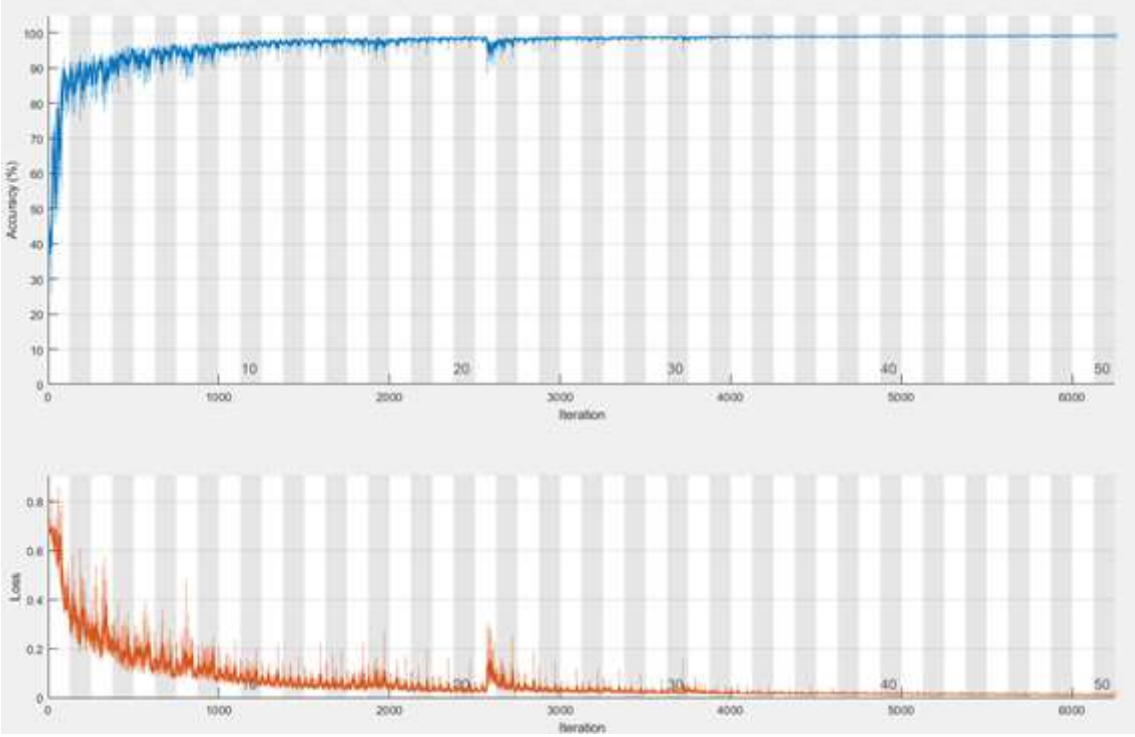


Figura 67 Gráficos da acurácia e função de perda para arquitetura CNN1, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização ADAM

A Tabela 19 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 67 durante a fase de validação, enquanto que a Tabela 20 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 19 Métricas de desempenho obtidas com a arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização ADAM, na etapa de validação

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98966	0,98818	0,96346	0,96857	0,95279	0,98813

Tabela 20 Métricas de desempenho obtidas por classe com a arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização ADAM, na etapa de validação

	Acurácia	Jaccard (IoU)	Score F1
Pulmão	0,98418	0,94971	0,93556
Background	0,99218	0,97722	0,97002

Na Figura 68 está a matriz de confusão obtida pela rede.

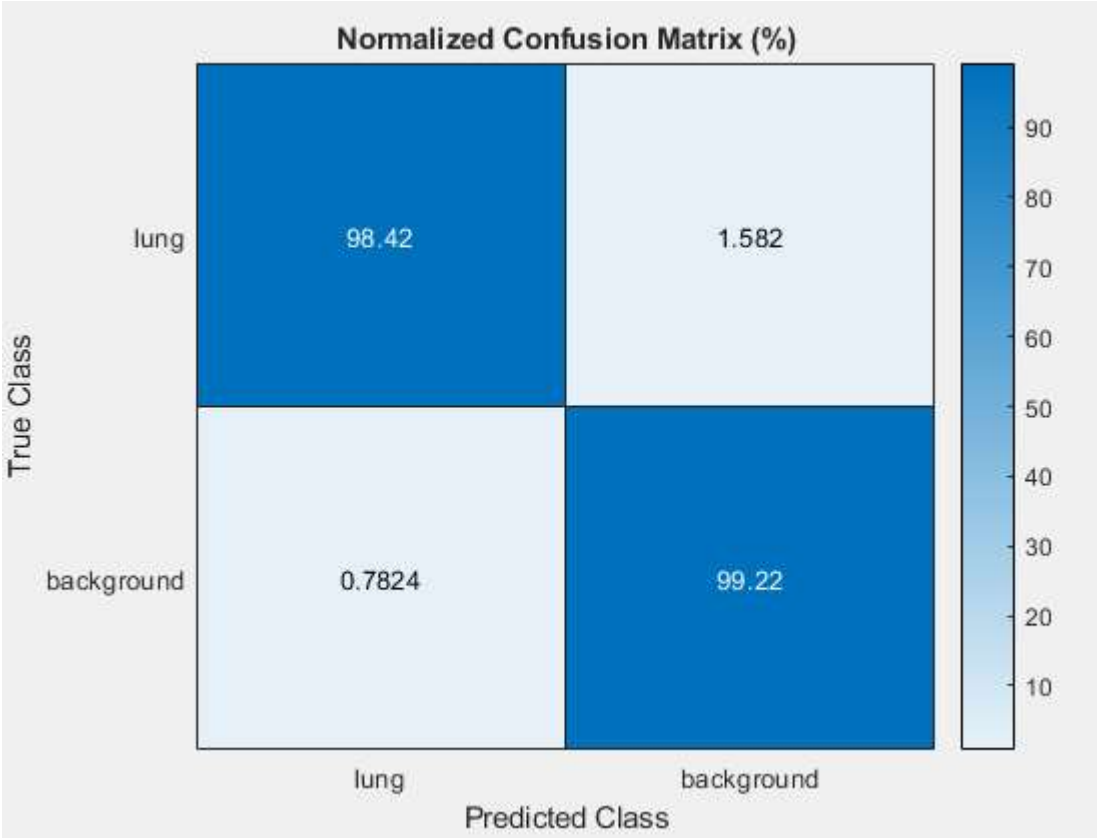


Figura 68 Matriz de confusão para arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização ADAM

A Figura 69 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 67.

Figura 69 Segmentação realizada pela arquitetura CNN1, utilizando regularização *Dropout*+L2 e otimização ADAM

Tendo concluído as nove simulações referentes a rede convolutiva CNN1, a melhor combinação obtida foi utilizando o método de regularização *Dropout*+L2 e o método de otimização RMSPROP.

5.2 SIMULAÇÕES REALIZADAS COM A REDE CONVOLUTIVA CNN2

A décima simulação foi realizada na arquitetura CNN2, utilizando o método de regularização *Dropout* e otimização SGDM. A Figura 70 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

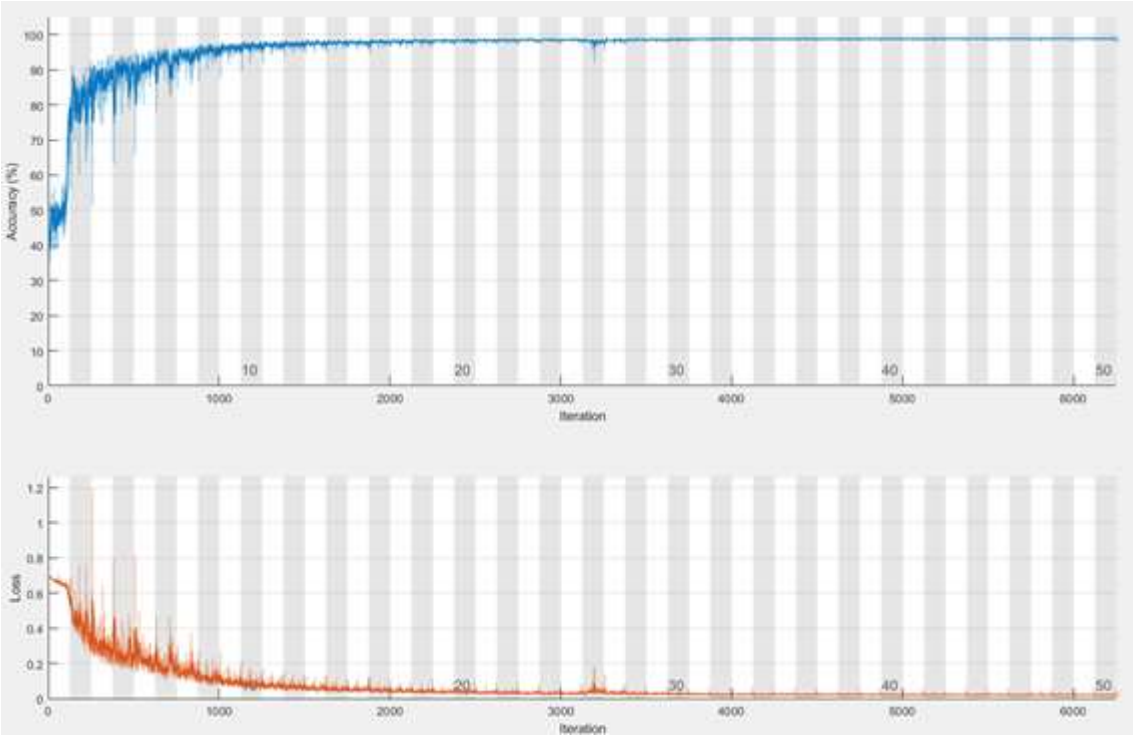


Figura 70 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização SGDM

A Tabela 21 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 70, durante a fase de validação, enquanto que a Tabela 22 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 21 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98048	0,98400	0,94230	0,95172	0,89364	0,98415

Tabela 22 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,99349	0,91691	0,84201
Background	0,97451	0,96769	0,94527

Na Figura 71 está a matriz de confusão obtida pela rede.

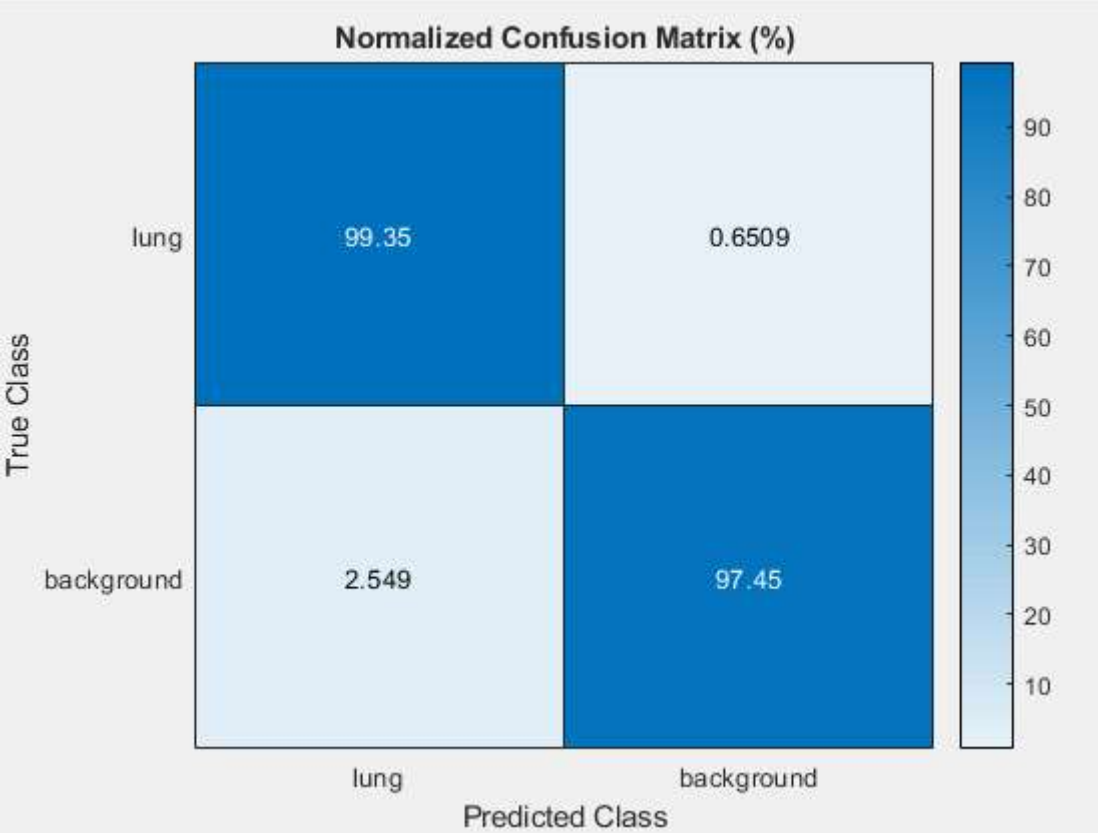


Figura 71 Matriz de confusão para arquitetura CNN2, utilizando regularização *Dropout* e otimização SGDM

A Figura 72 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 70.

Figura 72 Segmentação realizada pela arquitetura CNN2, utilizando regularização *Dropout* e otimização SGDM

Comparando os resultados obtidos nessa simulação com aqueles obtidos na primeira simulação (onde se usa os mesmos métodos de regularização e otimização, porém com a arquitetura CNN1) tem-se uma melhora substancial nas métricas obtidas, entretanto o Score F1 continua relativamente com um valor baixo.

A décima primeira simulação foi realizada na arquitetura CNN2, utilizando o método de regularização *Dropout* e otimização RMSPROP. A Figura 73 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

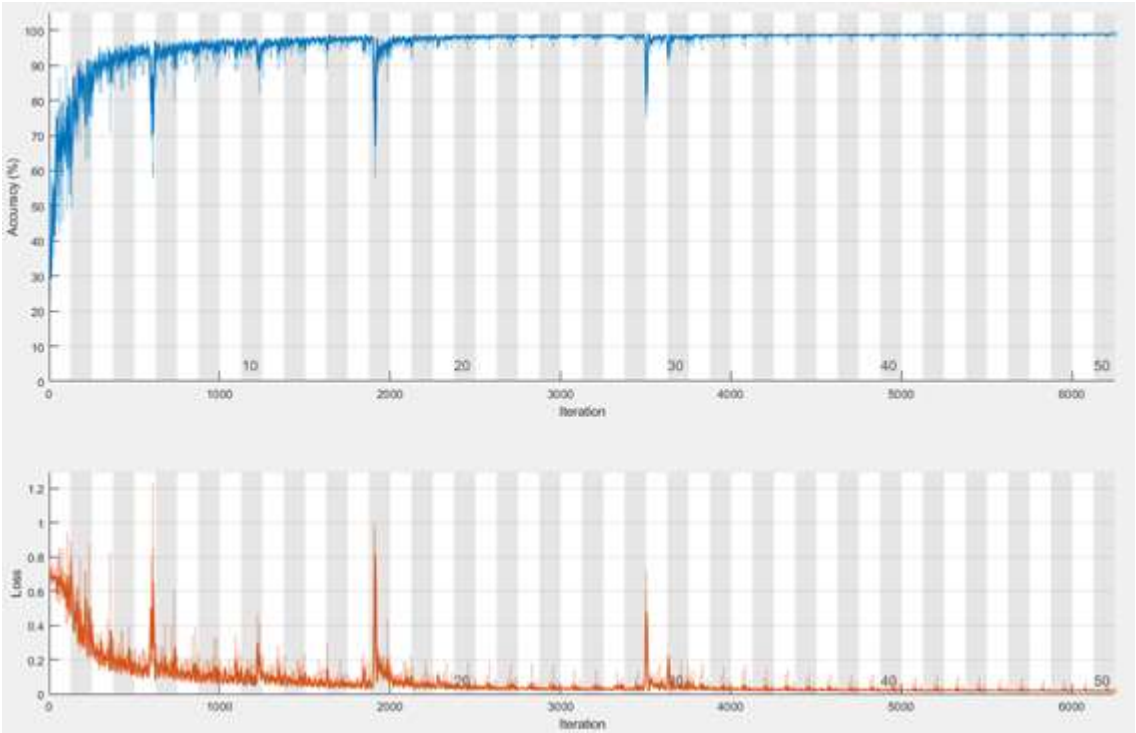


Figura 73 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização RMSPROP

A Tabela 23 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 73, durante a fase de validação, enquanto que a Tabela 24 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 23 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização *Dropout* e otimização RMSPROP, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98986	0,98970	0,96327	0,96905	0,95366	0,98970

Tabela 24 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização *Dropout* e otimização RMSPROP, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98927	0,94722	0,93302
Background	0,99014	0,97882	0,97429

Na Figura 74 está a matriz de confusão obtida pela rede.

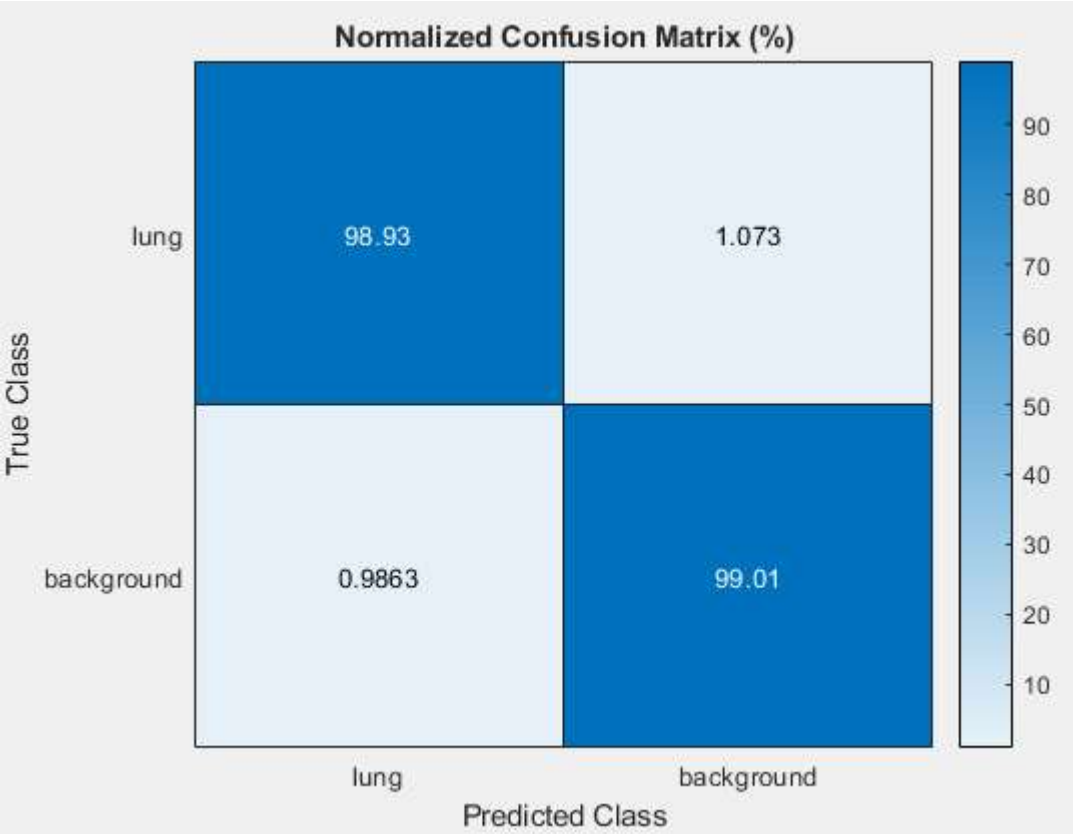


Figura 74 Matriz de confusão para arquitetura CNN2, utilizando regularização *Dropout* e otimização RMSPROP

A Figura 75 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 73.

Figura 75 Segmentação realizada pela arquitetura CNN2, utilizando regularização *Dropout* e otimização RMSPROP

As simulações realizadas nessa etapa obtiveram métricas elevadas, tendo-se conseguido uma melhoria no Score F1. Os bons valores das métricas foram refletidos na segmentação obtida pela CNN.

