UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Tese

Aprendizagem de Máquina em Análise de Expressão Gênica - Classificação e Seleção de Genes Relevantes em Câncer

Gisele Moraes Simas

Pelotas, 2022

Gisele Moraes Simas

Aprendizagem de Máquina em Análise de Expressão Gênica - Classificação e Seleção de Genes Relevantes em Câncer

Tese apresentada ao Programa de Pós-Graduação em Computação do Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

Orientador: Prof. Dr. Ricardo Matsumura de Araújo
Coorientadora: Dra. Marialva Sinigaglia

Pelotas, 2022

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

S588a Simas, Gisele Moraes

Aprendizagem de máquina em análise de expressão gênica : classificação e seleção de genes relevantes em câncer / Gisele Moraes Simas ; Ricardo Matsumura de Araújo, orientador ; Marialva Sinigaglia, coorientadora. — Pelotas, 2022.

199 f. : il.

Tese (Doutorado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2022.

1. Deep learning. 2. Redes neurais. 3. CNN. 4. GAN. 5. Dropout. I. Araújo, Ricardo Matsumura de, orient. II. Sinigaglia, Marialva, coorient. III. Título.

CDD : 005

Elaborada por Simone Godinho Maisonave CRB: 10/1733

Gisele Moraes Simas

Aprendizagem de Máquina em Análise de Expressão Gênica - Classificação e Seleção de Genes Relevantes em Câncer

Tese aprovada, como requisito parcial, para obtenção do grau de Doutor em Ciência da Computação, Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 24 de junho de 2022

Banca Examinadora:

Prof. Dr. Ricardo Matsumura de Araújo (orientador)
Doutor em Ciência da Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Frederico Schmitt Kremer
Doutor em Biotecnologia pela Universidade Federal de Pelotas.

Prof. Dr. Marilton Sanchotene de Aguiar
Doutor em Ciência da Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Ney Lemke
Doutor em Física pela Universidade Federal do Rio Grande do Sul.

AGRADECIMENTOS

Agradeço aos meus orientadores Prof. Dr. Ricardo Matsumura de Araújo e Dra. Marialva Sinigaglia (Meg), pesquisadores e pessoas incrivelmente maravilhosas, que tornaram possível meu sonho de utilizar a computação para investigar problemas biológicos em doenças. Aprendi tanto nesses últimos anos, graças a eles, que foi uma total mudança de percepção de mundo.

Agradeço, também, à minha família, minha mãe, pai e irmãos que me apoiaram nos momentos difíceis, sempre tão compreensíveis.

Ao grupo do Instituto do Câncer Infantil (ICI) de Porto Alegre que sempre estiveram dispostos a ajudar com explicações.

Aos professores do Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas (UFPel) pelos conhecimentos disponibilizados.

RESUMO

SIMAS, Gisele Moraes. **Aprendizagem de Máquina em Análise de Expressão Gênica -**
Classificação e Seleção de Genes Relevantes em Câncer. Orientador: Ricardo Matsumura de Araújo. 2022. 199 f. Tese (Doutorado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2022.

A seleção de biomarcadores gênicos pode beneficiar o gerenciamento clínico de pacientes e fornecer '*insights*' para a compreensão de doenças. No entanto, a análise de dados de expressão gênica ainda é um desafio, devido à: maldição de dimensionalidade (alta quantidade de *features* e pequeno número de amostras); presença de relações complexas não lineares; alto ruído de fundo; e dificuldade de análise de *datasets* em conjunto (com diferentes ruídos e escalas). Este trabalho visa analisar métodos de Aprendizagem de Máquina Profunda e Rasa para classificação de amostras e seleção de genes relevantes em *datasets* de *microarray* de diferentes tecidos. São explorados: as *Convolutional Neural Networks (CNNs)*; os *Vision Transformers*; a *Generative Adversarial Network (GAN)*; e a *Multilayer Perceptron (MLP)*. Propomos a adoção do método Transcriptograma para a análise de redes de interação gênica e obtenção de um ordenamento de genes que possa ser explorado pelas CNNs. Além disso, propomos uma abordagem, nomeada de MLPEns, que explora o *dropout* para tratar um modelo de Rede Neural como um *ensemble* de modelos, visando aproveitar a alta capacidade de generalização dos *ensembles*. Para a seleção de genes relevantes foram analisados: os coeficientes de *Support Vector Machines (SVMs)*; o Boruta Shap; e os pesos da MLPEns. Nossos resultados demonstram que, ao contrário da tendência recente na área, alguns métodos de Aprendizagem Rasa (como o SVM Linear L2 e a Regressão *Ridge*) fornecem resultados estatisticamente equivalentes à nossa melhor abordagem de Aprendizagem Profunda, tendo menor tempo de execução e simplicidade na interpretabilidade dos resultados; sendo, portanto, percebidos como as melhores opções para a análise desse tipo de dados. A abordagem de empregar o Transcriptograma possibilitou melhorar a acurácia da CNN; e a MLPEns alcançou resultados estatisticamente equivalentes às melhores abordagens analisadas.

Palavras-chave: *Deep Learning*. Redes Neurais. CNN. GAN. *Dropout*. Boruta Shap. Câncer. Biomarcadores. Expressão Gênica.

ABSTRACT

SIMAS, Gisele Moraes. **Machine Learning in Gene Expression Analysis - Classification and Selection of Relevant Genes in Cancer**. Advisor: Ricardo Matsumura de Araújo. 2022. 199 f. Thesis (Doctorate in Computer Science) – Technology Development Center, Federal University of Pelotas, Pelotas, 2022.

The selection of gene biomarkers can benefit the clinical management of patients and provide insights into the understanding of disease. However, the analysis of gene expression data is still a challenge, due to: curse of dimensionality (high number of features and small number of samples); presence of non-linear complex relationships; high background noise; and difficulty in analyzing datasets together (with different noise and scales). This work aims to analyze Deep and Shallow Machine Learning methods for sample classification and selection of relevant genes in microarray datasets from different tissues. We explore the following methods: Convolutional Neural Networks (CNNs); Vision Transformers; Generative Adversarial Network (GAN); and Multilayer Perceptron (MLP). We propose the adoption of the Transcriptogram method for analyzing gene interaction networks and obtaining an ordering of genes that can be explored by CNNs. Furthermore, we propose an approach, called MLPEns, which exploits dropout to treat a Neural Network model as an ensemble of models, aiming to take advantage of the high generalization capacity of the ensembles. For the selection of the relevant genes, we analyze: coefficients of Support Vector Machines (SVMs); Boruta Shap; and weights of MLPEs. Our results demonstrate that, contrary to the recent trend, some Shallow Learning methods (such as SVM Linear L2 and Ridge Regression) provide results that are statistically equivalent to our best Deep Learning approach and, in addition, have shorter execution times and greater simplicity in interpretability of results. Therefore, we perceive these methods as the best options for analyzing this type of data. The approach of employing the Transcriptogram has improved CNN accuracy; and MLPEns obtained results statistically equivalent to the best approaches analyzed.