A décima segunda simulação foi realizada na arquitetura CNN2, utilizando o método de regularização *Dropout* e otimização ADAM. A Figura 76 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

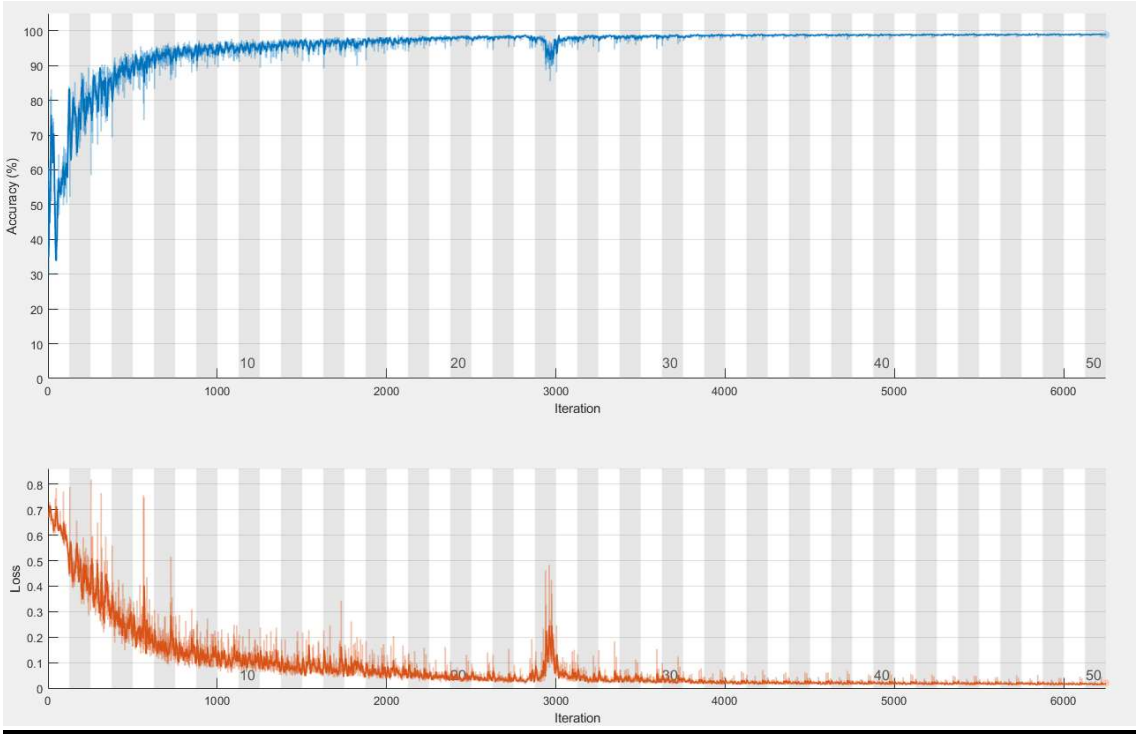


Figura 76 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização ADAM

A Tabela 25 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 76, durante a fase de validação, enquanto que a Tabela 26 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 25 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização *Dropout* e otimização ADAM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98929	0,98913	0,96199	0,96795	0,95405	0,98912

Tabela 26 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização *Dropout* e otimização ADAM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98871	0,94594	0,93447
Background	0,98955	0,97804	0,97362

Na Figura 77 está a matriz de confusão obtida pela rede.

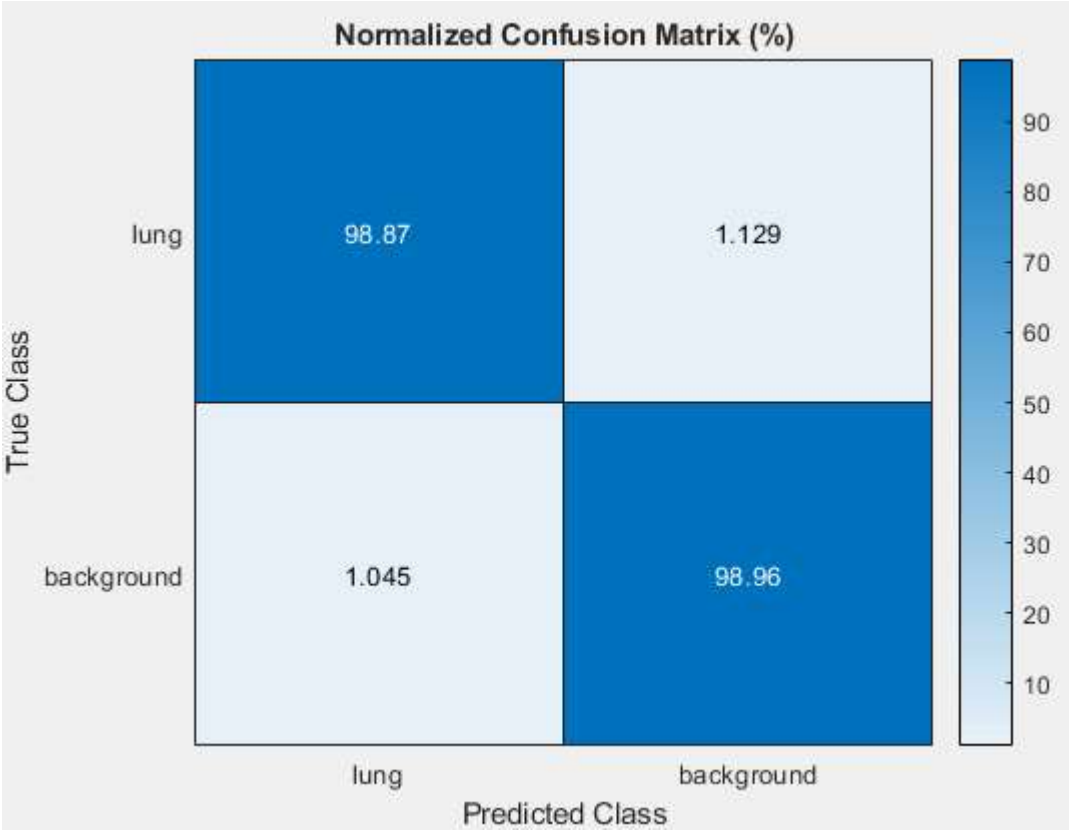


Figura 77 Matriz de confusão para arquitetura CNN2, utilizando regularização *Dropout* e otimização ADAM

A Figura 78 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 76.

Figura 78 Segmentação realizada pela arquitetura CNN2, utilizando regularização *Dropout* e otimização ADAM

A décima terceira simulação foi realizada na arquitetura CNN2, utilizando o método de regularização L2 e otimização SGDM. A Figura 79 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

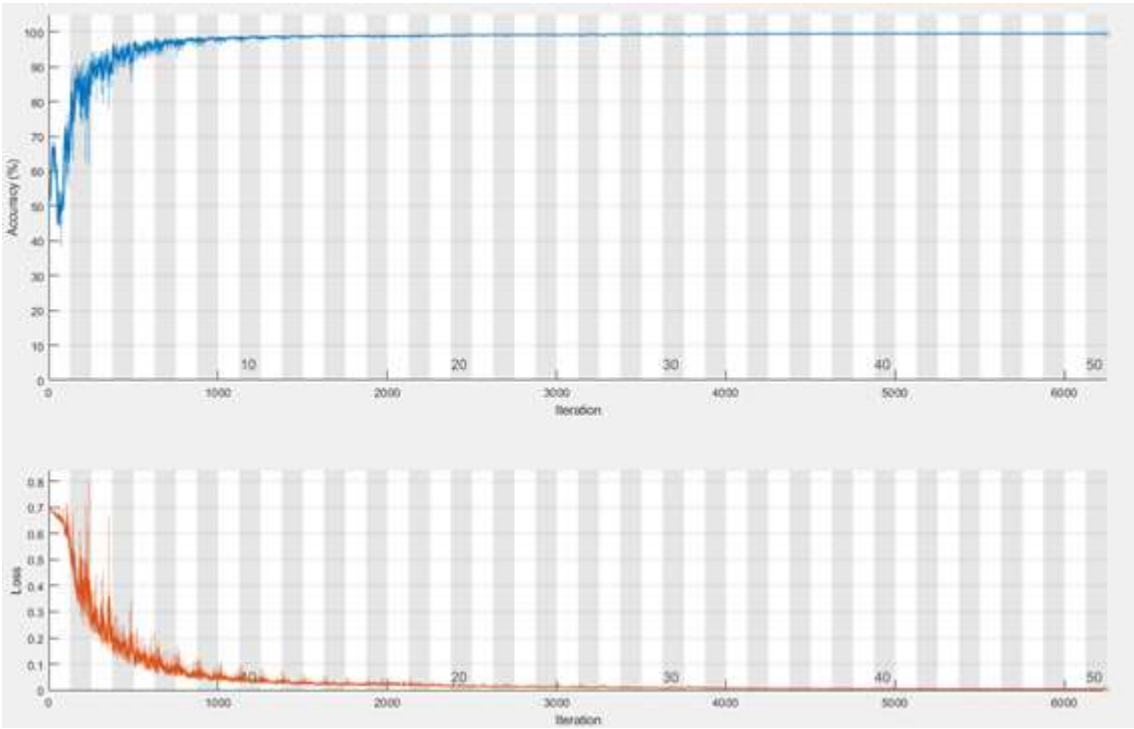


Figura 79 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização L2 e otimização SGDM

A Tabela 27 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 79, durante a fase de validação, enquanto que a Tabela 28 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 27 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização L2 e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98496	0,98171	0,95321	0,95957	0,92118	0,98155

Tabela 28 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização L2 e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,97295	0,93607	0,89499
Background	0,99047	0,97035	0,94736

Na Figura 80 está a matriz de confusão obtida pela rede.

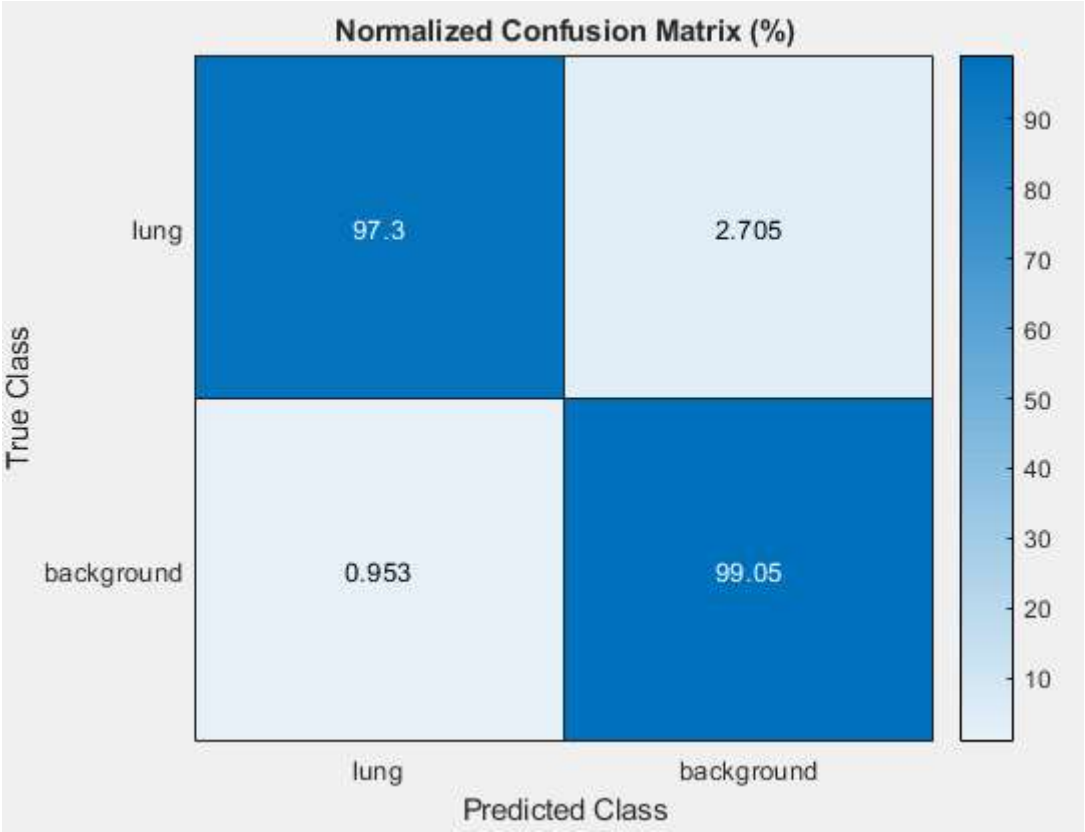


Figura 80 Matriz de confusão para arquitetura CNN2, utilizando regularização L2 e otimização SGDM

A Figura 81 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 79.

Figura 81 Segmentação realizada pela arquitetura CNN2, utilizando regularização L2 e otimização SGDM

A décima quarta simulação foi realizada na arquitetura CNN2, utilizando o método de regularização L2 e otimização RMSPROP. A Figura 82 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

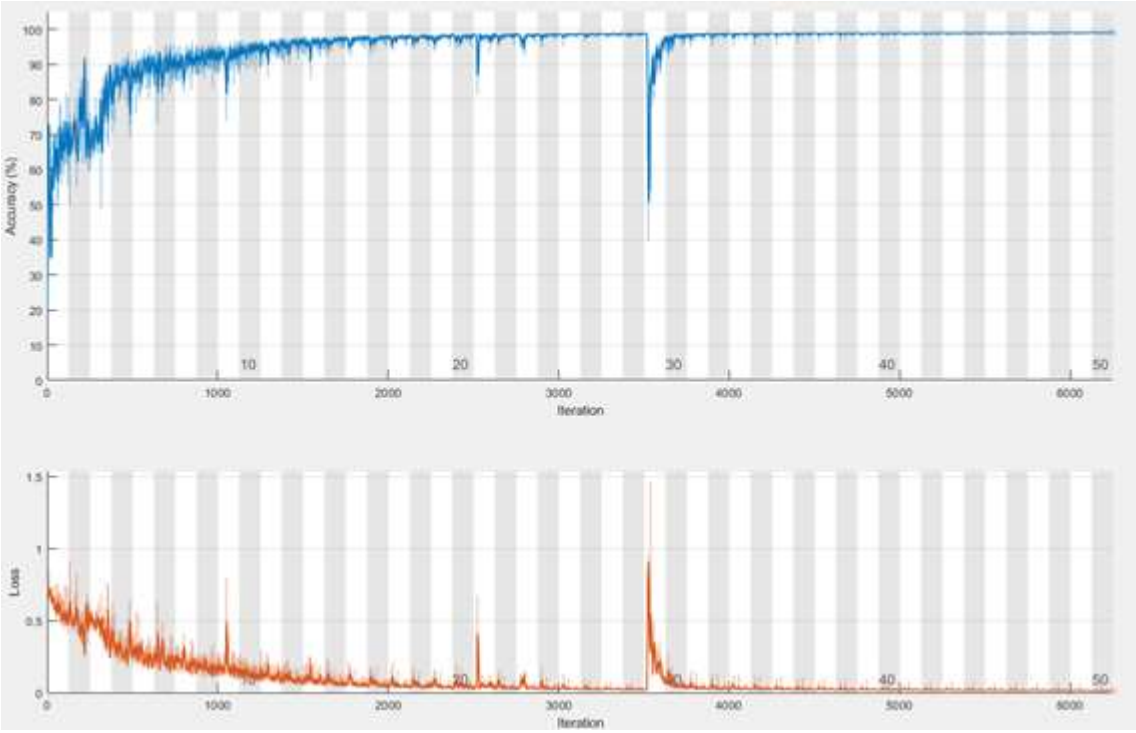


Figura 82 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização L2 e otimização RMSPROP

A Tabela 29 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 82, durante a fase de validação, enquanto que a Tabela 30 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 29 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização L2 e otimização RMSPROP, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98878	0,98770	0,96129	0,96692	0,94759	0,98766

Tabela 30 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização L2 e otimização RMSPROP, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98478	0,94613	0,92722
Background	0,99062	0,97645	0,96796

Na Figura 83 está a matriz de confusão obtida pela rede.

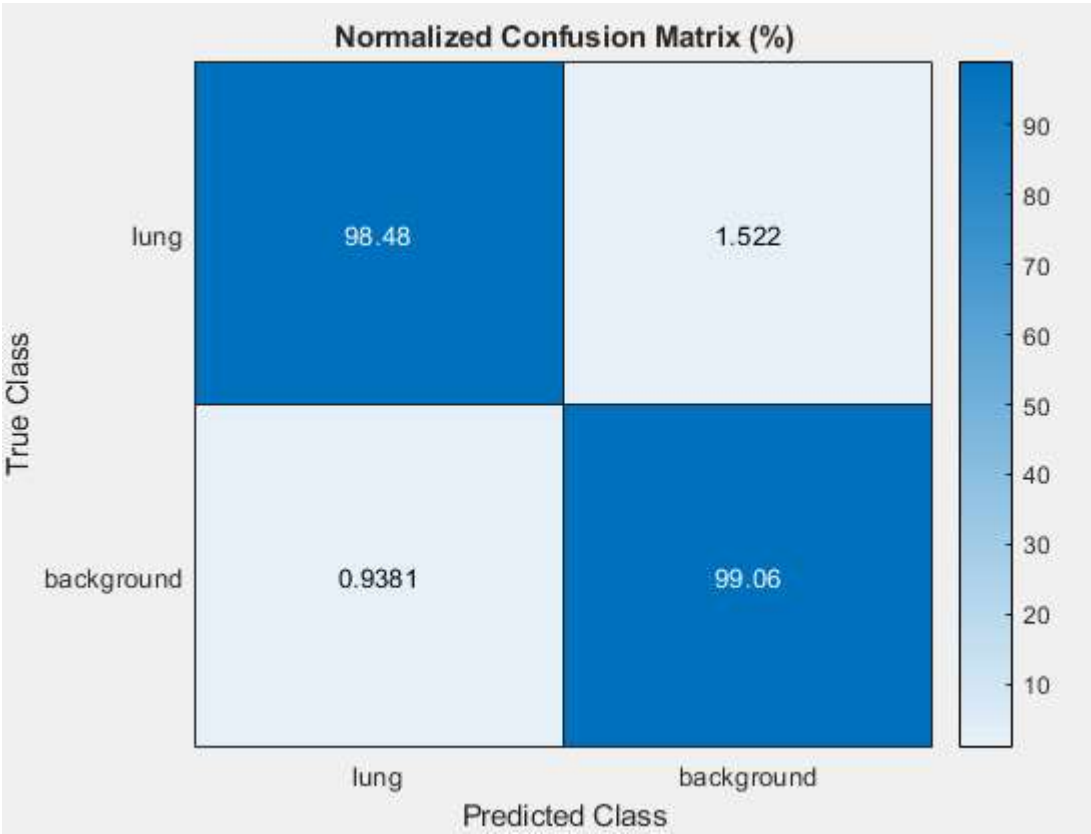


Figura 83 Matriz de confusão para arquitetura CNN2, utilizando regularização L2 e otimização RMSPROP

A Figura 84 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 82.