Keywords: Deep Learning. Neural Networks. CNN. GAN. Dropout. Boruta Shap. Cancer. Biomarkers. Gene Expression.

LISTA DE FIGURAS

Figura 1 Esquema de *Convolutional Neural Networks - CNNs* para classificação de imagens. Fonte (PRABHU, 2018). 34
Figura 2 Bloco da ResNet (fonte (SIMONYAN; ZISSERMAN, 2014) - adaptada). 36
Figura 3 Esquema da *Generative Adversarial Network - GAN*. Fonte (GOODFELLOW, 2016). 37
Figura 4 Ordenamento - Transcriptograma - Configuração final de uma matriz de adjacências. Fonte (KUENTZER, 2014). 42
Figura 5 Gráfico montado pelo Transcriptograma. Fonte (MORAIS; ALMEIDA; DALMOLIN, 2019). 43

Figura 6 Aplicações de Deep Learning em Bioinformática. 46
Figura 7 Trabalhos selecionados para serem detalhados. 48
Figura 8 Técnicas de *Deep Learning* usadas nos trabalhos selecionados. . . 49
Figura 9 Dados de entrada dos métodos propostos pelos trabalhos selecionados. 51
Figura 10 Aplicações tratadas pelos trabalhos selecionados. 52

Figura 11 Esquema da abordagem empregada. 70
Figura 12 Esquema da abordagem empregada - Comparação de Métodos de Aprendizagem de Máquina 70
Figura 13 Datasets empregados. 71
Figura 14 Relação dos datasets e experimentos. 72
Figura 15 Processo adotado para obtenção do melhor conjunto de hiperparâmetros por método de rede neural. 75
Figura 16 Esquema das abordagens de seleção de genes relevantes adotadas. 77

Figura 17 Arquitetura da MLPEns proposta. 80

Figura 18 Curva de aprendizagem do SVM e CNN para comparação de tendência de caimento da curva e verificação da polarização (bias) versus variância. A curva mostra valores de erro de acurácia, conforme se aumenta o número de amostras usadas no conjunto de treinamento para um mesmo conjunto de teste. 90
Figura 19 Comparação dos Kenels do SVM (20 execuções e 185 problemas). 96
Figura 20 Comparação dos Kenels do SVM - proporção de problemas que tem a maior média de acurácia em cada tipo de kernel (20 execuções e 185 problemas). 97

Figura 21 Kernel Linear L2 x Linear L1 - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 97
Figura 22 Comparação de diferentes valores de max_depth no XGBoost (20 execuções e 31 problemas). 98
Figura 23 Comparação de diferentes formas de entrada para o SVM Linear L2 (20 execuções e 185 problemas). 99
Figura 24 Comparação de diferentes formas de entrada para o XGBoost com $max\_depth = 3$ (20 execuções e 31 problemas). 100
Figura 25 Comparação de Melhores Formas de Entrada no SVM - proporção de problemas que tem a maior média de acurácia em cada tipo de entrada (20 execuções e 185 problemas). 100
Figura 26 Comparação SVM x XGBoost - proporção de problemas que tem a maior média de acurácia em cada tipo de método (20 execuções e 31 problemas). 101
Figura 27 Comparação entre Opções Experimentadas - proporção de problemas que tem a maior média de acurácia em cada opção experimentada - XGBoost variando max_depth, SVM variando o kernel, 10 formas de entrada, com e sem *class_weigth*(cw) (20 execuções e 31 problemas). 102
Figura 28 Matriz de adjacências com as posições dos nodos (genes) inicializadas aleatoriamente. Pontos pretos representam a existência de uma aresta entre os nodos na posição x (coluna) com o na posição y (linha). 105
Figura 29 Matriz de adjacência obtida pelo Transcriptograma - os nodos (genes) com mais arestas entre si (funções relacionadas) estão posicionados de forma próxima, conforme pode ser percebido pela diagonal preenchida. 105
Figura 30 ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - 11 problemas. 106
Figura 31 ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 107
Figura 32 ResNet 18 2D x 1D - 11 problemas. 108
Figura 33 ResNet 18 2D x 1D - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 109
Figura 34 ResNet 18 x ResNet 34 - 11 problemas. 110
Figura 35 MLP com 3 camadas escondidas x 4 camadas escondidas - 11 problemas. 111
Figura 36 MLP com 3 camadas escondidas x 4 camadas escondidas - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 113
Figura 37 MLP x ResNet 18 - 11 problemas. 114
Figura 38 MLP x ResNet 18 - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 115
Figura 39 ResNet 18 x ResNet 18 Pré-treinada - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 116
Figura 40 ResNet 18 e redes neurais com Mecanismo de Atenção - 11 problemas. 117

Figura 41 ResNet 18 x ResNet 18 Pre-Activated Simple Self-Attention - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 120
Figura 42 ResNet 18 x Vision Transformer - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 121
Figura 43 ResNet 18 x Hybrid Vision Transformer - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 122
Figura 44 Hiperparâmetros da GAN-MLP - 11 problemas. 124
Figura 45 MLP x GAN-MLP - 11 problemas. 125
Figura 46 MLP x GAN-MLP - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 126
Figura 47 MLPEns x ResNet x MLP x SVM - 11 problemas. 128
Figura 48 Comparação da MLPEns com o SVM - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 129
Figura 49 Comparação da MLPEns com a ResNet - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 130
Figura 50 Comparação da MLPEns com a MLP - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia. 130
Figura 51 Métodos de Aprendizagem Rasa do PyCaret (11 problemas e 20 sementes aleatórias). 132