Figura 84 Segmentação realizada pela arquitetura CNN2, utilizando regularização L2 e otimização RMSPROP

A décima quinta simulação foi realizada na arquitetura CNN2, utilizando o método de regularização L2 e otimização ADAM. A Figura 85 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

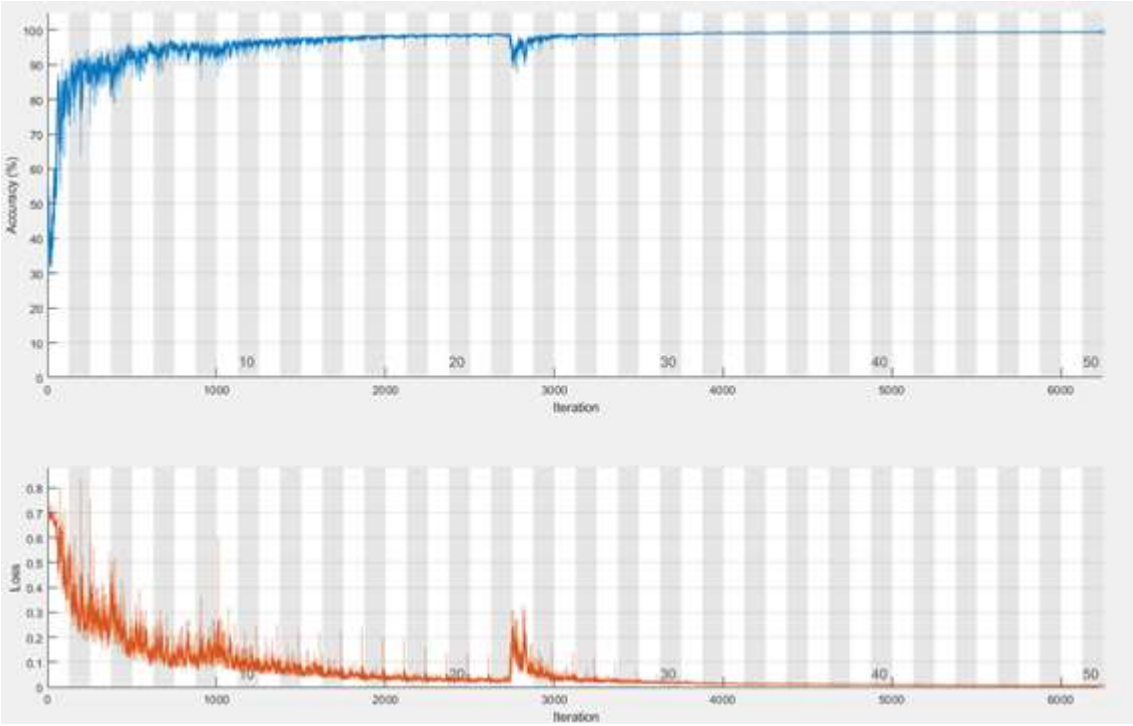


Figura 85 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização L2 e otimização ADAM

A Tabela 31 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 85, durante a fase de validação, enquanto que a Tabela 32 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 31 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização L2 e otimização ADAM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98610	0,98483	0,95587	0,96180	0,93041	0,98477

Tabela 32 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização L2 e otimização ADAM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98141	0,93914	0,90378
Background	0,98824	0,97219	0,95705

Na Figura 86 está a matriz de confusão obtida pela rede.

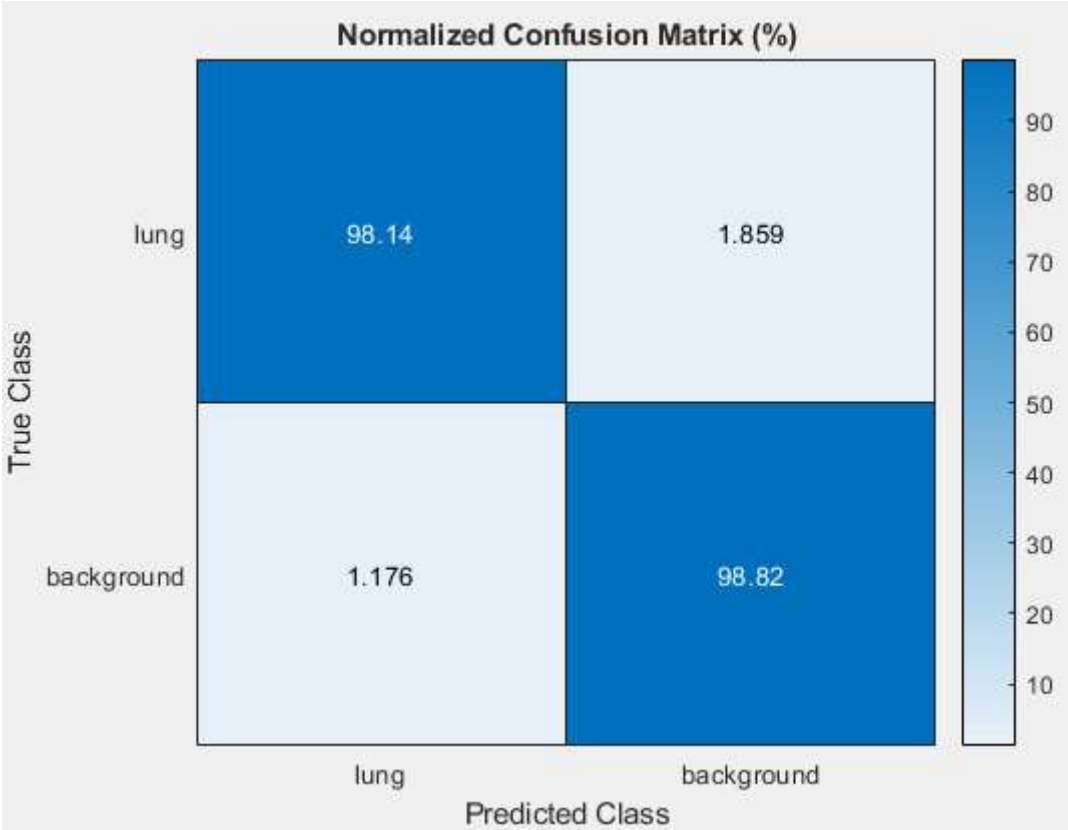


Figura 86 Matriz de confusão para arquitetura CNN2, utilizando regularização L2 e otimização ADAM

A Figura 87 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 85.

Figura 87 Segmentação realizada pela arquitetura CNN2, utilizando regularização L2 e otimização ADAM

A décima sexta simulação foi realizada na arquitetura CNN2, utilizando o método de regularização *Dropout*+L2 e otimização SGDM. A Figura 88 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

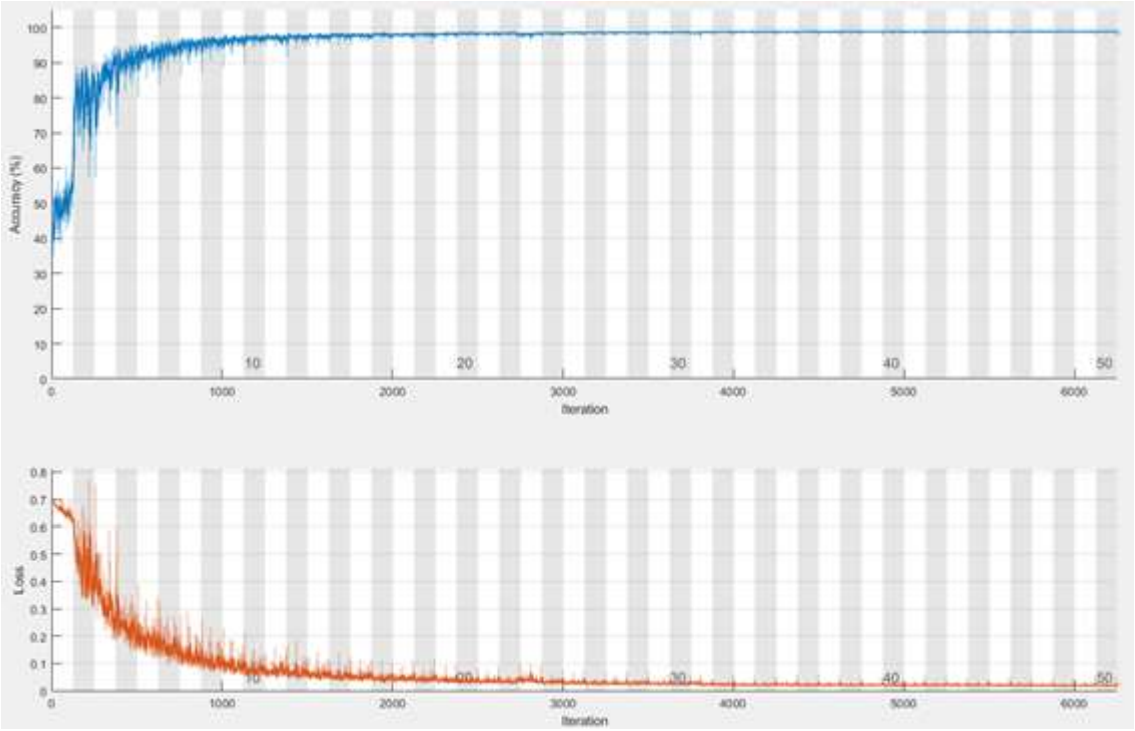


Figura 88 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização SGDM

A Tabela 33 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 88, durante a fase de validação, enquanto que a Tabela 34 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 33 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,97637	0,98129	0,93354	0,94430	0,85673	0,98154

Tabela 34 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,99454	0,90456	0,78425
Background	0,96804	0,96253	0,92921

Na Figura 89 está a matriz de confusão obtida pela rede.

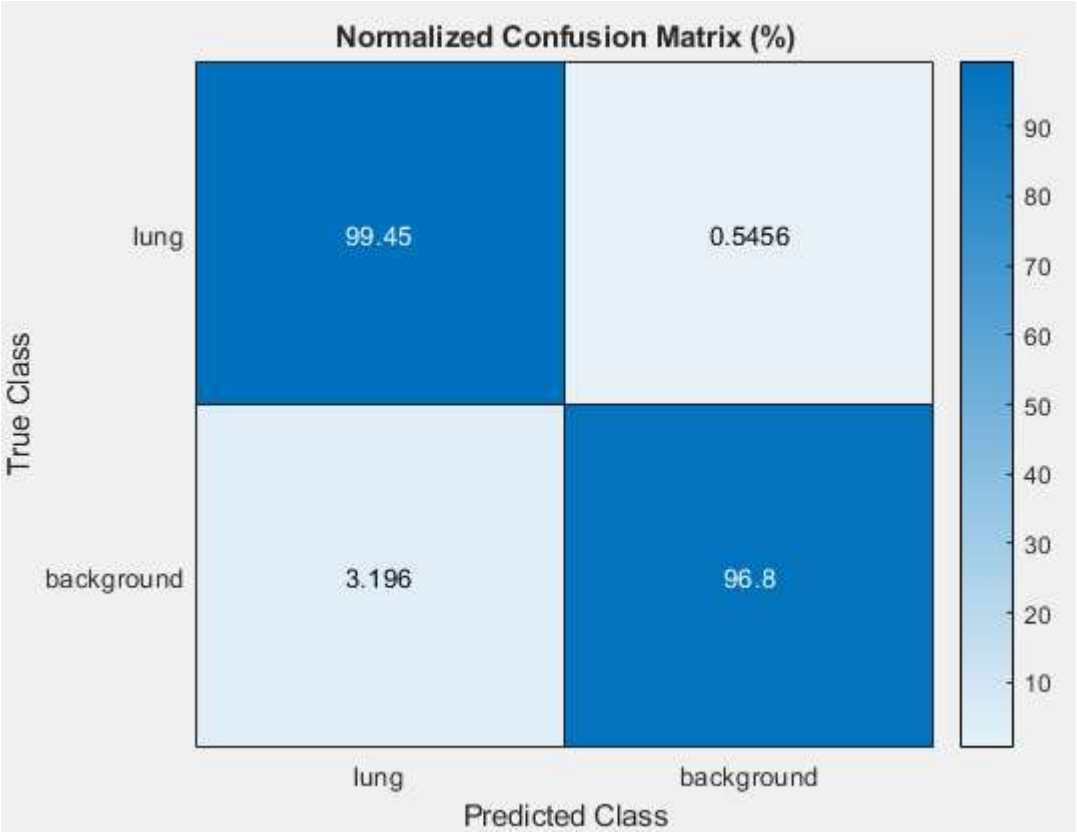


Figura 89 Matriz de confusão para arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização SGDM

A Figura 90 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 88.

Figura 90 Segmentação realizada pela arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização SGDM

A décima sétima simulação foi realizada na arquitetura CNN2, utilizando o método de regularização *Dropout*+L2 e otimização RMSPROP. A Figura 91 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

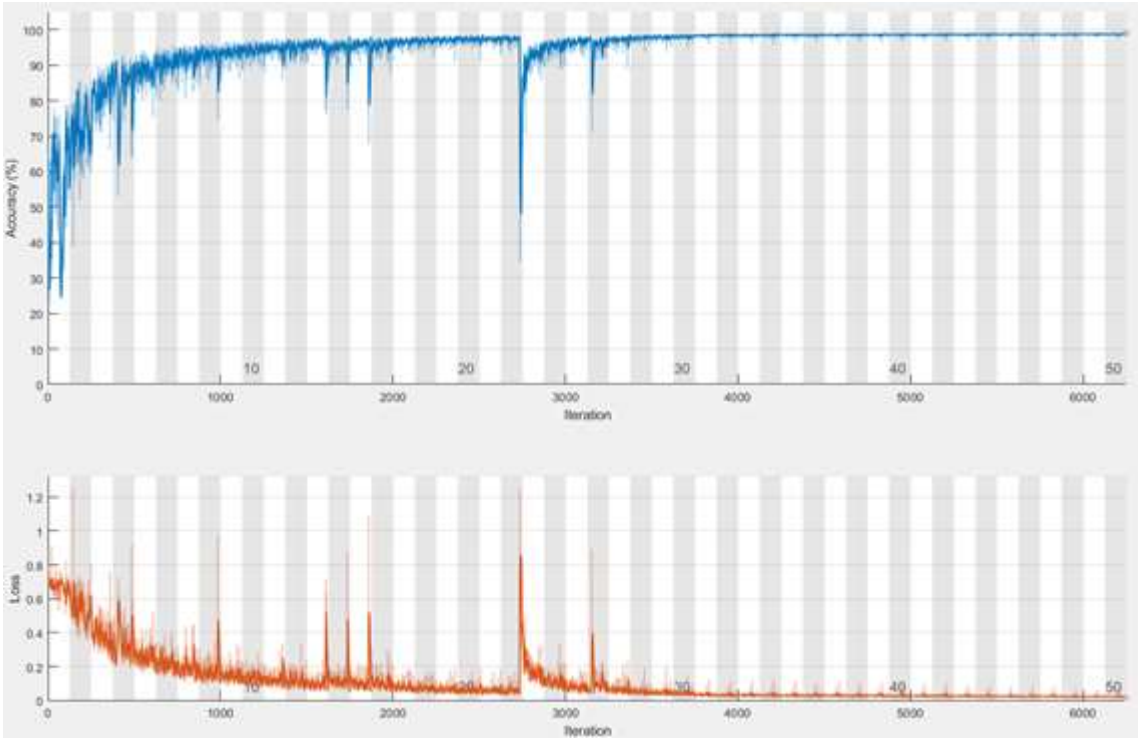


Figura 91 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A Tabela 35 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 91, durante a fase de validação, enquanto que a Tabela 36 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 35 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização RMSPROP, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98876	0,98826	0,96096	0,96693	0,94699	0,98824

Tabela 36 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização RMSPROP, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98692	0,94488	0,92402
Background	0,98960	0,97704	0,96996

Na Figura 92 está a matriz de confusão obtida pela rede.

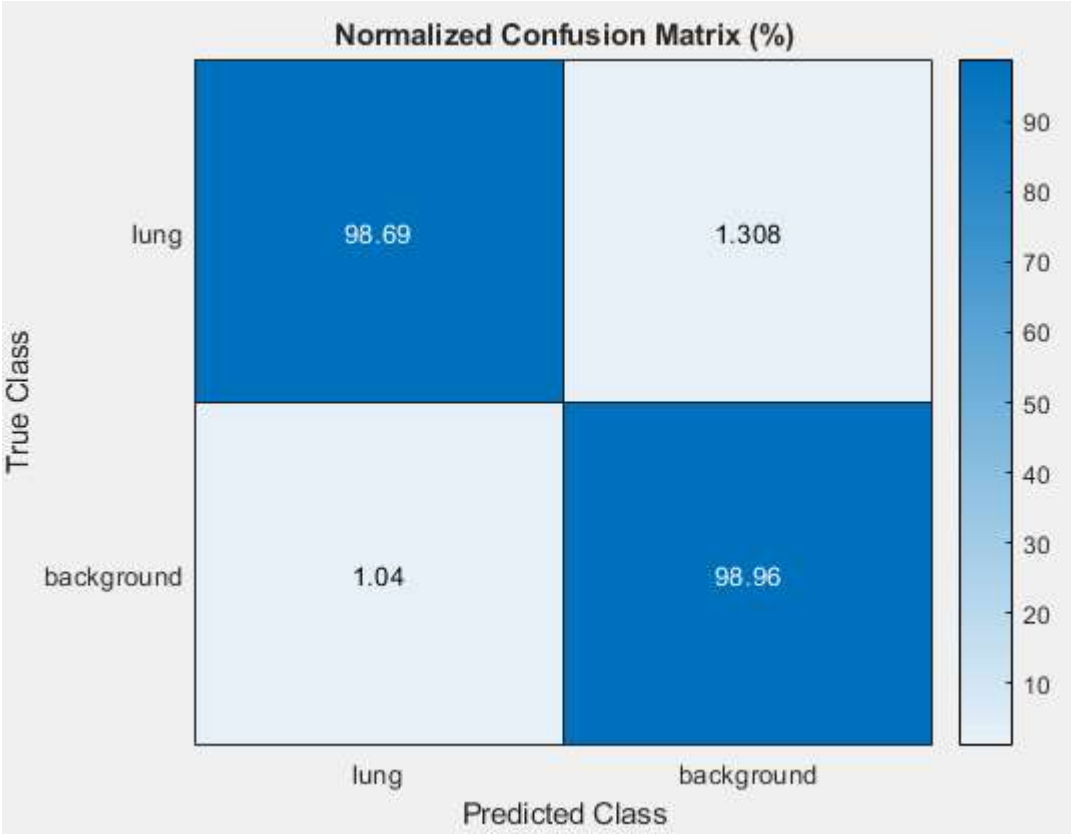


Figura 92 Matriz de confusão para arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A Figura 93 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 91.

Figura 93 Segmentação realizada pela arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A décima oitava simulação foi realizada na arquitetura CNN2, utilizando o método de regularização *Dropout*+L2 e otimização ADAM. A Figura 94 mostra o comportamento da acurácia da rede CNN2 durante uma seção de treinamento, bem como o gráfico da função de perda.

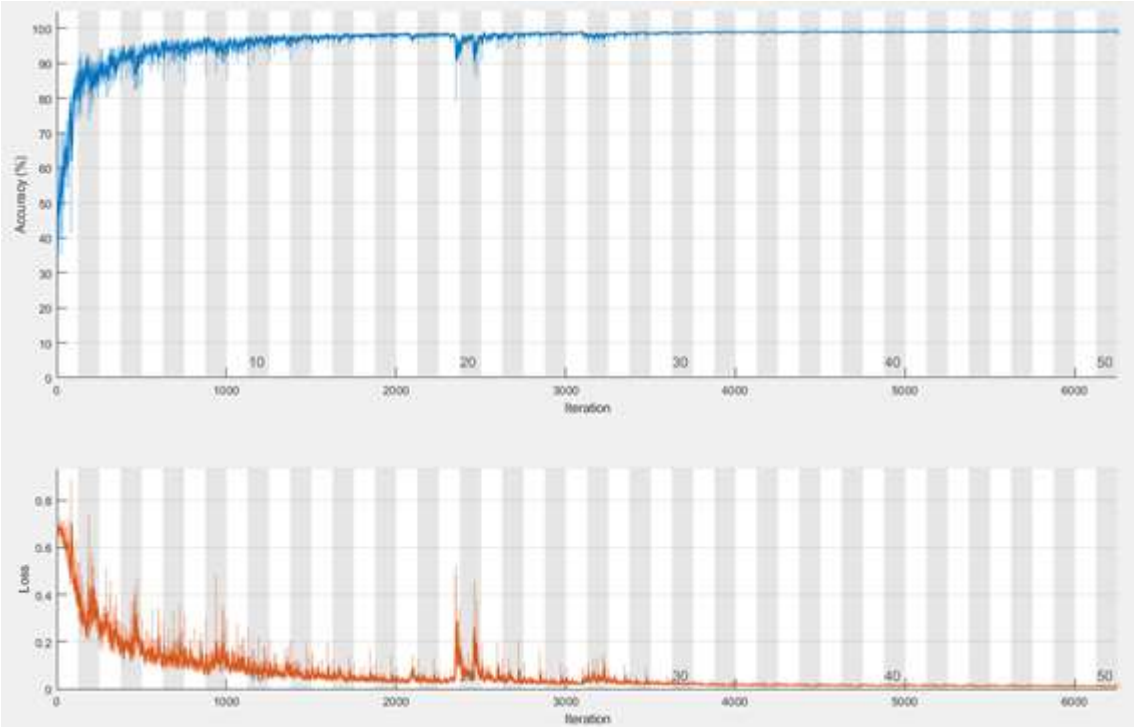


Figura 94 Gráficos da acurácia e função de perda para arquitetura CNN2, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização ADAM

A Tabela 37 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 94, durante a fase de validação, enquanto que a Tabela 38 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 37 Métricas de desempenho obtidas com a arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização ADAM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98917	0,98871	0,96184	0,96772	0,94960	0,98870

Tabela 38 Métricas de desempenho obtidas por classe com a arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização ADAM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98746	0,94602	0,92814
Background	0,98996	0,97767	0,97107

Na Figura 95 está a matriz de confusão obtida pela rede.