Figura 52 tSNE - Meduloblastoma - todos features (21.641 genes). 139
Figura 53 SVM pela Posição - Melhores Genes- Heatmap. 141
Figura 54 SVM pelo Coeficiente - Melhores Genes - Heatmap. 142
Figura 55 MLPEns pela Posição - Heatmap. 143
Figura 56 MLPEns pelo Coeficiente - Melhores Genes - Heatmap. 144
Figura 57 MLPEns pela Posição - Melhores Genes - Heatmap. 145
Figura 58 MLPEns pelo Coeficiente - Melhores Genes - Heatmap. 146
Figura 59 SVM - 10 genes - tSNE. 148
Figura 60 MLPEns - 10 genes - tSNE. 148
Figura 61 SVM - 30 genes - tSNE. 149
Figura 62 MLPEns - 30 genes - tSNE. 149
Figura 63 SVM - 100 genes - tSNE. 150
Figura 64 MLPEns - 100 genes - tSNE. 150
Figura 65 SVM - 300 genes - tSNE. 151
Figura 66 MLPEns - 300 genes - tSNE. 151
Figura 67 tSNE - Sarcoma de Ewing e outros tumores - todos features (4.924 genes). 152
Figura 68 SVM pela Posição - Melhores Genes- Heatmap. 153
Figura 69 SVM pelo Coeficiente - Melhores Genes - Heatmap. 154
Figura 70 MLPEns pela Posição - Heatmap. 155
Figura 71 MLPEns pelo Coeficiente - Melhores Genes - Heatmap. 156
Figura 72 SVM - 5 genes - tSNE 159
Figura 73 MLPEns - 5 genes - tSNE 159
Figura 74 SVM - 100 genes - tSNE 160
Figura 75 MLPEns - 100 genes - tSNE 160
Figura 76 Diagrama de Venn dos genes selecionados pelos métodos Extra Trees (ET), Random Forest (RF) e XGBoost. 161

Figura 77 Meduloblastoma - tSNE com os 58 genes selecionados pelo BorutaShap. 163
Figura 78 Meduloblastoma - Heatmap com os 58 genes selecionados pelo BorutaShap. 163

LISTA DE TABELAS

Tabela 1 11 Problemas Selecionados. 73

Tabela 2 Ensemble de SVM x Ensemble de CNN - com diferentes regras de combinação e número de modelos em cada ensemble. 92

Tabela 3 ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - 11 problemas. 106

Tabela 4 ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - média de acurácia por método - cada problema mostrado separadamente. 106

Tabela 5 ResNet 18 2D x 1D - 11 problemas. 108

Tabela 6 ResNet 18 2D x 1D - média de acurácia por método - cada problema mostrado separadamente. 108

Tabela 7 ResNet 18 x ResNet 34 - 11 problemas. 110

Tabela 8 ResNet18 e ResNet34 - média de acurácia por método - cada problema mostrado separadamente. 110

Tabela 9 MLP com 3 camadas escondidas x 4 camadas escondidas - 11 problemas. 112

Tabela 10 MLP com 3 camadas escondidas x 4 camadas escondidas - média de acurácia - cada problema mostrado separadamente. 112

Tabela 11 MLP x ResNet 18 - 11 problemas. 114

Tabela 12 MLP x ResNet 18 - média de acurácia - cada problema mostrado separadamente. 114

Tabela 13 ResNet e Redes Neurais com Mecanismo de Atenção - 11 problemas. 118

Tabela 14 ResNet e Redes Neurais com Mecanismo de Atenção - média de acurácia por método - cada problema mostrado separadamente. . . 118

Tabela 15 Hiperparâmetros da GAN-MLP - 11 problemas. 124

Tabela 16 Hiperparâmetros da GAN-MLP - média de acurácia - cada problema mostrado separadamente. 125

Tabela 17 MLP x GAN-MLP - 11 problemas. 126

Tabela 18 MLP x GAN-MLP - média de acurácia - cada problema mostrado separadamente. 126

Tabela 19 MLPEns x ResNet x MLP x SVM - 11 problemas. 128

Tabela 20 MLPEns x ResNet x MLP x SVM - média de acurácia por método - cada problema mostrado separadamente. 129

Tabela 21 Resultados da classificação com 18 métodos de Aprendizagem Rasa (dataset piloto e 1 semente aleatória). 131

Tabela 22 Métodos de Aprendizagem Rasa do PyCaret (11 problemas e 20 sementes aleatórias). 132
Tabela 23 Métodos de Aprendizagem Rasa do PyCaret - média de acurácia - cada problema mostrado separadamente (20 sementes aleatórias). 133
Tabela 24 Melhores Métodos (parte 1/3) - média de acurácia - cada problema mostrado separadamente. 134
Tabela 25 Melhores Métodos (parte 2/3) - média de acurácia - cada problema mostrado separadamente. 134
Tabela 26 Melhores Métodos (parte 3/3) - média de acurácia - cada problema mostrado separadamente. 134
Tabela 27 Tempo de Execução dos Principais Métodos - uma semente aleatória e um conjunto de hiperparâmetros - problema de classificação de subtipos moleculares de meduloblastoma (8,2) - GSE85217 (763 amostras) - executados em uma conta pro do Google Colab com GPU P100. 135

Tabela 28 SVM - Interseção dos genes selecionados i) pelo critério de posição; e ii) pelos coeficientes. 137
Tabela 29 MLPEns - Interseção dos genes selecionados i) pelo critério de posição; e ii) pelos coeficientes. 137
Tabela 30 SVM e MLPEns - Posição - Interseção dos conjuntos de genes selecionados pelo critério de posição pelo i) SVM e ii) MLPEns. 138
Tabela 31 SVM e MLPEns - Coeficiente - Interseção dos conjuntos de genes selecionados pelos coeficientes pelo i) SVM e ii) MLPEns 138
Tabela 32 SVM pela Posição - Melhores Genes. 140
Tabela 33 SVM pelo Coeficiente - Melhores Genes. 141
Tabela 34 MLPEns pela Posição - Melhores Genes. 143
Tabela 35 MLPEns pelo Coeficiente - Melhores Genes. 144
Tabela 36 Separação da classe Group 3 - SVM pela Posição - Melhores Genes. 147
Tabela 37 Separação da classe Group 4 - SVM pela Posição - Melhores Genes. 147
Tabela 38 Separação da classe SHH - SVM pela Posição - Melhores Genes. . 147
Tabela 39 Separação da classe WNT - SVM pela Posição - Melhores Genes. . 147
Tabela 40 SVM pela Posição - Melhores Genes. 153
Tabela 41 SVM pelo Coeficiente - Melhores Genes. 154
Tabela 42 MLPEns pela Posição - Melhores Genes. 155
Tabela 43 MLPEns pelo Coeficiente - Melhores Genes. 156
Tabela 44 Sarcoma de Ewing - Primeiros 7 genes selecionados pelos diferentes critérios com SVM e MLPEns. 158
Tabela 45 Resultado de classificadores usando, como *features*, os genes selecionados pelo BorutaShap. 162