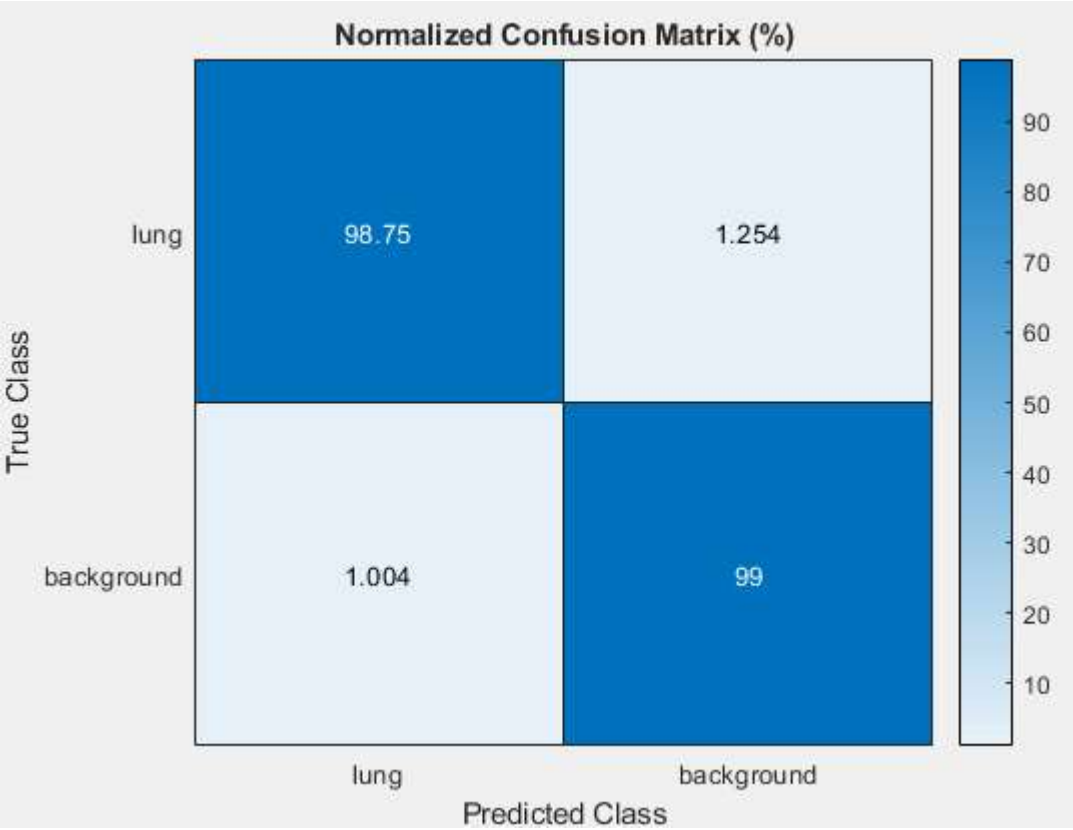


Figura 95 Matriz de confusão para arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização ADAM

A Figura 96 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 94.

Figura 96 Segmentação realizada pela arquitetura CNN2, utilizando regularização *Dropout*+L2 e otimização ADAM

Terminadas as simulações referentes a arquitetura da rede convolutiva CNN2, nota-se superioridade em relação a maioria dos resultados obtidos com a arquitetura CNN1. Motivado pela inserção de camadas adicionais de Convolução, *Batch* e *ReLu* entre as camadas de Convolução Transposta, as bordas das imagens segmentadas pela CNN2 ganharam mais suavidade. Destacaram-se como melhores resultados os obtidos utilizando regularização *Dropout* e métodos de otimização RMSPROP e ADAM.

5.3 SIMULAÇÕES REALIZADAS COM A REDE CONVOLUTIVA CNN3

A décima nona simulação foi realizada na arquitetura CNN3, utilizando o método de regularização *Dropout* e otimização SGDM. A Figura 97 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

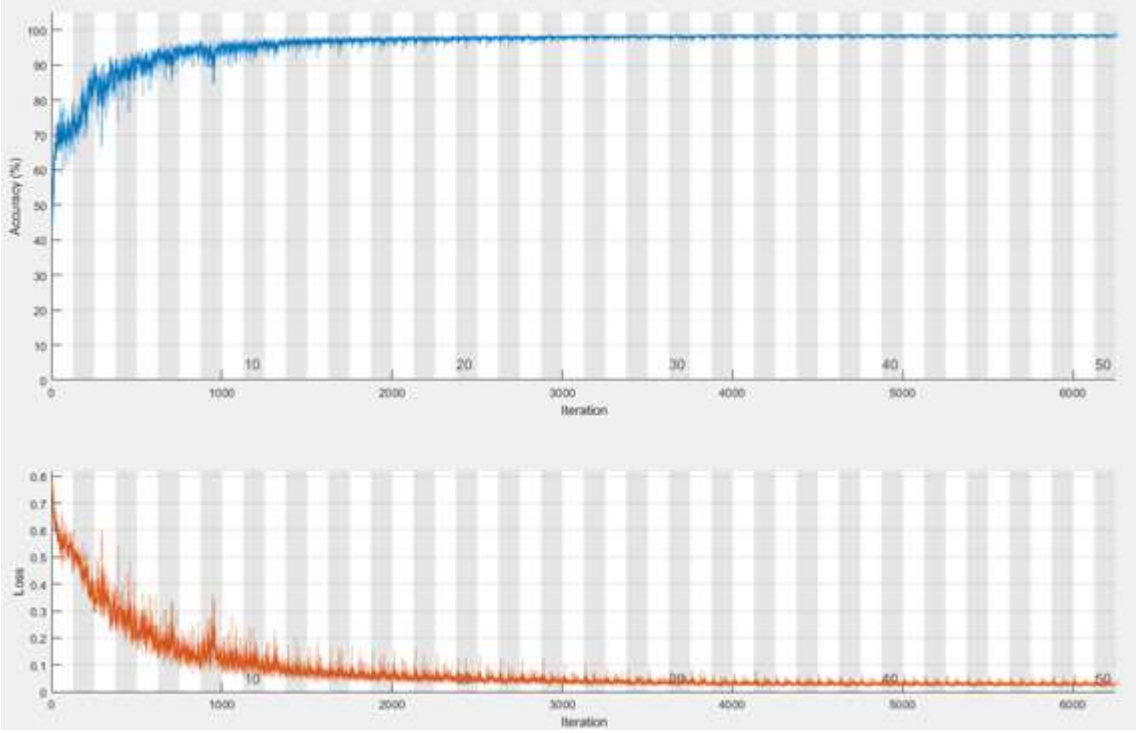


Figura 97 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização SGDM

A Tabela 39 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 97, durante a fase de validação, enquanto que a Tabela 40 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 39 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,97900	0,97647	0,95267	0,95903	0,91307	0,97631

Tabela 40 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização *Dropout* e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,96966	0,93556	0,88783
Background	0,98328	0,96979	0,93830

Na Figura 98 está a matriz de confusão obtida pela rede.

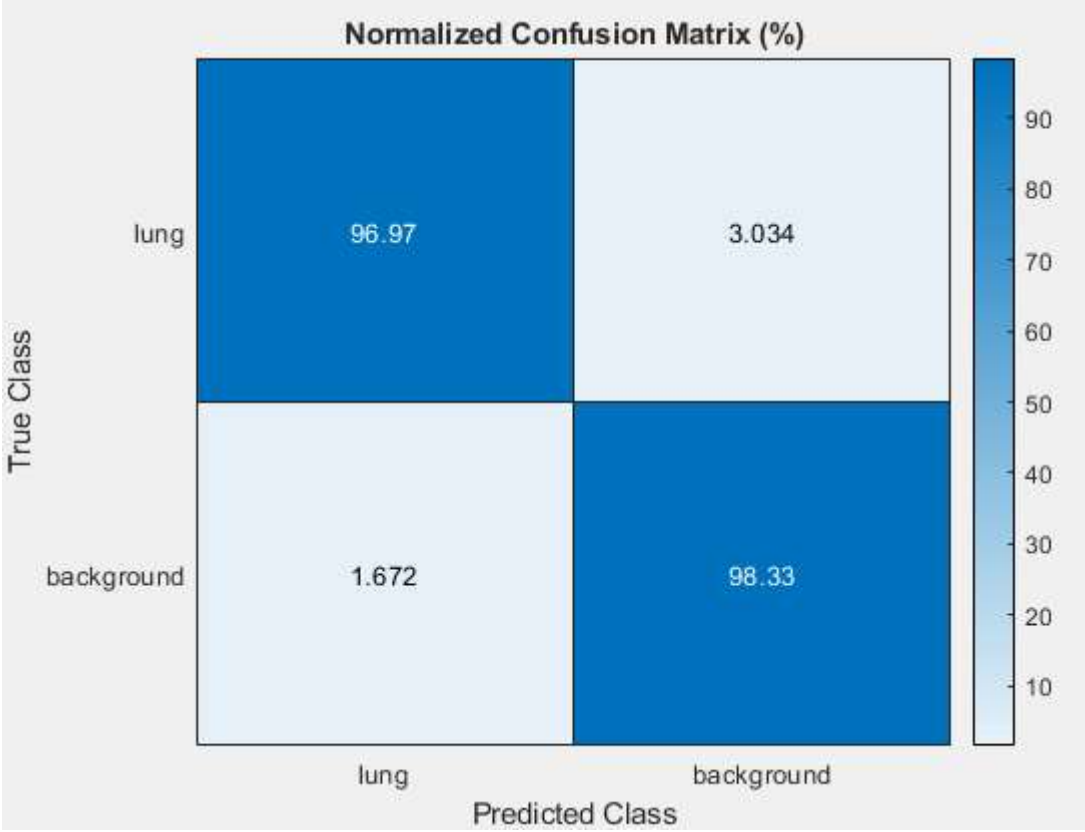


Figura 98 Matriz de confusão para arquitetura CNN3, utilizando regularização *Dropout* e otimização SGDM

A Figura 99 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 97.

Figura 99 Segmentação realizada pela arquitetura CNN3, utilizando regularização *Dropout* e otimização SGDM

Os valores obtidos para ScoreF1 e Jaccard foram baixos e isso se refletiu na segmentação obtida pela CNN. O método de otimização SGDM, em geral, não se mostrou satisfatório nessa tarefa de segmentação.

A vigésima simulação foi realizada na arquitetura CNN3, utilizando o método de regularização *Dropout* e otimização RMSPROP. A Figura 100 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

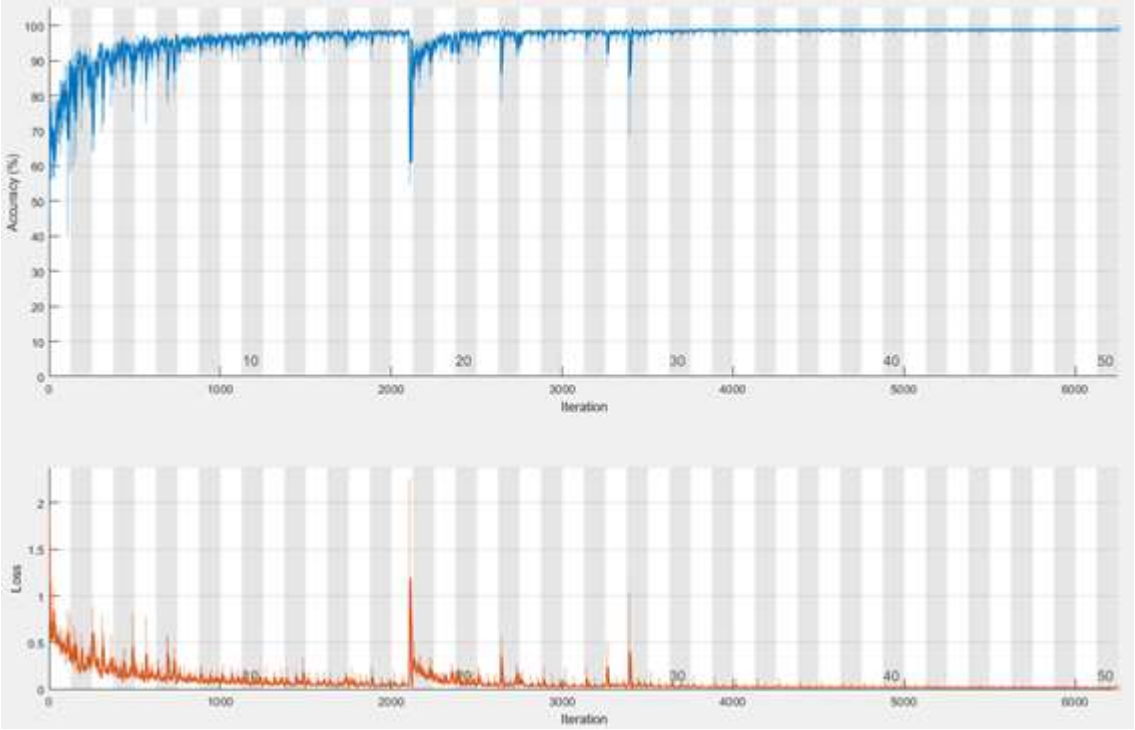


Figura 100 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização RMSPROP

A Tabela 41 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 100, durante a fase de validação, enquanto que a Tabela 42 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 41 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização *Dropout* e otimização RMSPROP, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,99041	0,98849	0,97802	0,98103	0,96777	0,98843

Tabela 42 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização *Dropout* e otimização RMSPROP, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98332	0,96992	0,95591
Background	0,99366	0,98612	0,97964

Na Figura 101 está a matriz de confusão obtida pela rede.

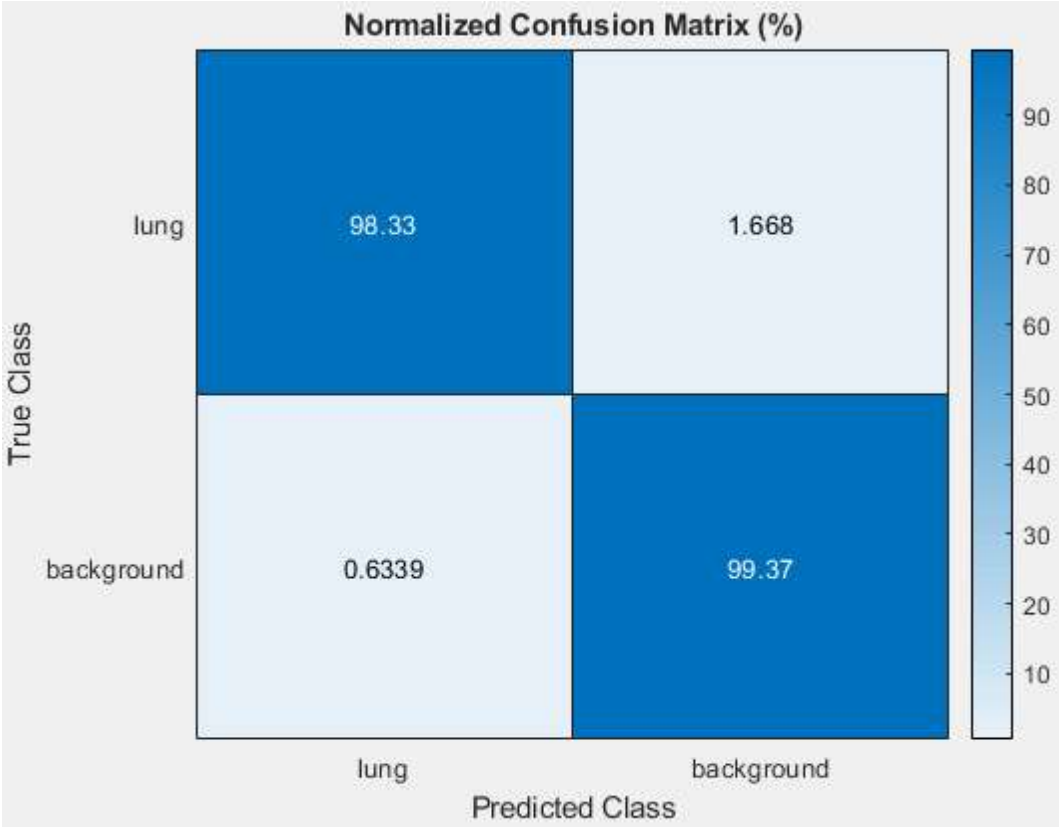


Figura 101 Matriz de confusão para arquitetura CNN3, utilizando regularização *Dropout* e otimização RMSPROP

A Figura 102 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 100.

Figura 102 Segmentação realizada pela arquitetura CNN3, utilizando regularização *Dropout* e otimização RMSPROP

A vigésima primeira simulação foi realizada na arquitetura CNN3, utilizando o método de regularização *Dropout* e otimização ADAM. A Figura 103 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

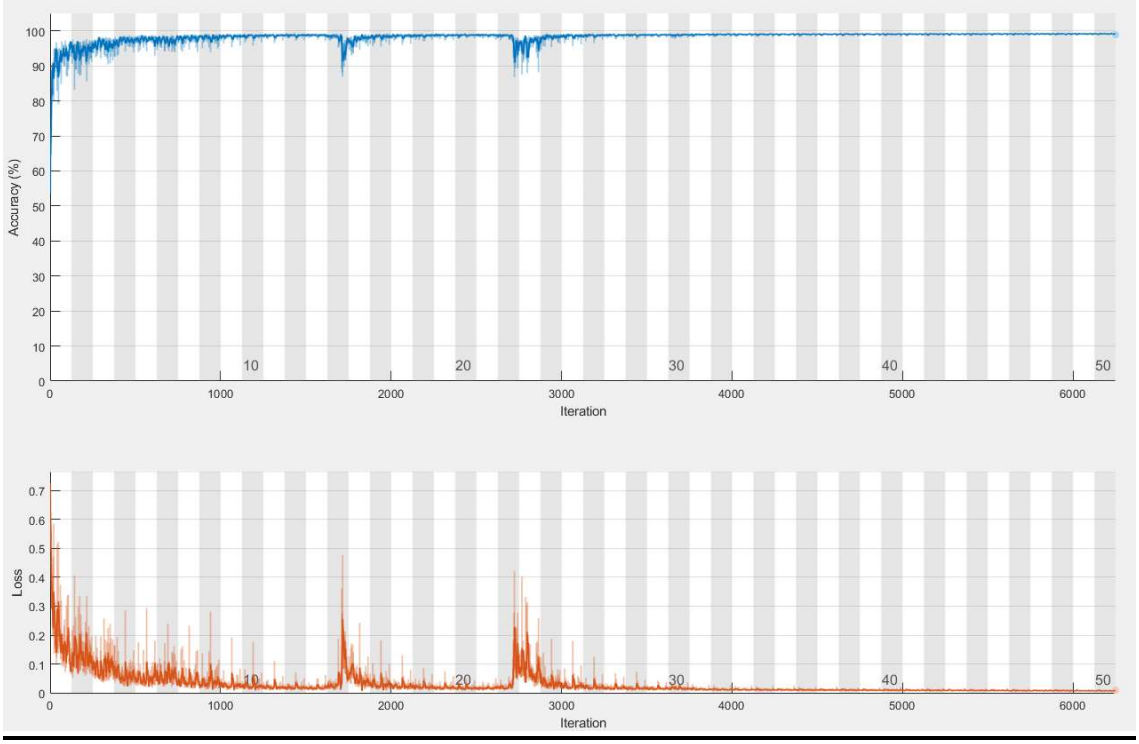


Figura 103 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização *Dropout* e otimização ADAM

A Tabela 43 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 103, durante a fase de validação, enquanto que a Tabela 44 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 43 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização *Dropout* e otimização ADAM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,99168	0,99056	0,98092	0,98352	0,97660	0,99053

Tabela 44 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização *Dropout* e otimização ADAM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98755	0,97391	0,96881
Background	0,99358	0,98794	0,98439

Na Figura 104 está a matriz de confusão obtida pela rede.

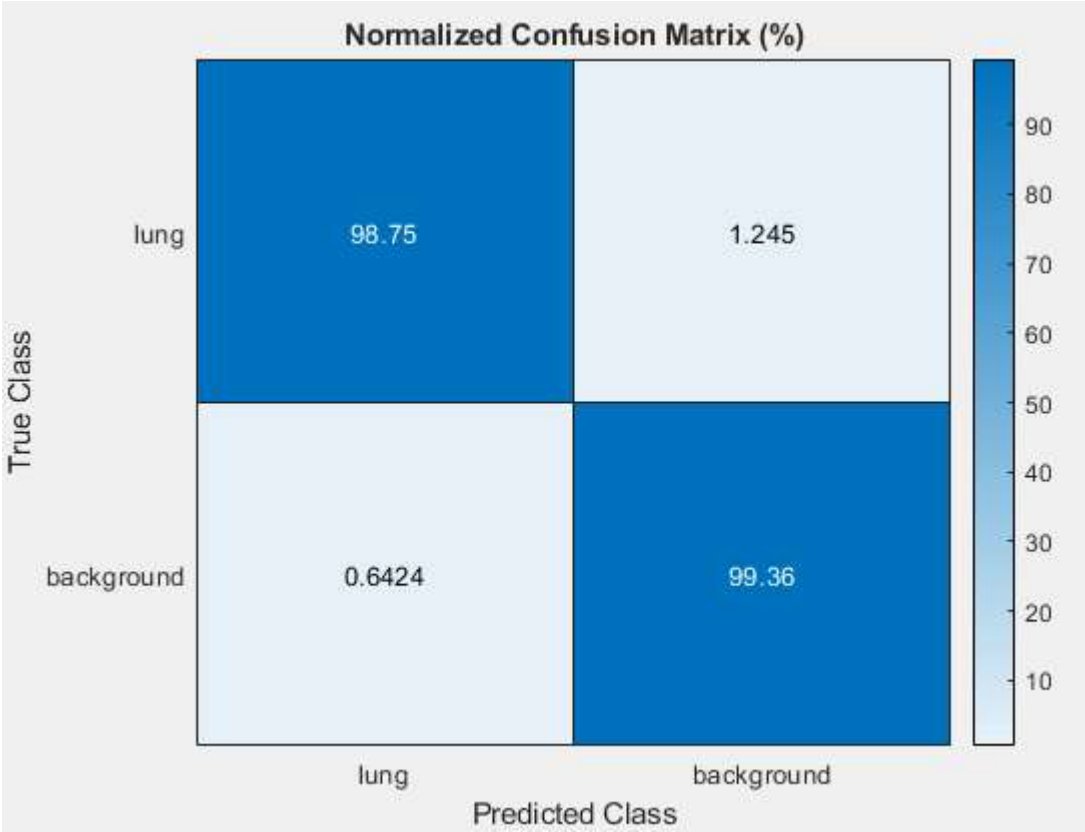


Figura 104 Matriz de confusão para arquitetura CNN3, utilizando regularização *Dropout* e otimização ADAM

A Figura 105 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 103.

Figura 105 Segmentação realizada pela arquitetura CNN3, utilizando regularização *Dropout* e otimização ADAM

As métricas obtidas nesse caso foram sensivelmente superiores a casos anteriores, tendo alcançado aqui o maior valor do índice Dice. O alto valor do Score F1 indica um bom alinhamento entre a bordas do padrão ouro e da imagem obtida pela segmentação da rede.

A vigésima segunda simulação foi realizada na arquitetura CNN3, utilizando o método de regularização L2 e otimização SGDM. A Figura 106 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

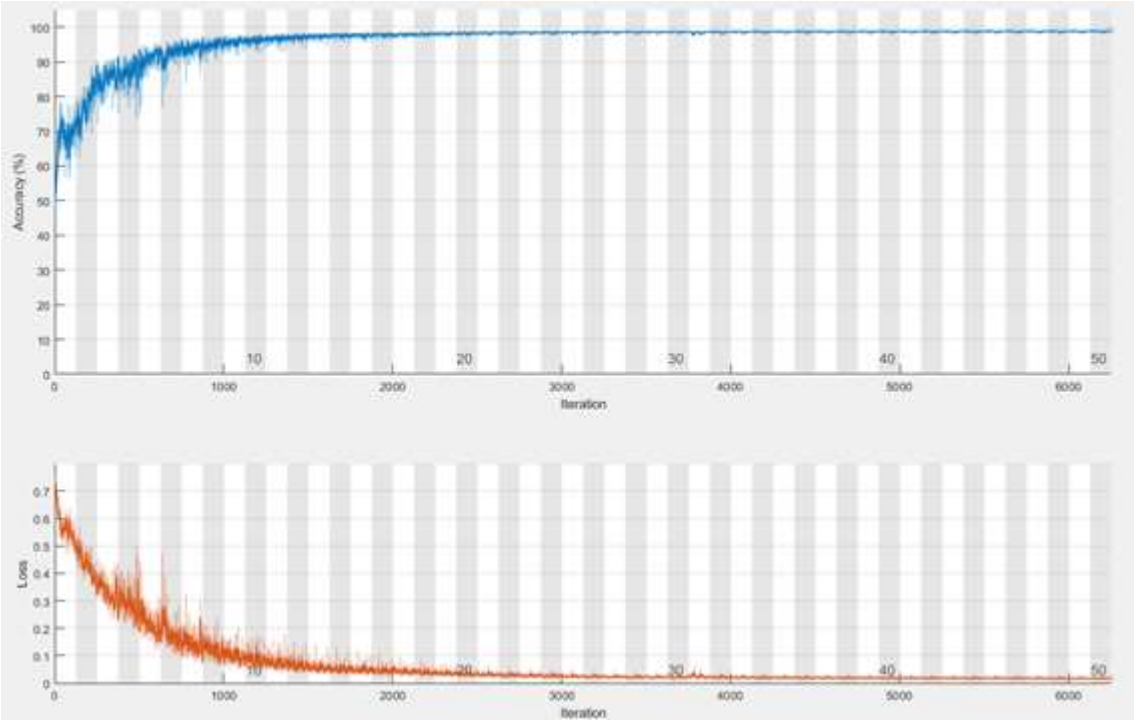


Figura 106 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização L2 e otimização SGDM

A Tabela 45 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 106, durante a fase de validação, enquanto que a Tabela 46 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 45 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização L2 e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,97601	0,96836	0,94567	0,95315	0,89315	0,96770

Tabela 46 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização L2 e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,94776	0,92550	0,85461
Background	0,98897	0,96583	0,94124

Na Figura 107 está a matriz de confusão obtida pela rede.

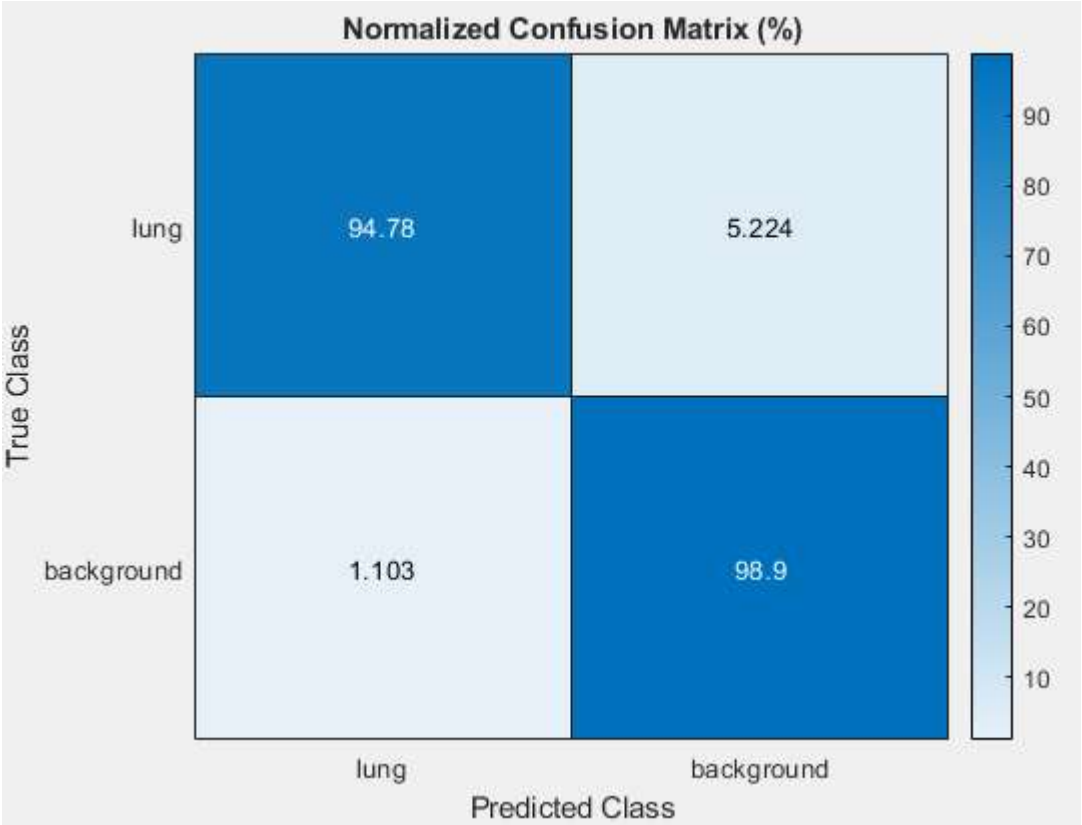


Figura 107 Matriz de confusão para arquitetura CNN3, utilizando regularização L2 e otimização SGDM

A Figura 108 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 106.

Figura 108 Segmentação realizada pela arquitetura CNN3, utilizando regularização L2 e otimização SGDM

A vigésima terceira simulação foi realizada na arquitetura CNN3, utilizando o método de regularização L2 e otimização RMSPROP. A Figura 109 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

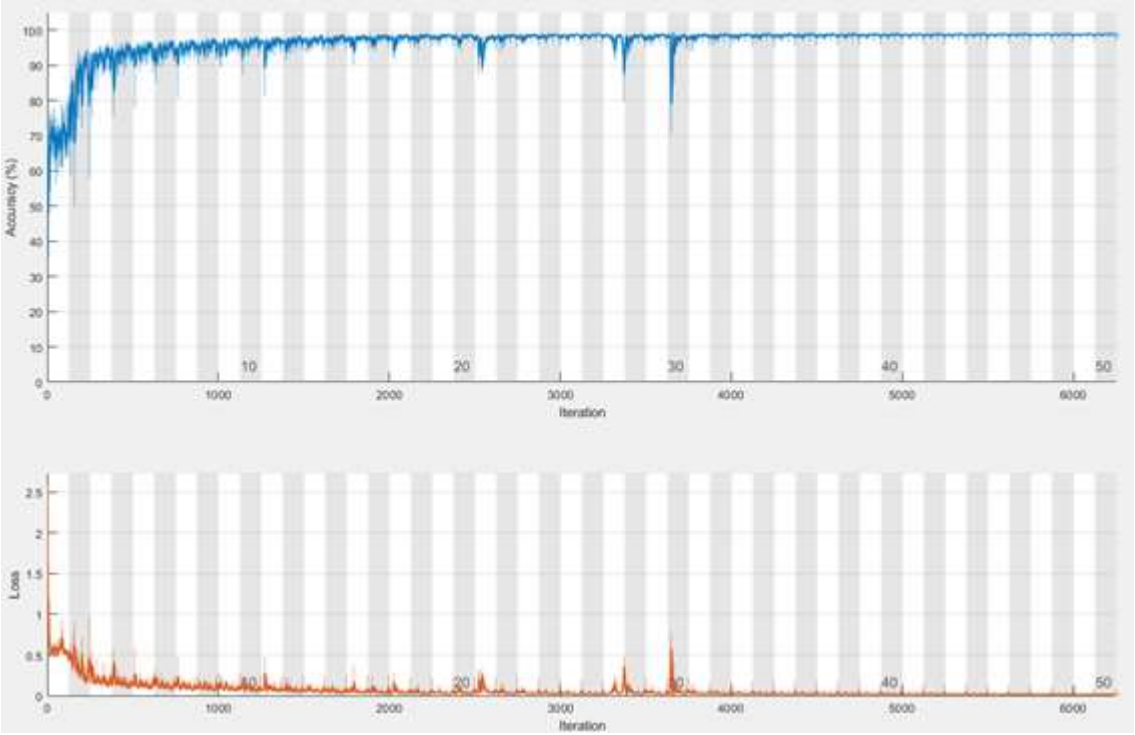


Figura 109 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização L2 e otimização RMSPROP

A Tabela 47 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 109, durante a fase de validação, enquanto que a Tabela 48 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 47 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização L2 e otimização RMSPROP, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,98769	0,98437	0,97182	0,97569	0,95495	0,98423

Tabela 48 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização L2 e otimização RMSPROP, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,97543	0,96140	0,93619
Background	0,99331	0,98224	0,97371

Na Figura 110 está a matriz de confusão obtida pela rede.

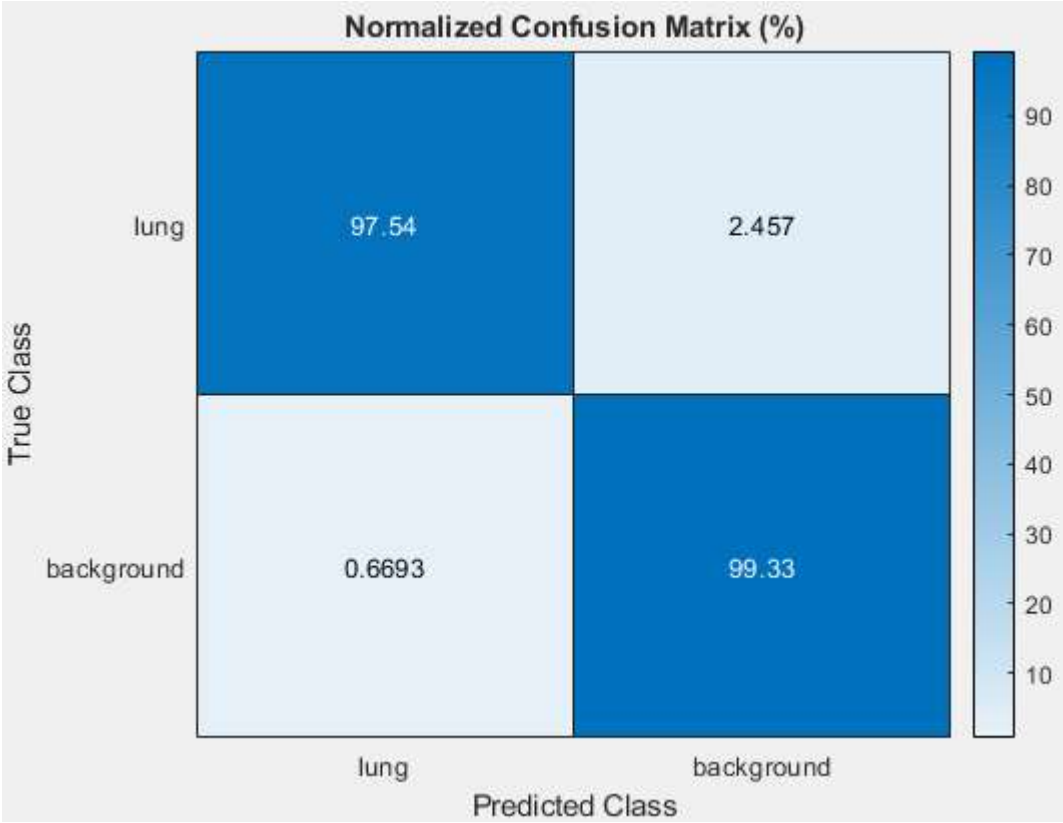


Figura 110 Matriz de confusão para arquitetura CNN3, utilizando regularização L2 e otimização RMSPROP

A Figura 111 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 109.

Figura 111 Segmentação realizada pela arquitetura CNN3, utilizando regularização L2 e otimização RMSPROP

A vigésima quarta simulação foi realizada na arquitetura CNN3, utilizando o método de regularização L2 e otimização ADAM. A Figura 112 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

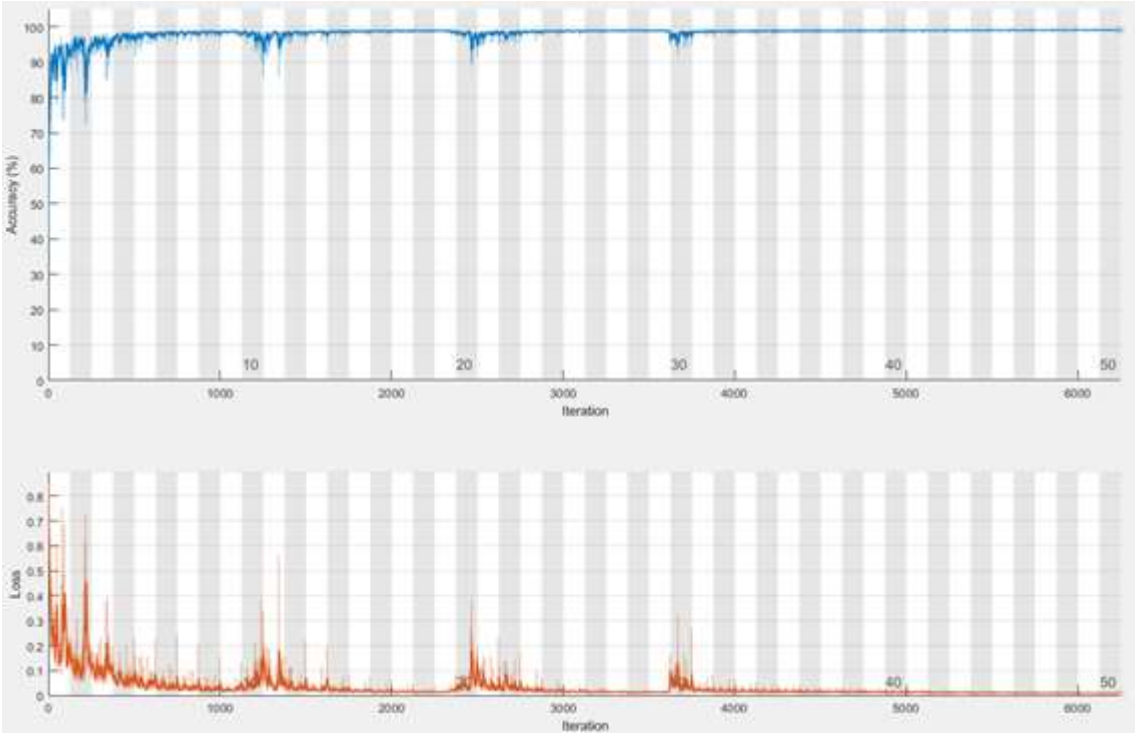


Figura 112 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização L2 e otimização ADAM

A Tabela 49 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 112, durante a fase de validação, enquanto que a Tabela 50 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 49 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização L2 e otimização ADAM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,99152	0,98958	0,98052	0,98319	0,97714	0,98953

Tabela 50 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização L2 e otimização ADAM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98436	0,97333	0,96839
Background	0,99480	0,98772	0,98589

Na Figura 113 está a matriz de confusão obtida pela rede.