Tabela 46 Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 1/5. 186
Tabela 47 Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 2/5. 187
Tabela 48 Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 3/5. 188
Tabela 49 Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 4/5. 189

Tabela 50 Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 5/5. 190

Tabela 51 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 1/9. 191
Tabela 52 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 2/9. 192
Tabela 53 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 3/9. 193
Tabela 54 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 4/9. 194
Tabela 55 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 5/9. 195
Tabela 56 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 6/9. 196
Tabela 57 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 7/9. 197
Tabela 58 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 8/9. 198
Tabela 59 Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 9/9. 199

LISTA DE ABREVIATURAS E SIGLAS

BCDForest	Boosting Cascate Deep Forest
BioGRID	Biological General Repository for Interaction Datasets
CASP	Critical Assessment of protein Structure Prediction
CDC	Centers for Disease Control and Prevention
CNA	Copy Number Alteration
CNN	Convolutional Neural Network
CuMiDa	Curated Microarray Database
DAE	Denoising Autoencoders
DBN	Deep Belief Network
DFS	Deep Feature Selection
DGS	Deep Genomic Signature
DL	Deep Learning
DNA	Deoxyribonucleic Acid
DNN	Deep Neural Network
eQTL	Expression Quantitative Trait Locus
GAN	Generative Adversarial Network
GeneCT	Generalizable Cancerous-status and Tissue-of-origin classifier
GEO	Gene Expression Omnibus
GTEx	Genotype-Tissue Expression project
ICtextsubscript50	Half maximal Inhibitory Concentration
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
INCA	Instituto Nacional de Câncer
InfoGAN	Information Maximizing Generative Adversarial Network
KL	Kullback-Leibler
k-NN	k-Nearest Neighbors
LRP	Layer-wise Relevance Propagation

LSTM	Long Short Term Memory
MGC	Molecular Graph Convolution
MI	Mutual Information
MINE	Mutual Information Neural Estimator
NLP	Natural Language Processing
microRNA	Micro Ribonucleic Acid
miRNA	Micro Ribonucleic Acid (o mesmo que microRNA)
miRNA-Seq	Micro Ribonucleic Acid Sequencing
ML	Machine Learning
MLFS	Multi-Level Feature Selection
MLP	Multilayer Perceptron
MLPEns	MLP Ensemble
NGS	Next-Generation Sequencing
NN	Neural Network
ORF	Open Reading Frame
PCA	Principal Component Analysis
PFI	Permutation Feature Importance
PPI	Protein-Protein Interaction
RBM	Restricted Boltzmann Machine
RNA	Ribonucleic Acid
RNA-Seq	RNA Sequencing
RNN	Recurrent Neural Network
SAE	Stacked Autoencoders
scRNA-seq	Single-Cell RNA Sequencing
SDAE	Stacked Denoising Autoencoder
SNP	Single Nucleotide Polymorphisms
SVM	Support Vector Machine
TARGET	Therapeutically Applicable Research to Generate Effective Treatments
TCGA	The Cancer Genome Atlas
TF	Transcriptional Factor
UFPel	Universidade Federal de Pelotas
VAE	Variational Autoencoder

SUMÁRIO

1 INTRODUÇÃO 20
1.1 Bioinformática e Deep Learning 21
1.2 Análise de Expressão Gênica 24
1.2.1 Análise Integrada de Plataformas 25
1.2.2 Maldição de Dimensionalidade 26
1.2.3 Engenharia de Features 27
1.2.4 Medicina Personalizada e Desenvolvimento de Novos Fármacos 27
1.3 Objetivo Geral 28
1.4 Objetivos Específicos 29
1.5 Organização do Trabalho 29

2 REFERENCIAL TEÓRICO 31
2.1 Aprendizagem de Máquina 31
2.2 Deep Neural Networks 32
2.3 Arquiteturas de DNNs 33
2.4 Rede Neural Convolucional - CNN 34
2.4.1 ResNet 35
2.5 Generative Adversarial Network - GAN 37
2.5.1 MetaGAN 38
2.6 Transfer Learning 39
2.6.1 Bases de Dados ImageNet 39
2.7 Regularização - Dropout 40
2.8 Transcriptograma 40
2.8.1 Ordenamento 40
2.8.2 Cálculo de Médias 42
2.8.3 Comparação entre Classes 42
2.8.4 Utilização do Transcriptograma para Estabelecer a Entrada de CNNs . . 43

3 TRABALHOS RELACIONADOS 44
3.1 Aplicações 45
3.2 Métodos Relacionados - Expressão Gênica 47
3.3 Considerações sobre os Trabalhos Relacionados 66

4 METODOLOGIA 69
4.1 Esquema Geral 69
4.2 Datasets Empregados 71
4.2.1 Amostras de Expressão Gênica em Cânceres 72
4.2.2 Auxiliares 73

4.3 Propostas Originais . . . 74
4.4 Definição dos Melhores Conjuntos de Hiperparâmetros . . . 74
4.4.1 Métodos de Redes Neurais . . . 74
4.4.2 Métodos de Aprendizagem Rasa . . . 75
4.5 Comparação dos Métodos de Aprendizagem de Máquina para Classificação . . . 76
4.6 Sistema de Execução dos Métodos . . . 76
4.7 Interpretabilidade - Seleção de Genes Relevantes . . . 77
4.8 Comparação dos Métodos de Aprendizagem de Máquina para Seleção de Genes Relevantes . . . 78

5 MLPENS . . . 79
5.1 Rede Neural Vista como um Ensemble de Caminhos . . . 79
5.2 Arquitetura com Ensemble de Caminhos . . . 80
5.2.1 Loss para a Arquitetura MLPEns . . . 81
5.3 Predição - Tratando a Rede Neural como um *Ensemble* . . . 81
5.4 Determinando o Melhor Conjunto de Pesos . . . 81
5.5 Interpretabilidade da Rede Neural - *Feature Selection* . . . 82

6 SELEÇÃO DE GENES RELEVANTES . . . 83
6.1 Seleção de *Features* . . . 83
6.2 Seleção de Genes Relevantes . . . 84
6.2.1 Seleção de Biomarcadores para Diagnóstico . . . 84
6.2.2 Seleção de Genes Relevantes para a Compreensão de um Fenômeno / Doença . . . 85
6.3 SVM e MLPEns . . . 85
6.3.1 Combinando Valores de Importância para Vários Modelos . . . 85
6.4 BorutaShap . . . 85
6.5 Visualização - t-SNE . . . 86