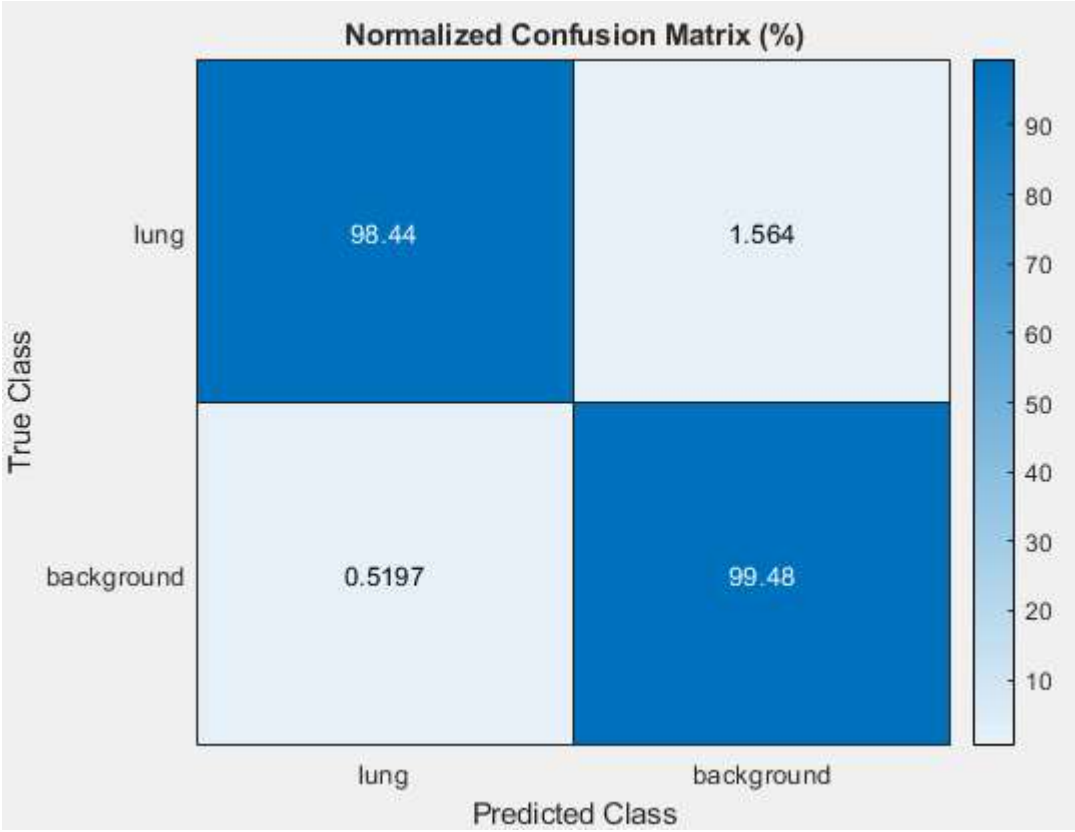


Figura 113 Matriz de confusão para arquitetura CNN3, utilizando regularização L2 e otimização ADAM

A Figura 114 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 112.

Figura 114 Segmentação realizada pela arquitetura CNN3, utilizando regularização L2 e otimização ADAM

A vigésima quinta simulação foi realizada na arquitetura CNN3, utilizando o método de regularização *Dropout*+L2 e otimização SGDM. A Figura 115 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

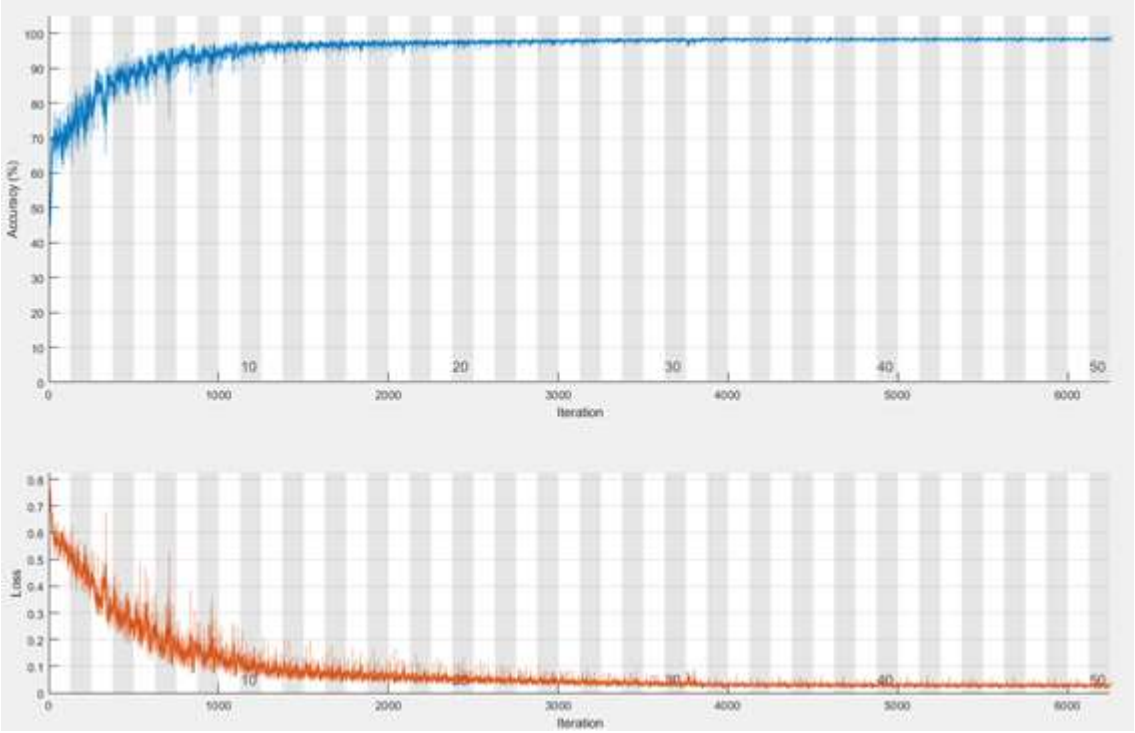


Figura 115 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização SGDM

A Tabela 51 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 115, durante a fase de validação, enquanto que a Tabela 52 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 51 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização SGDM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,97918	0,97069	0,95252	0,95914	0,91298	0,97000

Tabela 52 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização *Dropout*+ L2 e otimização SGDM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,94781	0,93470	0,87097
Background	0,99357	0,97035	0,95499

Na Figura 116 está a matriz de confusão obtida pela rede.

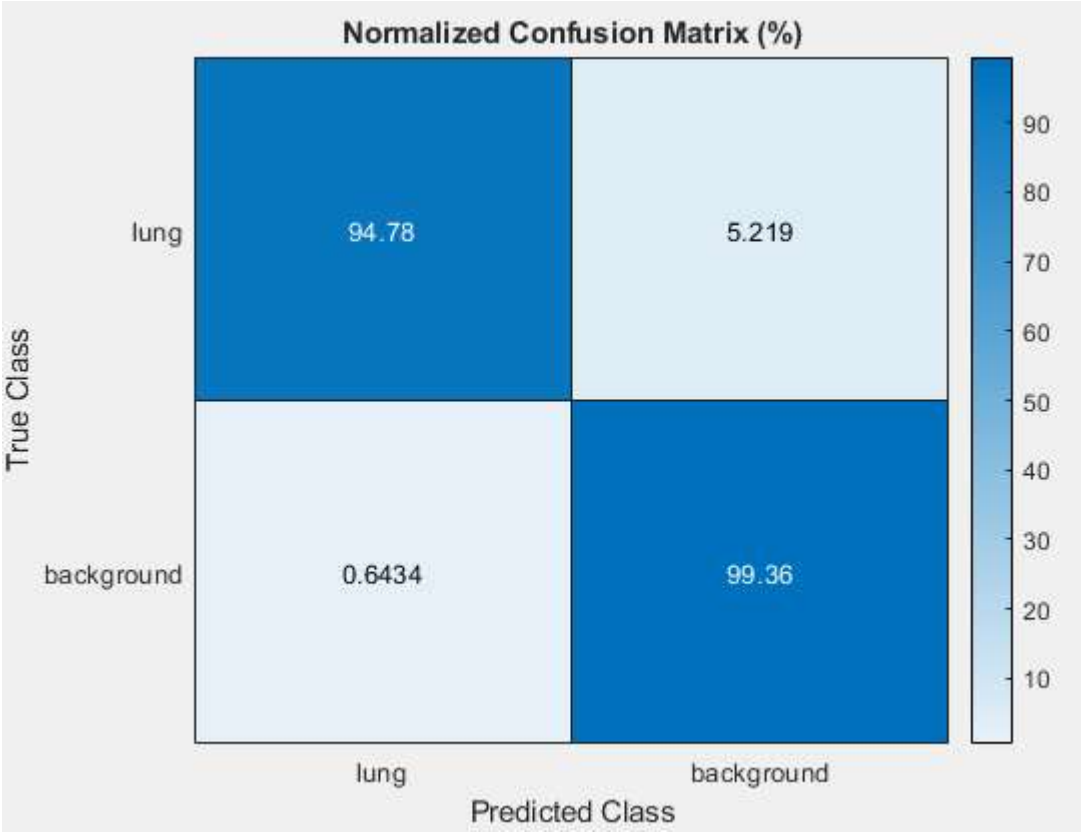


Figura 116 Matriz de confusão para arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização SGDM

A Figura 117 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 115.

Figura 117 Segmentação realizada pela arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização SGDM

A métrica Score F1 apresenta valor baixo, indicando um alinhamento de bordas ruim entre a imagem do padrão ouro e a imagem segmentada pela rede. Na matriz de confusão da Figura 116 nota-se um valor alto para falsos negativos (FN), indicando que muitos pixels pertencentes a classe Pulmão foram erroneamente classificados pela CNN como pertencente a classe *Background*.

A vigésima sexta simulação foi realizada na arquitetura CNN3, utilizando o método de regularização *Dropout*+L2 e otimização RMSPROP. A Figura 118 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

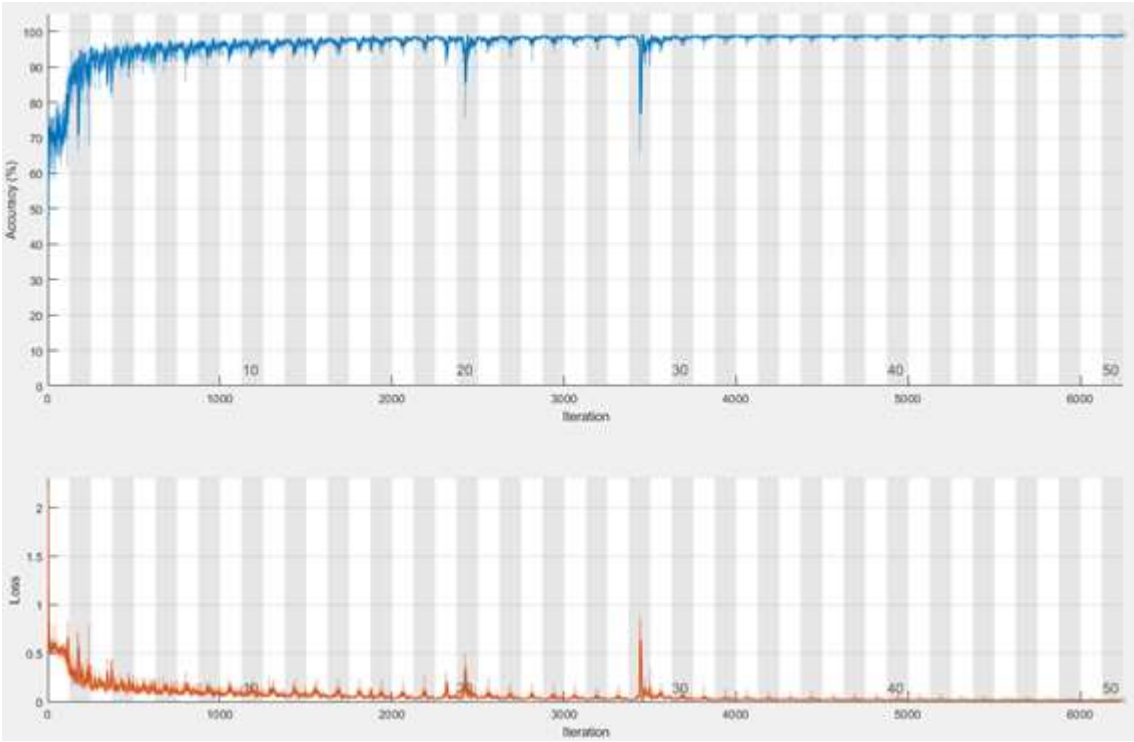


Figura 118 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A Tabela 53 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 118, durante a fase de validação, enquanto que a Tabela 54 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 53 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização RMSPROP, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,97918	0,98930	0,98675	0,97547	0,97884	0,98665

Tabela 54 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização *Dropout*+ L2 e otimização RMSPROP, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,97988	0,96642	0,94731
Background	0,99361	0,98453	0,97826

Na Figura 119 está a matriz de confusão obtida pela rede.

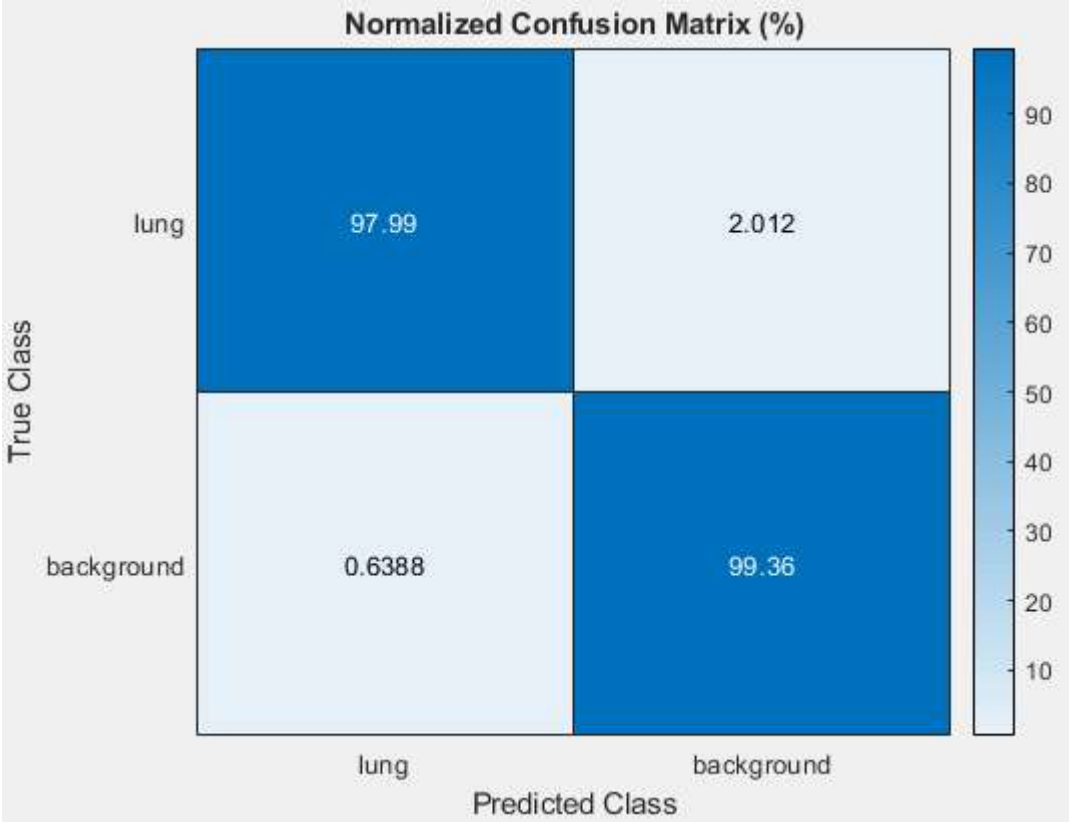


Figura 119 Matriz de confusão para arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A Figura 120 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 118.

Figura 120 Segmentação realizada pela arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização RMSPROP

A vigésima sétima simulação foi realizada na arquitetura CNN3, utilizando o método de regularização *Dropout*+L2 e otimização ADAM. A Figura 121 mostra o comportamento da acurácia da rede CNN3 durante uma seção de treinamento, bem como o gráfico da função de perda.

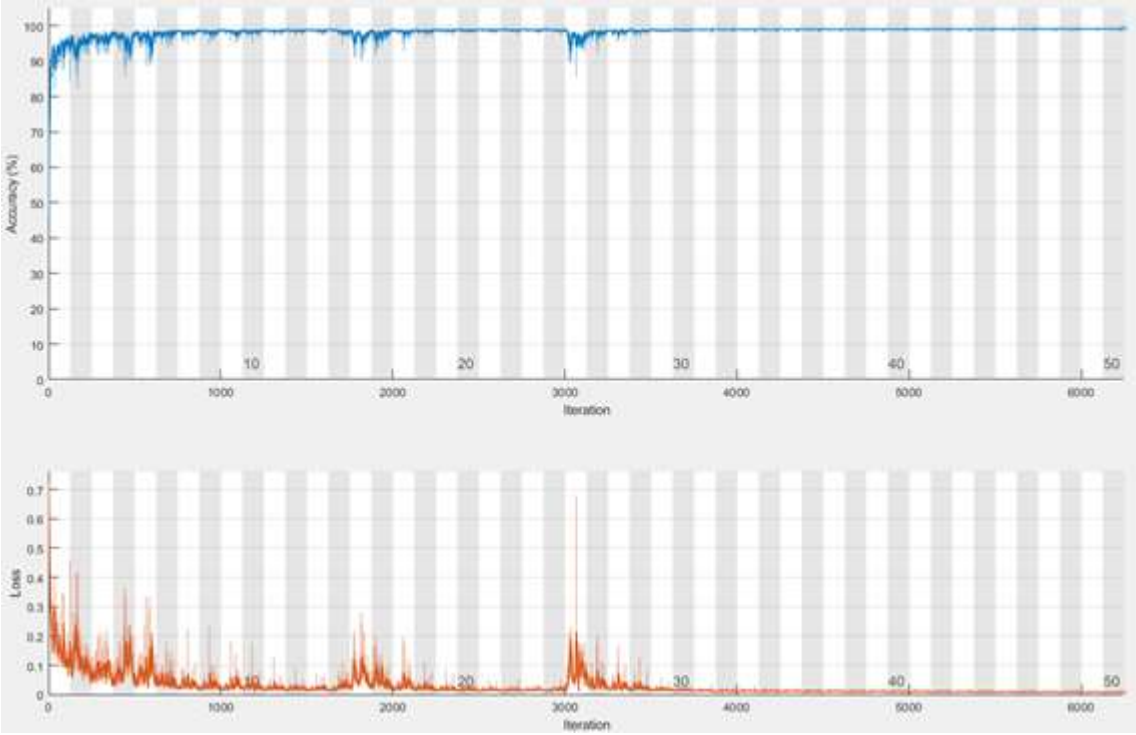


Figura 121 Gráficos da acurácia e função de perda para arquitetura CNN3, durante uma seção de treinamento, utilizando regularização *Dropout*+L2 e otimização ADAM

A Tabela 55 mostra as métricas de desempenho obtidas pela rede, cujo treinamento foi mostrado na Figura 121, durante a fase de validação, enquanto que a Tabela 56 mostra as métricas obtidas por cada classe (*lung* e *background*).

Tabela 55 Métricas de desempenho obtidas com a arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização ADAM, na etapa de validação.

Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
0,97918	0,99180	0,99027	0,98118	0,98375	0,99022

Tabela 56 Métricas de desempenho obtidas por classe com a arquitetura CNN3, utilizando regularização *Dropout*+ L2 e otimização ADAM, na etapa de validação.

	Acurácia	Jaccard (IOU)	Score F1
Pulmão	0,98614	0,97424	0,97074
Background	0,99440	0,98812	0,98672

Na Figura 122 está a matriz de confusão obtida pela rede.