7 RESULTADOS E DISCUSSÕES - CLASSIFICAÇÃO . . . 87
7.1 Experimentos Iniciais - CNN x SVM . . . 87
7.1.1 Resultados e Curva de Aprendizagem . . . 90
7.2 Ensemble de CNN x Ensemble de SVM . . . 91
7.3 Próximos Experimentos . . . 93
7.3.1 Formas de Entradas . . . 93
7.4 SVM e XGBoost - Diferentes Formas de Entrada . . . 95
7.4.1 Hiperparâmetros . . . 95
7.4.2 Demais Especificações . . . 95
7.4.3 Resultados - Kernel - SVM . . . 96
7.4.4 Resultados - max_depth - XGBoost . . . 98
7.4.5 Resultados - Ponderação de Classes - SVM e XGBoost . . . 98
7.4.6 Resultados - Diferentes Formas de Entrada - SVM e XGBoost . . . 99
7.4.7 Resultados - SVM x XGBoost . . . 101
7.5 ResNet e Transcriptograma . . . 103
7.5.1 Datasets / Problemas . . . 103
7.5.2 Definição dos hiperparâmetros . . . 104
7.5.3 Transcriptograma . . . 104
7.5.4 Ordem Aleatória x Ordem Fornecida pelo Transcriptograma . . . 105

7.5.5 ResNet 1D x ResNet 2D . . . 107
7.5.6 ResNet 18 x ResNet 34 . . . 109
7.6 MLP . . . 111
7.7 ResNet 18 Pré-treinada . . . 115
7.8 Redes Neurais Profundas com Mecanismo de Atenção . . . 116
7.8.1 ResNet 18 Pre-Activated Simple Self-Attention . . . 118
7.8.2 Vision Transformer . . . 120
7.8.3 Hybrid Vision Transformer - Features Obtidos com a ResNet . . . 122
7.9 GAN-MLP . . . 123
7.10 MLPEns . . . 127
7.11 Métodos de Aprendizagem Rasa . . . 131
7.12 Comparação entre os Principais Métodos de Classificação . . . 133

8 RESULTADOS E DISCUSSÕES - SELEÇÃO DE GENES . . . 136
8.1 Seleção de Genes Relevantes . . . 136
8.2 SVM Linear L2 e MLPEns . . . 136
8.2.1 Meduloblastoma - (8,2) - GSE85217 . . . 138
8.2.2 Sarcoma de Ewing - (1,4) - GSE2553 . . . 152
8.3 BorutaShap . . . 161
8.4 Comparação entre as Abordagens para Seleção de Genes Relevantes 164

9 CONCLUSÃO . . . 165
9.1 Contribuições do Trabalho . . . 167
9.2 Trabalhos Futuros . . . 168
9.3 Resumo do Texto . . . 169

REFERÊNCIAS . . . 170

APÊNDICE A TRABALHOS RELACIONADOS - APLICAÇÃO DE DEEP LEARNING EM BIOINFORMÁTICA . . . 186

APÊNDICE B PROBLEMAS E DATASETS DE EXPRESSÃO GÊNICA . . . 191

1 INTRODUÇÃO

Câncer é a segunda principal causa de morte em todo o mundo e foi responsável por cerca de 9.6 milhões de mortes em 2018. Durante o ano de 2018, foram diagnosticados 18.1 milhões de novos casos de câncer[1]. Estimativas indicam que um em cada cinco homens e uma em cada seis mulheres desenvolverá câncer durante sua vida (CANCER, 2018). Globalmente, uma em cada seis mortes é devido a cânceres, segundo a Organização Mundial de Saúde (WHO, 2019).

O Instituto Nacional de Câncer (INCA) informa que, durante o ano de 2018, foram registrados 582.590 novos casos de cânceres no Brasil (INCA, 2019). Em 10% das cidades brasileiras, mortes por câncer superaram as causadas por doenças cardiovasculares, sendo maiores que as causadas por qualquer outro aspecto, seja em relação a doenças ou até mesmo acidentes de trânsito e homicídios[2] (JORNAL, 2018).

Dados do CDC (*Centers for Disease Control and Prevention*) (HERON, 2018), sobre mortes nos Estados Unidos, durante o ano de 2016 apontam que câncer é a causa líder de mortes nesse país para a população de 45-64 anos (com 29.2% de todas as mortes). Na população de 65 anos ou mais, é a segunda causa lider (com 21.1% das mortes), perdendo apenas para doenças cardíacas (com 25.3%). Sem separação por idade, câncer assume 21.8% das mortes, perdendo apenas para mortes causadas por doenças cardíacas (com 23.1%).

Segundo estimativas da Organização Mundial de Saúde (OMS), em 2015, o câncer foi a primeira ou segunda causa de morte antes dos 70 anos, em 91 de 172 países analisados e ocupa o terceiro ou quarto lugar em outros 22 países (BRAY et al., 2018). A incidência e a mortalidade por câncer estão crescendo rapidamente em todo o mundo. As razões são complexas, mas refletem tanto envelhecimento da população, bem como mudanças de prevalência e distribuição dos principais fatores de risco para o câncer (BRAY et al., 2018).

[1] Conforme dados do projeto GLOBOCAN que provê estimativas de incidência e mortalidade de 185 países e 36 tipos de cânceres (CANCER, 2019).

[2] Com base em números oficiais mais recentes do Sistema de Informações de Mortalidade (SIM) de 2015 - análise divulgada pelo Observatório de Oncologia do movimento Todos Juntos Contra o Câncer, em parceria com o Conselho Federal de Medicina (CFM).

Portanto, atividades de pesquisa na área de análise de informações sobre cânceres são de suma importância. **Câncer**[3] é um termo genérico para um grande grupo de doenças (mais de 100 tipos de cânceres conhecidos (LIU et al., 2018)) que podem afetar qualquer parte do corpo. Todas essas doenças têm uma característica em comum: crescimento desordenado (maligno) de células anormais que podem invadir partes adjacentes do corpo e se espalhar para outros órgãos, o último processo é chamado de **metástase**. As metástases são uma das principais causas de morte por câncer (WHO, 2019).

O câncer é associado a múltiplas aberrações genéticas e regulatórias na célula, que refletem em diferenças em níveis de **expressão gênica** (FAKOOR et al., 2013). Todas as células de determinada pessoa apresentam o mesmo DNA (desconsiderando pequenas mutações ao longo da vida). Entretanto, os diferentes órgãos exercem funções distintas. Isso se dá através da expressão gênica diferenciada entre células. Doenças e distúrbios fisiológicos podem modificar a expressão de células saudáveis. Quando ocorrem alterações no material genético, como nos cânceres, o padrão de expressão gênico das células também é modificado. Assim, a análise de expressão gênica é fundamental para se obter novos '*insights*' sobre os diferentes tipos de cânceres.

Recentemente, com o avanço das tecnologias de sequenciamento de larga escala (*Next-Generation Sequencing* - NGS), tem-se produzido uma enorme quantidade de dados, que fornecem oportunidades sem precedentes para investigar mudanças genômicas ou transcriptômicas associadas a cânceres, de forma a ser possível pensar em análises em níveis moleculares (GUO et al., 2018).

1.1 Bioinformática e Deep Learning

Com essa quantidade sem precedentes de dados biológicos disponíveis, há uma necessidade urgente de desenvolvimento de novos métodos computacionais que explorem esses dados, de forma a contribuir para novas descobertas científicas (XIE et al., 2016; SUN; WANG; LI, 2018).

Neste escopo, uma área de pesquisa importante é a **Bioinformática**. Esse é o termo dado ao campo interdisciplinar (envolve áreas como biologia molecular, genética, ciência da computação, matemática, física e estatística) que visa investigar e compreender processos biológicos em nível molecular (RAVÌ et al., 2017). Algumas áreas de estudo da bioinformática são: análises filogenéticas; análise de sequências; bioinformática estrutural; análise de dados de expressão gênica; biologia de sistemas; sequenciamento genomas; genômica comparativa; planejamento racional de fármacos; medicina personalizada.

[3] Outros termos utilizados para designar cânceres são tumores malignos e neoplasias (WHO, 2019).

No escopo de desenvolvimento de novas abordagens computacionais capazes de tratar essa enorme quantidade de dados, a **Aprendizagem de Máquina - *Machine Learning (ML)*** se torna indispensável. A ML visa desenvolver algoritmos computacionais que melhoram (aprendam) com a experiência, de forma a permitir que os computadores ajudem os seres humanos na análise de conjuntos de dados complexos (LIBBRECHT; NOBLE, 2015; LIN; LANE, 2017).

Nos últimos anos um tipo de Aprendizagem de Máquina, a **Aprendizagem Profunda - *Deep Learning (DL)***, tem predominado o campo de ML, obtendo resultados no estado-da-arte, em muitas áreas de pesquisas, indo desde a classificação de imagens, reconhecimento de fala, até a Bioinformática (LECUN; BENGIO; HINTON, 2015; MAMOSHINA et al., 2016; ZHANG et al., 2018; CHING et al., 2018; SUN; WANG; LI, 2018; ZHANG et al., 2018).

O termo *Deep Learning* refere-se a métodos que realizam a aprendizagem hierarquicamente, através de múltiplas camadas de abstração (ZHANG et al., 2018). Dessa forma, a DL consegue construir representações de alto nível de abstração (SHARIFI-NOGHABI et al., 2018).

Em contraste com outros métodos de Aprendizagem de Máquina, frequentemente empregados, que precisam de engenharia de *features* (elaborados manualmente); a *Deep Learning* pode extrair *features*, automaticamente, de dados brutos e aprender abstrações complexas, diretamente, combinando *features* mais simples, hierarquicamente, através de arquiteturas profundas com múltiplas camadas (MIN; LEE; YOON, 2016).

Destaca-se, também que a DL possui capacidade de aprendizagem de dados complexos, com estruturas não lineares (MAMOSHINA et al., 2016), sem a necessidade impor suposições sobre a distribuição dos dados analisados (CUI et al., 2018) ou requerer conhecimento a priori (CHIU et al., 2018).

Um dos tipos de métodos de *Deep Learning* mais usados são as **Redes Neurais Profundas - Deep Neural Network (DNN)**. Uma DNN é uma rede neural artificial composta de múltiplas camadas computacionais que processam dados de forma hierárquica.

A DL apresenta um conjunto de diferenciais em relação aos métodos tradicionais de ML que a fazem despontar, no campo de Bioinformática, como uma abordagem promissora (CHING et al., 2018; MAMOSHINA et al., 2016; UNTERTHINER et al., 2014).

A DL possibilita que a aprendizagem ocorra em níveis hierárquicos, de forma a tornar possível a aprendizagem de estrutura de dados complexos, indo da aprendizagem de padrões mais simples até padrões altamente abstratos (LECUN; BENGIO; HINTON, 2015).

Os métodos de DL aprendem automaticamente atributos (*featu-*

res) dos dados de entrada; sem requerer engenharia e processamento extensivo dos dados com o emprego considerável de conhecimento de especialistas na área de aplicação dos métodos (KALININ et al., 2018; JING et al., 2018). Essa característica é muito importante para tratar alguns problemas que a comunidade científica ainda não conseguiu elucidar completamente (por exemplo, a predição conformacional de proteínas, a partir da sequência de aminoácidos (GAO; YANG; ZHOU, 2018; RENHULDT, 2018)).

Os métodos de DL apresentam maior capacidade de generalização (existem técnicas que evitam o *overfitting* - especialização dos métodos) (SRIVASTAVA et al., 2014).

A DL permite analisar conjuntos de *Big-data* de alta dimensionalidade e integrar dados de diferentes tipos, favorecendo análises de Biologia de Sistemas (MA et al., 2018).

A DL também pode trazer benefícios no desenvolvimento de novas abordagens de aprendizagem não supervisionadas (por exemplo, através do emprego de Redes Neurais Autocodificadoras). Ressalta-se que, na área de bioinformática, têm-se uma quantidade incremental enorme de dados sendo gerados (ANGERMUELLER et al., 2016) e muitas vezes o aprendizado supervisionado não é suficiente para tratá-los, pela falta de conhecimento (rótulos) agregado aos dados (SHARIFI-NOGHABI et al., 2018).

Existem, também, abordagens de DL para aprendizagem por reforço, que podem ser empregadas em áreas na qual não existem grandes quantidades de dados disponíveis (como, por exemplo, em planejamento de fármacos (SERRANO et al., 2018)).

Novas abordagens em DL, como mecanismos de atenção (GAO et al., 2018), têm possibilitado maior transparência e interpretabilidade sobre as escolhas tomadas pelos métodos de DL.