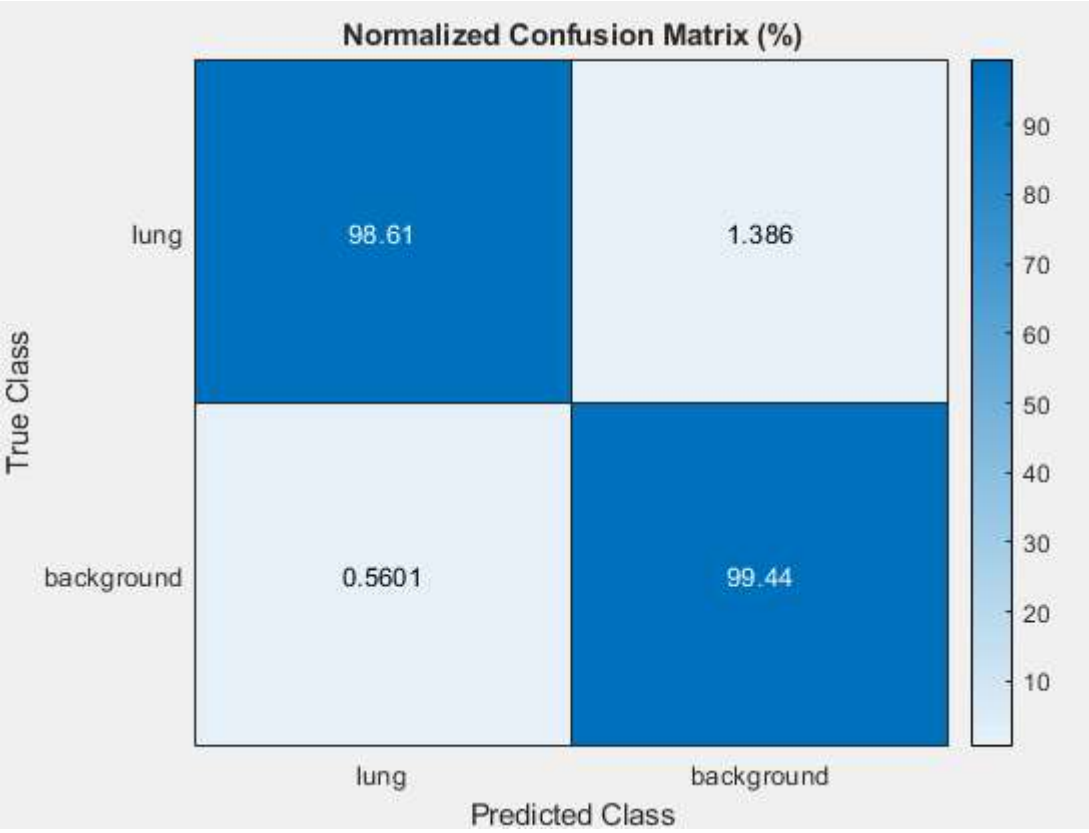


Figura 122 Matriz de confusão para arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização ADAM

A Figura 123 apresenta um exemplo de segmentação realizado pela mesma CNN cujo treinamento foi mostrado na Figura 121.

Figura 123 Segmentação realizada pela arquitetura CNN3, utilizando regularização *Dropout*+L2 e otimização ADAM

Concluídas todas as simulações referentes a arquitetura CNN3, nota-se que algumas redes não realizaram a tarefa de maneira satisfatória, principalmente as que utilizaram SGDM como método de otimização, no entanto, as três simulações utilizando o método de otimização ADAM foram superiores as demais.

5.4 COMPARAÇÃO COM O BENCHMARK

A Tabela 57 mostrada a seguir sintetiza as métricas obtidas com as três arquiteturas de redes convolutivas, na fase de validação.

Tabela 57 Resultados obtidos com o treinamento de arquiteturas usando o conjunto de treinamento com o desempenho avaliado usando o conjunto de validação

Arquitetura - Regularização - Otimização	Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
CNN1 – *Dropout* – SGDM	0,93935	0,95539	0,86247	0,88047	0,52924	0,96375
CNN1 – *Dropout* – RMSPROP	0,98884	0,98704	0,97450	0,97797	0,96889	0,98698
CNN1 – *Dropout* – ADAM	0,98763	0,98684	0,95836	0,96477	0,94156	0,98681
CNN1 – L2 – SGDM	0,98591	0,98426	0,95520	0,96143	0,91755	0,98419
CNN1 – L2 – RMSPROP	0,98900	0,98732	0,97486	0,97828	0,96748	0,98726
CNN1 – L2 – ADAM	0,98820	0,98645	0,96000	0,96578	0,94189	0,98638
CNN1 – *Dropout* + L2 – SGDM	0,93560	0,95270	0,85582	0,87428	0,51559	0,95479
CNN1 – *Dropout* + L2 – RMSPROP	0,98931	0,98860	0,97560	0,97891	0,97327	0,98858
CNN1 – *Dropout* + L2 – ADAM	0,98966	0,98818	0,96346	0,96857	0,95279	0,98813
CNN2 – *Dropout* - SGDM	0,98048	0,98400	0,94230	0,95172	0,89364	0,98415
CNN2 – *Dropout* – RMSPROP	0,98986	0,98970	0,96327	0,96905	0,95366	0,98970
CNN2 – *Dropout* – ADAM	0,98929	0,98913	0,96199	0,96795	0,95405	0,98912
CNN2 – L2 – SGDM	0,98496	0,98171	0,95321	0,95957	0,92118	0,98155
CNN2 – L2 – RMSPROP	0,98878	0,98770	0,96129	0,96692	0,94759	0,98766
CNN2 – L2 – ADAM	0,98610	0,98483	0,95567	0,96180	0,93041	0,98477
CNN2 – *Dropout* + L2 – SGDM	0,97637	0,98129	0,93354	0,94430	0,85673	0,98154
CNN2 – *Dropout* + L2 – RMSPROP	0,98876	0,98826	0,96096	0,96693	0,94699	0,98824
CNN2 – *Dropout* + L2 – ADAM	0,98917	0,98871	0,96184	0,96772	0,94960	0,98870
CNN3 – *Dropout* - SGDM	0,97900	0,97647	0,95267	0,95903	0,91307	0,97631
CNN3 – *Dropout* – RMSPROP	0,99041	0,98849	0,97802	0,98103	0,96777	0,98843
CNN3 – *Dropout* – ADAM	0,99168	0,99056	0,98092	0,98352	0,97660	0,99053
CNN3 – L2 – SGDM	0,97601	0,96836	0,94567	0,95315	0,89792	0,96770
CNN3 – L2 – RMSPROP	0,98769	0,98437	0,97182	0,97569	0,95495	0,98423
CNN3 – L2 – ADAM	0,99152	0,98958	0,98052	0,98319	0,97714	0,98953
CNN3 – *Dropout* + L2 – SGDM	0,97918	0,97069	0,95252	0,95914	0,91298	0,97000
CNN3 – *Dropout* + L2 – RMSPROP	0,98930	0,98675	0,97547	0,97884	0,96278	0,98665
CNN3 – *Dropout* + L2 – ADAM	0,99180	0,99027	0,98118	0,98375	0,97873	0,99022

Analisando as métricas obtidas, nota-se superioridade entre as arquiteturas CNN2 e CNN3, principalmente aquelas que utilizam o método de otimização ADAM. A combinação CNN3 – *Dropout* – ADAM alcançou o melhor resultados em duas métricas, e a combinação CNN3 – *Dropout*+L2 – ADAM alcançou o melhor resultados na outras 4 métricas restantes, mantendo-se ainda com resultados satisfatórios nas restantes. Sendo assim, a combinação **CNN3 – *Dropout*+L2 – ADAM** apresenta-se como a que obteve a melhor performance para a tarefa designada.

Aplicando-se a validação cruzada, conforme descrito na seção 4.3, na arquitetura CNN3, com regularização *Dropout*+L2 e método de otimização ADAM, as métricas obtidas estão mostradas na primeira linha da Tabela 58, juntamente com as métricas obtidas por outros trabalhos da literatura usados como *benchmark*. A arquitetura DAG obteve um coeficiente de Jaccard superior aos obtidos por Ginneken *et. Al*, Chondro *et. Al* e Hwang e Park em seus trabalhos, alcançando assim resultados mais que satisfatórios. Quanto ao índice Dice, o resultado obtido nesse estudo foi superior ao obtido por Novikov *et. Al* e similar ao obtido por Chondro *et. Al*.

Tabela 58 Comparação dos resultados obtidos usando validação cruzada (média ± desvio padrão) com *benchmark*

Rede Neural Convolutiva	Acurácia Global	Acurácia	Jaccard	Jaccard Ponderado	Score F1	Dice
CNN3 – *Dropout* + L2 - ADAM	0,99139 ± 0,00098	0,98927 ± 0,00161	0,97967 ± 0,00232	0,98294 ± 0,00191	0,97475 ± 0,00357	0,98921 ± 0,00163
Chondro et. Al	----------	----------	0,963 ± 0,012	----------	----------	0,983 ± 0,007
Ginneken et. Al	----------	----------	0,938 ± 0,027	----------	----------	----------
Hwang e Park	----------	----------	0,961 ± 0,015	----------	----------	----------
Novikov et. Al	----------	----------	----------	----------	----------	0,95

Analisando as métricas obtidas nesse trabalho com os já existentes na literatura, nota-se que se trata de um método eficiente, visto ter alcançado o atual estado da arte.

A **Figura** 124 mostra a imagem segmentada obtida pela rede após validação cruzada, onde nela mostramos também a borda da imagem padrão ouro em linha preta, facilitando assim a visualização do alinhamento de suas bordas.

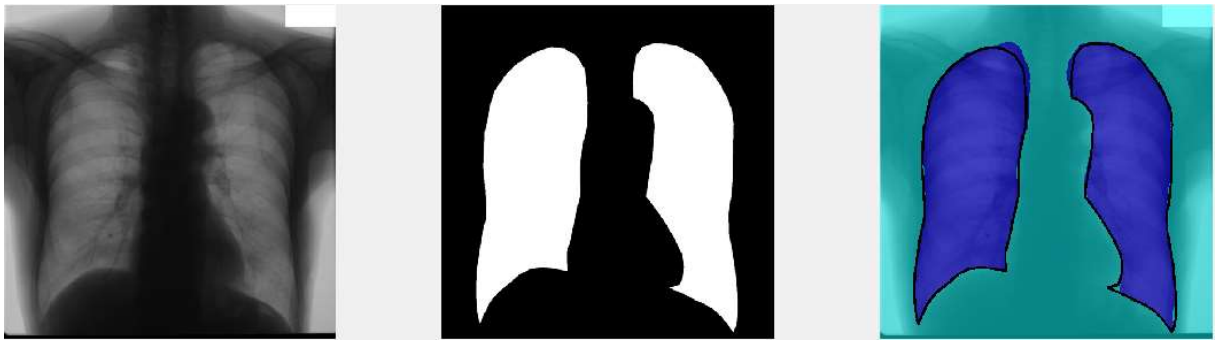


Figura 124 Segmentação realizada pela rede DAG, onde a linha fechada em preto representa a borda da imagem padrão ouro

Ao observar a figura percebe-se um bom alinhamento nas bordas da imagem segmentada e de seu respectivo padrão ouro, bem como uma boa classificação dos pixels em suas respectivas classes, obtendo assim uma segmentação satisfatória. Cabe destacar também que o tempo levado para a CNN realizar a segmentação de forma automática foi da ordem de 1 segundo, mostrando ser um método rápido e eficiente.

6 CONCLUSÕES

Esta dissertação apresentou o desenvolvimento de método automatizado baseado em Aprendizagem Profunda para realizar a tarefa de segmentação da região pulmonar em imagens de radiografia torácica, técnica essa que vem alcançando cada vez mais espaço em estudo envolvendo imagens médicas. Para isso foram utilizadas três arquiteturas, conjuntamente com três métodos de regularização e três métodos de otimização, totalizando um total de 27 simulações. Foi utilizada uma base de dados contendo 247 imagens em escala de cinza, com tamanho original e 2048x2048 pixels e redimensionadas para 512x512 pixels.

Ao observarmos as métricas obtidas nas simulações, podemos dizer que, em sua maioria, obtiveram resultados satisfatórios, principalmente referentes às métricas Coeficiente de Jaccard e Dice, onde grande parte ficaram acima do valor 0,95. Ressalva-se que as piores segmentações foram nos estudos realizados utilizando o método de otimização SGDM, o que não significa ser um método inferior aos outros, mas tão somente o que menos se adequou para a tarefa aqui investigada. Entre as arquiteturas de rede direta e utilizando grafos acíclicos direcionados, obteve-se melhores resultados nas redes DAG, mostrando que a concatenação de informações advindas das camadas de convolução - contendo informações ainda pouco processadas - com informações constando nas camadas de deconvolução se mostraram eficientes nesse estudo.

O método aqui apresentado mostrou-se competitivo com o estado da arte, superando o coeficiente de Jaccard e o índice Dice de alguns estudos anteriores. Destaca-se aqui que uma vantagem do estudo feito aqui é a não necessidade de realizar qualquer tipo de pós-processamento, como aplicação de técnicas morfológicos, ou seja, usou-se apenas o trio "Imagem de entrada → Rede → Imagem de saída", lembrando que o pré-processamento realizado foi por limitação de hardware e não limitação da técnica. Um outro ponto a ser destacado é sua facilidade de realizar a extração de características de forma automática durante a fase de treinamento, usando os mapas de características, eliminando assim o esforço do processo de extração de características da imagem. Deve-se salientar também sua vantagem frente aos métodos morfológicos, pois o estudo aqui realizado trata-se de uma tarefa automatizada.

Como trabalho futuro, pode-se investigar o uso de um outro método de regularização, o *data augmentation*, com o objetivo de se obter um método de segmentação cada vez mais robusto, capaz de lidar com uma gama maior de imagens.

REFERÊNCIAS BIBLIOGRÁFICAS

Cavalcanti, P, G,, Meng, J,, Shirani, S,, Scharcanski, J,, Fong, C,, Castelli, J,, & Koff, D, (2016), Lung nodule segmentation in chest computed tomography using a novel background estimation method, *Quantitative Imaging in Medicine and Surgery*, *6*(1), 16–24, https://doi,org/10,3978/j,issn,2223-4292,2016,02,06

Chondro, P,, Yao, C, Y,, Ruan, S, J,, & Chien, L, C, (2018), Low order adaptive region growing for lung segmentation on plain chest radiographs, *Neurocomputing*, *275*, 1002–1011, https://doi,org/10,1016/j,neucom,2017,09,053

Ginneken, B,, Stegmann, M,, & Loog, M, (2005), *Segmentation od anatomical structures in chest radiographs using supervised methods: a comparative study on a public database*,

Goodfellow, I,, Bengio, Y,, & Courville, A, (2016), *Deep Learning*, Retrieved from https://books,google,com,br/books?id=Np9SDQAAQBAJ

Gordienko, Y,, Gang, P,, Hui, J,, Zeng, W,, Kochura, Y,, Alienin, O,, … Stirenko, S, (2019), Deep learning with lung segmentation and bone shadow exclusion techniques for chest X-ray analysis of lung cancer, *Advances in Intelligent Systems and Computing*, *754*, 638–647, https://doi,org/10,1007/978-3-319-91008-6_63

Hwang, S,, & Park, S, (2017), Accurate lung segmentation via network-wise training of convolutional networks, *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10553 LNCS*, 92–99, https://doi,org/10,1007/978-3-319-67558-9_11

INCA, (2018), Instituto Nacional de Câncer-INCA, Estimativas da incidência e mortalidade por câncer, In *Ministério da Saúde,* Retrieved from http://www,inca,gov,br/estimativa/2018/estimativa-2018,pdf

Islam, J,, & Zhang, Y, (2018), *Towards Robust Lung Segmentation in Chest Radiographs with Deep Learning*, Retrieved from http://arxiv,org/abs/1811,12638

LeCun, Y,, Haffner, P,, Bottou, L,, & Bengio, Y, (1998), *Object Recognition with Gradient-Based Learning*, (0),

McCulloch, W,, & Pitts, W, (1943), *A logical calculus of the ideas immanent in nervous activity, Originally published in: Bulletin of Mathematical Biophysics, Vol, 5, 1943, p, 115-133*, *5*, 115–133,

Miyagawa, M,, Costa, M, G, F,, Gutierrez, M, A,, Costa, J, P, G, F,, & Filho, C, F, F, C, (2019), Detecting Vascular Bifurcation in IVOCT Images Using Convolutional Neural Networks With Transfer Learning, *IEEE Access*, *7*(c), 66167–66175, https://doi,org/10,1109/ACCESS,2019,2918017

Novikov, A, A,, Lenis, D,, Major, D,, Hladuvka, J,, Wimmer, M,, & Buhler, K, (2018), Fully Convolutional Architectures for Multiclass Segmentation in Chest Radiographs, *IEEE Transactions on Medical Imaging*, *37*(8), 1865–1876, https://doi,org/10,1109/TMI,2018,2806086
OMS, (2018), Retrieved from https://www,who,int/es

Ronneberger, O,, Fischer, P,, & Brox, T, (2015), U-net: Convolutional networks for biomedical image segmentation, *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9351*, 234–241, https://doi,org/10,1007/978-3-319-24574-4_28

Shiraishi, J,, Katsuragawa, S,, Ikezoe, J,, Matsumoto, T,, Komatsu, K,, Matsui, M,, … Doi, K, (1999), *Development of a Digital Image Database for Chest Radiographs with and without a Lung Nodule: Receiver Operating Characteristic Analysis of Radiologists' Detection of Pulmonary Nodules*,

Silva, I, N, da, Spatti, D, H,, & Flauzino, R, A, (2010), *Redes Neurais Artificiais para engenharia e ciências aplicadas*, Retrieved from https://artliber,com,br/index,php?route=product/search&search=redes neurais artificiais

Skourt, B,, Hassani, A,, & Majda, A, (2018), *Lung CT Image Segmentation Using Deep Neural Networks* (pp, 109–113), pp, 109–113,

Uzelaltinbulat, S,, & Ugur, B, (2017), Lung tumor segmentation algorithm, *Procedia Computer Science*, *120*, 140–147, https://doi,org/10,1016/j,procs,2017,11,221

Yassine, B,, Taylor, P,, & Story, A, (2018), Fully automated lung segmentation from chest radiographs using SLICO superpixels, *Analog Integrated Circuits and Signal Processing*, *95*(3), 423–428, https://doi,org/10,1007/s10470-018-1153-1

TRABALHO ACEITO PARA PUBLICAÇÃO NO 42nd ANNUAL INTERNATIONAL CONFERENCES OF THE IEEE ENGINEERING IN MEDICINE AND BIOLOGY.

Lung Region Segmentation in Chest X-Ray Images using Deep Convolutional Neural Networks

R. D. S. Portela, J. R. G. Pereira, M. G. F. Costa, *Member, IEEE*, C. F. F. Costa Filho, *Member, IEEE*,

***Abstract*— Lung cancer is, by far, the leading cause of cancer death in the world. Tools for automated medical imaging analysis development of a Computer-Aided Diagnosis method comprises several tasks. In general, the first one is the segmentation of region of interest, for example, lung region segmentation from Chest X-ray imaging in the task of detecting lung cancer. Deep Convolutional Neural Networks (DCNN) have shown promising results in the task of segmentation in medical images. In this paper, to implement the lung region segmentation task on chest X-ray images, was evaluated three different DCNN architectures in association with different regularization (Dropout, L2, and Dropout + L2) and optimization methods (SGDM, RMSPROP and ADAM). All networks were applied in the Japanese Society of Radiological Technology (JSRT) database. The best results were obtained using Dropout + L2 as regularization method and ADAM as optimization method. Considering the Jaccard Coefficient obtained (0.97967 ± 0.00232) the proposal outperforms the state of the art.**

***Clinical Relevance*— The presented method reduces the time that a professional takes to perform lung segmentation, improving the effectiveness.**

I. Introduction

According to the World Health Organization (WHO), cancer is the second leading cause of death worldwide, and is responsible for an estimated 9.6 million deaths in 2018. Lung cancer is the most common cause of cancer death when compared to the number of deaths caused by any other cancer type [1]. Medical imaging, such as radiography, assists the physician in decision make about diagnosis and treatment of lung cancer. Computer-Aided Lung Cancer Diagnosis methods may require the segmentation of lung tissue in the radiographic image. Therefore, using tools that apply techniques for automatic segmentation can greatly save effort and help in obtaining more accurate diagnosis. There are several studies published in the literature aiming lung region segmentation in chest X-ray images, using different methods. Chondro et al. [2] used digital imaging techniques, such as histogram equalization and Gaussian filter, to segment the lung region on chest X-ray images. Yassine et al. [3] performed pulmonary segmentation using the Superpixels technique. In recent years, Deep Convolutional Networks, one of the most important Deep Learning Network, have become increasingly prominent in tasks involving medical imaging, such as segmentation, detection, etc., due to the its superiority over other methods used so far in these tasks. Deep Learning is a particular type of machine learning, which uses multiple layers of processing to achieve high levels of abstraction when learning through data representations. Gordienko et al. [4] used U-Net architecture to investigate the ability of network to segment the lung region in radiographic images with and without its bone structures. Islam and Zhang [5] used convolutive networks to segment in the Montgomery and Shenzhen databases. Mayan et al. [6] used the architecture U-Net with an ImageNet Pre-trained encoder.