Nos últimos três anos, foi publicada uma expressiva quantidade de trabalhos que envolvem a aplicação de DL em problemas biológicos, vários desses trabalhos superam ou alcançam o estado-da-arte em muitas áreas (KAMARUDIN et al., 2018; GAO et al., 2018; SALEKIN, 2018; BAEK et al., 2018; YANG et al., 2018; PAN et al., 2018; SUNSERI et al., 2018; DU et al., 2018; GAO et al., 2018; ÖZTÜRK; OZKIRIMLI; ÖZGÜR, 2018; GAO; YANG; ZHOU, 2018; WANG; HOEKSEMA; LIANG, 2018; DIZAJI; WANG; HUANG, 2018; RENHULDT, 2018).

Recentemente, foram publicadas revisões focadas em demonstrar possíveis aplicações da DL em Bioinformática (MIN; LEE; YOON, 2016; ANGERMUELLER et al., 2016; RAMPASEK; GOLDENBERG, 2016; MAMOSHINA et al., 2016; CELESTI et al., 2018; CAMACHO et al., 2018; BALDI, 2018; BACCIU et al., 2018; CHING et al., 2018; YUE; WANG, 2018; LAN et al., 2018; WANG et al., 2018); e na área de desenvolvimento de novos fármacos e triagem virtual (*virtual screening*) (UNTERTHINER et al., 2014; PÉREZ-SIANES; PÉREZ-SÁNCHEZ; DÍAZ, 2016;

GAWEHN; HISS; SCHNEIDER, 2016; EKINS, 2016; CHEN et al., 2018; PANTELEEV; GAO; JIA, 2018; KALININ et al., 2018; JING et al., 2018; GHASEMI et al., 2018).

Entretanto, segundo (BACCIU et al., 2018), o impacto da DL em Bioinformática ainda é limitado; devido, possivelmente, a alguns desafios que muitos *datasets* biológicos apresentam, como dados insuficientes e não balanceados e falta de formas de interpretação direta dos modelos aprendidos pela DL.

Também, foram publicadas recentes revisões sobre Machine Learning (em geral) na área biológica e médica (LIBBRECHT; NOBLE, 2015; LEUNG et al., 2016; LIN; LANE, 2017).

1.2 Análise de Expressão Gênica

Análises de **transcriptômica** exploram a variação de vários tipos de transcritos (RNA mensageiro (RNAm), RNA longo não codificante (RNAnc), microRNA (miRNA), etc.) para coletar uma variedade de informações funcionais (MAMOSHINA et al., 2016). Uma quantidade massiva desses dados está disponível em vários *datasets* (como o *Gene Expression Omnibus* - GEO (GEO, 2019) e o ArrayExpress (ARRAYEXPRESS, 2019)).

Os dados de **expressão gênica** medem o nível de atividade de genes dentro de um determinado tecido e, portanto, fornecem informações sobre atividades complexas dentro das respectivas células. Esses dados são, geralmente, obtidos medindo a quantidade de ácido ribonucleico mensageiro (RNAm), produzido durante a transcrição, que, por sua vez, é uma medida do quão ativo cada gene está (FAKOOR et al., 2013).

O perfil de expressão gênica é uma ferramenta poderosa para medir o padrão de expressão de dezenas de milhares de genes, em uma determinada célula ou tecido (DIZAJI; WANG; HUANG, 2018). Entre os diversos métodos desenvolvidos para a **predição do câncer**, a utilização dos níveis de expressão gênica é um dos principais assuntos de pesquisa neste campo (XIAO et al., 2018).

Estudos de análise de expressão gênica são, frequentemente, usados para determinar os **genes implicados em um distúrbio** (LIU et al., 2018; DANAEE; GHAEINI; HENDRIX, 2016; SHARIFI-NOGHABI et al., 2018; IBRAHIM et al., 2014; CHAUDHARY et al., 2017; LYU; HAQUE, 2018). Por exemplo, pode-se comparar dados de células cancerosas e saudáveis, para determinar os transcritos que são sobre-expressos e sub-expressos, no tipo de câncer em questão (LIU et al., 2018; DANAEE; GHAEINI; HENDRIX, 2016).

A partir dessa informação, podem-se priorizar estudos de desenvolvimento de **fármacos**, voltados aos genes de expressão diferencial, investigando, por exemplo, es-

ses genes como alvos para fármacos (*target genes*). Também, pode-se usar a informação de expressão gênica para inferir vias regulatórias nas células; melhorar o diagnóstico; personalizar tratamentos (**medicina personalizada**).

Dados de expressão gênica, também, podem ser estudados para a seleção de **biomarcadores**, capazes de determinar: em estágios iniciais, a presença de **câncer** (LIU et al., 2018; DANAEE; GHAEINI; HENDRIX, 2016) e **metástase** (SHARIFI-NOGHABI et al., 2018; IBRAHIM et al., 2014); **prognóstico** (CHAUDHARY et al., 2017); **respostas a fármacos** (CHIU et al., 2018; DINCER et al., 2018); tipo (LYU; HAQUE, 2018) e **subtipos de cânceres** (IBRAHIM et al., 2014).

Essas análises são especialmente úteis em doenças heterogêneas como os cânceres (CHIU et al., 2018), mas também podem ser aplicadas em outras doenças.

A análise dos dados de expressão gênica tem o potencial de levar a descobertas biológicas significativas (DANAEE; GHAEINI; HENDRIX, 2016). Entretanto, a análise de tais dados ainda são um **desafio** (DANAEE; GHAEINI; HENDRIX, 2016), devido à: alta dimensionalidade dos dados (FAKOOR et al., 2013); pouca quantidade de amostras em cada *dataset* (GUO et al., 2018); complexidade inerente aos dados (com relações não lineares) (DUTIL et al., 2018); e os ruídos de fundo (GUO et al., 2018) (derivados dos próprios métodos de medição do sinal).

1.2.1 Análise Integrada de Plataformas

Os dados de expressão gênica podem ser obtidos por diferentes tipos de **plataformas**, que se diferenciam pelo conjunto de genes analisados e pelo método de detecção de sinal. A expressão de genes é determinada medindo os níveis de RNAm, o quê pode ser feito por múltiplas técnicas: *microarrays*; *single-cell RNA sequencing (scRNA-seq)*; *expressed cDNA sequence tag (EST) sequencing*; *serial analysis of gene expression (SAGE) tag sequencing*; *massively parallel signature sequencing (MPSS)*; *RNA Sequencing (RNA-Seq)*; *in-situ hybridization* (MAMOSHINA et al., 2016). Todas essas técnicas são extremamente propensas a **ruídos de fundo** (MAMOSHINA et al., 2016; GUO et al., 2018).