This study aims to evaluate the ability of Convolutional Networks to perform the task of lung segmentation in chest X-ray images. The article is organized as follows: Section II presents the dataset used, the implemented architectures, as well as their training parameters. Also, section II shows the metrics used for segmentation evaluation. Section III presents the results obtained from simulations and the discussion.

II. Methodology

A. Dataset

The Japanese Society of Radiological Technology (JSRT) database [7] was chosen to evaluate the Deep Convolutional Neural Network (DCNN) models. It is an annotated image database containing 247 chest X-ray images (154 with a single pulmonary nodule and 93 with no pulmonary nodules) with their respective gold standard segmented lung regions. All images in the database are gray level and have 2048x2048 pixels. Fig. 1 shows two examples of database images, one with a pulmonary nodule and another without a pulmonary nodule. To each Chest X-ray image there are two gold standard images: one corresponding to the segmented region of the left lung and the other corresponding to the segmented region of

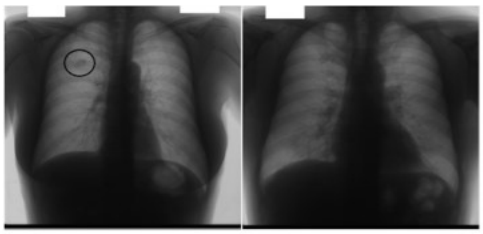
* This research, according for in Article 48 of Decree nº 6.008/2006, was funded by Samsung Electronics of Amazonia Ltda, under the terms of Federal Law nº 8.387/1991, through agreement nº 004, signed with the Center for R&D in Electronics and Information from the Federal University of Amazonas - CETELI/UFAM..

The authors are with Federal University of Amazonas, Amazonas, Brazil (e-mails: portela.ronaldo@hotmail.com, mcosta@ufam.edu.br, ccosta@ufam.edu.br, josergpereira@gmail.com)
.

the right lung. To perform convolutional neural network training, it was necessary to fusion the two images. Fig. 2 shows gold standard images of the images shown in Fig. 1, after the fusion operation. Due to hardware memory limitation, it was necessary to resize all images, as well as their respective gold standards, to a resolution of 512x512 pixels.

B. Deep Convolutional Neural Network Architectures and Training Parameters

Miyagawa et al. [8] used three architectures to perform lumen segmentation in IVOCT images. Int his paper we use these same architectures for segmentation of the pulmonary region. The first architecture proposed (DCNN-1) is a semantic network. This network consists of 51 layers. Like all semantic networks, it is made up of subsampling layers followed by oversampling layers. The network contains 4 subsampling modules. Each module contains two sequence of the following layers: convolution operation, batch normalization operation and ReLu operation followed by a maxpooling operation. In the first subsample layer, 32 feature maps were used, while the remaining layers used 64 feature maps. All convolution filters are 3x3 in size and have zero padding at the edges, so, after the convolution layer, the image dimension does not change. All filters used in subsampling are 2x2 in size with horizontal and vertical steps equal to 2. After the subsampling layer, its size will be reduced by a factor of 2, both horizontally and vertically. In the sequence, follow 4 layers of oversampling, so that the image retrieves the dimension lost in the previous steps and returns to its original size. In all oversampling steps, 4x4 filters were used, with horizontal and vertical steps equal to 2. In the first three oversampling layers, 64 feature maps were used, while, in the fourth oversampling layer, a 32 feature map was used. Subsequently, the image went through a 1x1 convolution layer, where its volume is equal 2 (lung and background). Finally, there is a classification layer. In this layer, a Softmax function classifies each pixel oh the image as belonging to the lung or background. DCNN-1 is shown in Fig. 3(a).



(a) (b)

Figure 1. Example of Chest X-ray images from the database. (a) Image with a nodule (b) Image without a nodule.

(a) (b)

Figure 2. Gold Standard segmented images corresponding to images shown in Figure 1.

The second architecture (DCNN-2) is also a direct network but has Convolution, Batch and ReLu layers between its transposed convolution layers, totaling 75 layers in its structure. ReLu layers added between the transposed convolution steps should give greater nonlinearity and abstraction capability during each oversampling step. DCNN-2 is shown in Fig. 3(b).

The third architecture (DCNN-3) is a network applying Directed Acyclic Graphs (DAG). In this type of architecture, information from the initial subsampling steps is passed on to the final oversampling steps. The information that is passed from the initial layers is linked to the information contained in the final oversampling layers and, as they have the same dimensions, can be connected without any size adjustments being required, as shown in Fig. 3(c).

The database was divided into three sets: training, validation and testing, in the following proportion: 50% - 25% - 25%, respectively. Also, it was maintained the same proportion for images with and without nodules. For example, to compose the training set, 50% of the images without nodule and 50% of the images with nodule were selected.

Dropout, L2 and Dropout + L2 was chosen as regularization methods and SGDM, RMSPROP and ADAM was chosen as optimization methods. In a first step, to choose the best network, the training and validation set was used. After choosing the best network, cross-validation with five folders between the training and test sets was performed.

For this work, it was used a computer with Windows 10 operating system, Intel Core I & -8700 CPU @3.20GHz 3.19 GHz processor, 16 GB of RAM and 8GB NVIDIA GeForce GTX 1070 GPU. The environment development was MATLAB R2019a. Parameter values for DCNNs training were adjusted experimentally. Table I show these values.

C. Evaluation metrics

The following metrics were used to evaluate the networks: Global Accuracy, Accuracy, Jaccard Coefficient, Weighted Jaccard Coefficient, Dice Index and Score F1.

III. Results and Discussion

Combining the three architectures, the three regularization methods and the three optimization methods, result in 27 experiments. Table II shows the results of the 27 simulations performed using the training set and the performance obtained using the validation set. The objective was to select the DCNN with best performance in the validation set. As observed, DCNN-1 architecture presents low values for F1 Score, an index that indicates how well the edges of each segmented region align with the edges of the respective gold standard. DCNN-2 architecture outperforms DCNN-1. It is due to the Convolution, Batch and ReLu layers that were inserted between the transposed convolution layers. The best simulation results, however, were obtained with DCNN-3 using Dropout + L_2 regularization in association with the ADAM optimization method. After selecting the best DCNN model, we performed the 5-folder cross-validation using the training and testing sets. The obtained metric values (average ± standard deviation) are shown in Table III. Fig. 4 exemplify

the result of a segmented chest X-ray image and presents its respective gold standard. Additionally, a comparison of the results obtained with the existing works in the literature using the same database was made. In [9], a Jaccard Coefficient of 0.903 ± 0.057 was obtained. In [2], a Jaccard Coefficient of 0.963 ± 0.012 and Dice Index of 0.983 ± 0.007 were obtained. In [10], a Jaccard Coefficient of 0.961 ± 0.015 was obtained. As shown in Table 3, in this work, it was obtained a Jaccard Coefficient of 0.97967 ± 0.00232, a Dice Index of 0.98921 ± 0.00163 and F1 Score of 0.97475 ± 0.00357.

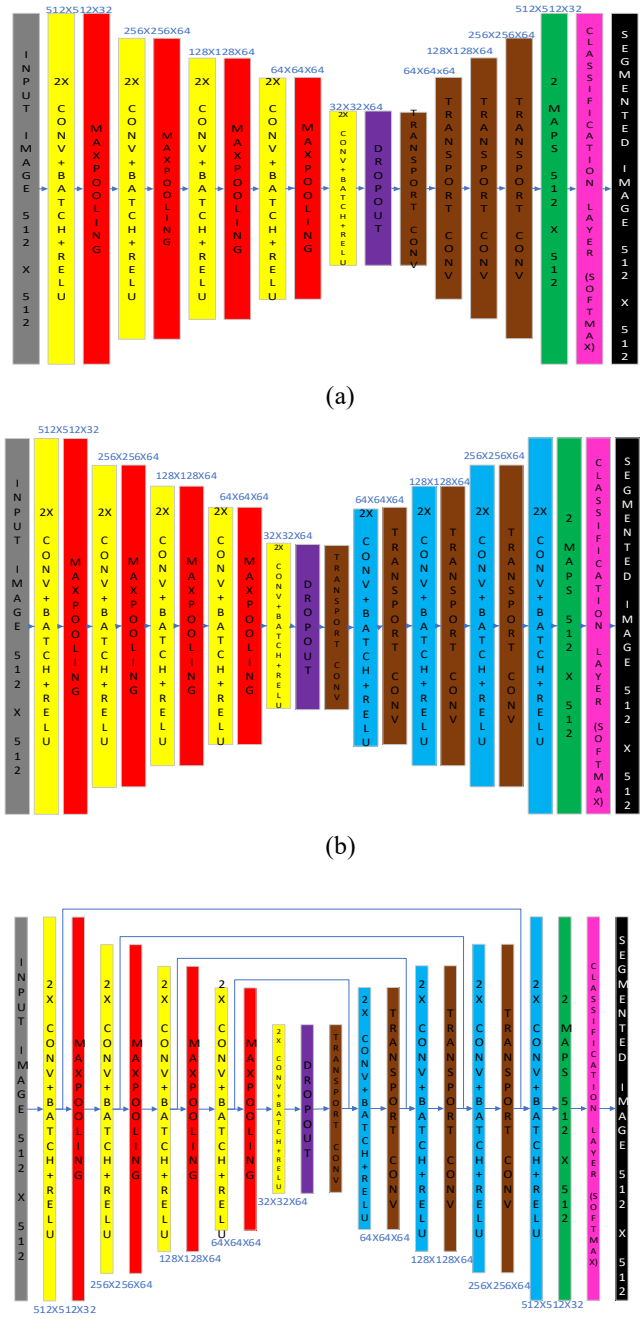


Figure 3. (a) First Architecture (DCNN-1) – Direct Network, (b) Second Architecture (DCNN-2) – Direct network and (c) Third Architecture Network (DCNN-3) – Directed Acyclic Graphs

Table I. Parameters used in the network training step

Parameter	Value
Initial Learn Rate (η)	10^{-3}
Learn Rate Drop Factor	0.1
Learn Rate Drop Period	30
Mini Batch Size	1
Max Epochs	50
Squared Gradient Decay Factor (β_2)	0.999
Gradient Decay Factor (β_1)	0.9
Momentum (α)	0.9
Epsilon (ε)	10^{-8}
L_2 Regularization (λ)	10^{-4}
Dropout	50%

When comparing with other works published in the literature, we realize that the results obtained in this work are in accordance with the state of the art until then. This paper helps to highlight the great performance of Deep Convolutional Neural Network in medical images segmentation tasks, and particularly confirms the remarkable performance of DAG networks in segmentation tasks, as previously observed by Miyagawa et al. [8], showing that they can be used to assist the medical professional.

The great advantage of using DCNNs in the lung segmentation, is that there is no need to perform pre or post processing steps, such as threshold application, morphological filters application, histogram equalization etc, because these previous steps are intrinsic to the DCNN.

REFERENCES

[1] WHO. 2018. World Health Organization – WHO, Folha informative – Câncer. Retrieved from https://www.paho.org/bra/index.php?option=com_content&view=article&id=5588:folha-informativa-cancer&Itemid=1094

[2] P. Chondro, C. Y. Yao, S. J. Ruan, and L. C. Chien, "Low order adaptive region growing for lung segmentation on plain chest radiographs". *Neurocomputing,* 275, 1002–1011, September 2017.

[3] B. Yassine, P. Taylor, and A. Story, "Fully automated lung segmentation from chest radiographs using SLICO superpixels". *Analog Integrated Circuits and Signal Processing*, 95(3), 423–428, March 2018.

[4] Y. Gordienko, P. Gang, J. Hui, W. Zeng, Y. Kochura, O. Alienin, S. Stirenko, "Deep learning with lung segmentation and bone shadow exclusion techniques for chest X-ray analysis of lung cancer". *International Conference on Computer Science, Engineering and Education Applications*, 638–647. Springer, Cham. January 2018.

[5] J. Islam, and Y. Zhang. "Towards Robust Lung Segmentation in Chest Radiographs with Deep Learning". *Machine Learning for Health (ML4H) Workshop at NeurIPS*. November 2018. arXiv preprint arXiv: 1811.12638.

[6] M. Frid-Adar, A. Ben-Cohen, R. Amer, and H. Greenspan, "Improving the segmentation of anatomical structures in chest radiographs using U-net with an ImageNet pre-trained encoder". *Image Analysis for Moving Organ, Breast, and Thoracic Images. RAMBO 2018, BIA 2018, TIA 2018 Lectures Notes in Computer Science, vol 11040.* Springer, Cham. September 2018.

[7] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, K. Komatsu, M. Matsui, … K. Doi (1999). "Development of a Digital Image Database for Chest Radiographs with and without a Lung Nodule: Receiver Operating Characteristic Analysis of Radiologists' Detection of Pulmonary Nodules", *American Journal of Roentgenology, 174(1), pp.71-74. Annual meeting of the Radiological Society of North America, Chicago*. November 1999.

[8] M. Miyagawa, M. G. F. Costa, and C. F. F. Costa Filho, "Lumen Segmentation in optical coherence tomography images using

convolutional neural network", *Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Honolulu-USA*. July 2018.

[9] B. Ginneken, M. Stegmann, and M. Loog, "Segmentation of anatomical structures in chest radiographs using supervised methods: a comparative study on a public database", *Medical Image Analysis*, 10(1), pp.19-40. 2006.

[10] S. Hwang, and S. Park, "Accurate lung segmentation via network-wise training of convolutional networks". *In 3rd Workshop on Deep Learning in Medical Image Analysis*, Quebec. August 2017. arXiv preprint arXiv: 1708.00710

[11] A. A. Novikov, D. Lenis, D. Major, J. Hladûvka, M. Wimmer, and K. Buhler, "Fully convencional architectures for multi-class segmentation in chest radiographs". *IEEE transactions on medical imaging*, 37(8), pp.1865-1876. February 2018.

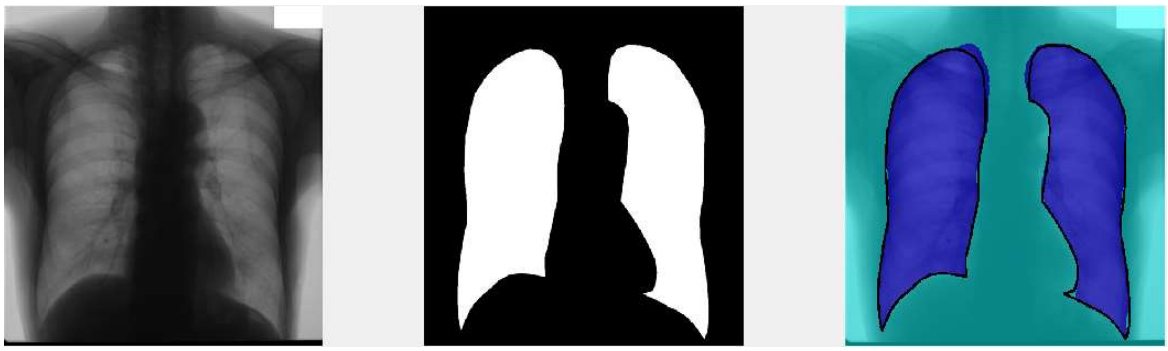
[12] B. A. Skourt, A. Hassani, A. majda, "Lung CT image segmentation using deep neural networks", *The First Internacional Conference On intelligent Computing in Data Sciences*, pp. 109 – 113. Morocco, 2018.

Table II. Results obtained from the training of architectures using the training set with the performance evaluated using the validation sets

Architecture - Regularization - Optimization	Global Accuracy	Accuracy	Jaccard	Weighted Jaccard	Score F1	Dice
DCNN1 – Dropout – SGDM	0.93935	0.95539	0.86247	0.88047	0.52924	0.96375
DCNN1 – Dropout – RMSPROP	0.98884	0.98704	0.97450	0.97797	0.96889	0.98698
DCNN1 – Dropout – ADAM	0.98763	0.98684	0.95836	0.96477	0.94156	0.98681
DCNN1 – L_2 – SGDM	0.98591	0.98426	0.95520	0.96143	0.91755	0.98419
DCNN1 – L_2 – RMSPROP	0.98900	0.98732	0.97486	0.97828	0.96748	0.98726
DCNN1 – L_2 – ADAM	0.98820	0.98645	0.96000	0.96578	0.94189	0.98638
DCNN1 – Dropout + L_2 – SGDM	0.93560	0.95270	0.85582	0.87428	0.51559	0.95479
DCNN1 – Dropout + L_2 – RMSPROP	0.98931	0.98860	0.97560	0.97891	0.97327	0.98858
DCNN1 – Dropout + L_2 – ADAM	0.98966	0.98818	0.96346	0.96857	0.95279	0.98813
DCNN2 – Dropout - SGDM	0.98048	0.98400	0.94230	0.95172	0.89364	0.98415
DCNN2 – Dropout – RMSPROP	0.98986	0.98970	0.96327	0.96905	0.95366	0.98970
DCNN2 – Dropout – ADAM	0.98929	0.98913	0.96199	0.96795	0.95405	0.98912
DCNN2 – L_2 – SGDM	0.98496	0.98171	0.95321	0.95957	0.92118	0.98155
DCNN2 – L_2 – RMSPROP	0.98878	0.98770	0.96129	0.96692	0.94759	0.98766
DCNN2 – L_2 – ADAM	0.98610	0.98483	0.95567	0.96180	0.93041	0.98477
DCNN2 – Dropout + L_2 – SGDM	0.97637	0.98129	0.93354	0.94430	0.85673	0.98154
DCNN2 – Dropout + L_2 – RMSPROP	0.98876	0.98826	0.96096	0.96693	0.94699	0.98824
DCNN2 – Dropout + L_2 – ADAM	0.98917	0.98871	0.96184	0.96772	0.94960	0.98870
DCNN3 – Dropout - SGDM	0.97900	0.97647	0.95267	0.95903	0.91307	0.97631
DCNN3 – Dropout – RMSPROP	0.99041	0.98849	0.97802	0.98103	0.96777	0.98843
DCNN3 – Dropout – ADAM	0.99168	0.99056	0.98092	0.98352	0.97660	0.99053
DCNN3 – L_2 – SGDM	0.97601	0.96836	0.94567	0.95315	0.89792	0.96770
DCNN3 – L_2 – RMSPROP	0.98769	0.98437	0.97182	0.97569	0.95495	0.98423
DCNN3 – L_2 – ADAM	0.99152	0.98958	0.98052	0.98319	0.97714	0.98953
DCNN3 – Dropout + L_2 – SGDM	0.97918	0.97069	0.95252	0.95914	0.91298	0.97000
DCNN3 – Dropout + L_2 – RMSPROP	0.98930	0.98675	0.97547	0.97884	0.96278	0.98665
DCNN3 – Dropout + L_2 – ADAM	0.99180	0.99027	0.98118	0.98375	0.97873	0.99022

Table III. Result obtained using cross-validation (average ± standard deviation)

Deep Convolutional Network	Global Accuracy	Accuracy	Jaccard	Weighted Jaccard	Score F1	Dice
DCNN-3 – Dropout + L_2 - ADAM	0.99139 ± 0.00098	0.98927 ± 0.00161	0.97967 ± 0.00232	0.98294 ± 0.00191	0.97475 ± 0.00357	0.98921 ± 0.00163



(a) (b) (c)

Fig.4. Segmentation obtained by Neural Network DCNN-3, using Dropout + L2 regularization and ADAM optimization (a) left and right lung image, (b) gold standard images and (c) segmented images with gold standard contour superimposed as a black line.