Portanto, uma importante área de pesquisa, em Bioinformática, envolve o desenvolvimento de ferramentas estatísticas para separar o sinal de expressão gênica dos ruídos de fundo (SMYTH; YANG; SPEED, 2003). A **normalização** é necessária mesmo para análise de dados de uma única plataforma. A análise entre plataformas requer, mais ainda, técnicas de normalização - essas consistem em um desafio importante a ser considerado (MAMOSHINA et al., 2016; JÄRVINEN et al., 2004; HUA et al., 2008).

(JÄRVINEN et al., 2004) investigaram se perfis de expressão gênica, obtidos por diferentes plataformas de *microarray*, poderiam ser comparados. Seus resultados in-

dicaram que a combinação de dados de diferentes plataformas de *microarray* não é direta e destacaram que a variabilidade dos dados representa um desafio, para o desenvolvimento de aplicações de diagnóstico, a partir de dados de expressão, obtidos por *microarray*.

(MAMOSHINA et al., 2016) realizaram uma revisão sobre o emprego de *Deep Learning* (DL) em biomedicina, prospectando possíveis aplicações para a DL. (MAMOSHINA et al., 2016) defenderam uma posição otimista de que as *Deep Neural Networks* (DNNs) poderiam vir a propiciar uma **análise integrada de plataformas** de expressão gênica, devido à sua alta capacidade de generalização (evitando a necessidade de maiores cuidados em relação à normalização). (MAMOSHINA et al., 2016) argumentaram ainda que as DNNs, também, estão bem equipadas para lidar com alguns dos outros problemas importantes em transcriptômica, como alta dimensionalidade dos dados.

1.2.2 Maldição de Dimensionalidade

Embora se tenha uma quantidade expressiva de dados de expressão gênica sendo gerados e depositados em repositórios públicos; o número de amostras em cada *dataset* em separado (geradas por uma mesma plataforma e laboratório - com mesmas condições experimentais) é pequeno (FAKOOR et al., 2013). A obtenção de grandes *datasets* de amostras de expressão gênica ainda é um processo caro e difícil de se obter (DIZAJI; WANG; HUANG, 2018).

Para evitar os problemas, mencionados na Seção anterior 1.2.1, de comparação de amostras com diferentes ruídos de fundo e escalas; atualmente, costuma-se realizar as análises de expressão gênica, em separado, para cada tipo de plataforma de obtenção de dados e, até mesmo, algumas vezes, em separado por cada laboratório (JÄRVINEN et al., 2004), de forma a isolar efeitos causados pelas condições experimentais e métodos usados na medição do sinal.

Dessa forma, tem-se um grande desafio - tratado na literatura como a **maldição de dimensionalidade** (*dimensionality curse*) - em que a dimensionalidade dos dados (dezenas de milhares de genes) é muito alta em relação ao número de amostras dos *datasets* (algumas centenas de amostras ou menos). Esse problema é ainda mais grave em análises de cânceres raros e comparações de amostras de tecido normal versus tecido canceroso.

A maldição de dimensionalidade é um dos maiores desafios em análises de expressão gênica (SUN; WANG; LI, 2018; DUTIL et al., 2018; DINCER et al., 2018; IBRAHIM et al., 2014; GUO et al., 2018; DANAEE; GHAEINI; HENDRIX, 2016; BHAT; VISWANATH; LI, 2016; FAKOOR et al., 2013). Pequenos conjuntos de amostras aumentam o risco de especialização (*overfitting*) na aprendizagem dos métodos (FA-

KOOR et al., 2013).

Esse problema, também, constitui um aspecto pelo qual pesquisadores defendem a determinação de assinaturas gênicas (biomarcadores) (IBRAHIM et al., 2014), de forma a selecionar apenas um grupo de genes mais discriminativos, a serem usados em classificadores, desconsiderando os demais genes, que podem funcionar como fonte de ruído.

1.2.3 Engenharia de Features

A maioria dos métodos de análises de expressão gênica empregam engenharia de *features* (seleção e construção de *features* manualmente ou de forma supervisionada) (FAKOOR et al., 2013). Por causa disso, muitos métodos computacionais, para análise de tais dados, são não escaláveis e não podem ser generalizados para, por exemplo, outros tipos de cânceres, além daqueles usados no conjunto do treinamento, sem reprojeto de *features* (FAKOOR et al., 2013).

(ZHANG et al., 2018), também, destaca esse aspecto, comentando que classificadores baseados em assinaturas gênicas ainda são instáveis e independentes do estudo, devido: i) à heterogeneidade dos dados de expressão gênica; e ii) ao fato serem muito dependentes de engenharia de *features* (projetados manualmente).

1.2.4 Medicina Personalizada e Desenvolvimento de Novos Fármacos

Informações de **assinaturas moleculares** permitem refinar as características gerais de uma determinada doença, permitindo destacar variações sutis moleculares capazes de explicar as diferenças observadas entre cada paciente (LOPEZ-RINCON et al., 2018). A integração dessas variações individuais, na prática clínica e no gerenciamento terapêutico, é uma nova estratégia chamada **medicina personalizada** ou medicina de precisão (LOPEZ-RINCON et al., 2018).

A análise de dados de expressão gênica, também, exerce papel fundamental em estudos de **descoberta de novos fármacos**. Dados de expressão gênica têm sido aplicados na construção de redes de interação fármacos-alvos (*drug-target*) e descoberta de fármacos (DIZAJI; WANG; HUANG, 2018). Nessas tarefas, a caracterização de diferentes padrões de expressão gênica, em resposta a perturbações de pequenas moléculas distintas, facilita a análise dos mecanismos e efeitos de fármacos (DIZAJI; WANG; HUANG, 2018). Neste escopo, emerge o campo de pesquisa de **farmacogenômica**, que estuda como as características genômicas e transcriptômicas determinam a resposta a fármacos (CHIU et al., 2018).

A identificação de pequenos conjuntos de *features* (**biomarcadores**) que efetivamente caracterizam diferentes estados de doenças é, também, uma importante área de aplicação de análise de dados de expressão gênica (ZHANG et al., 2018). Aná-

lises estatísticas tradicionais podem ser insuficientes para realizar essa tarefa, uma vez que o surgimento de certos tipos de doenças (como os cânceres) pode estar correlacionado de forma **não linear** com a presença de várias sobre-expressões (ou sub-expressões) de genes (LOPEZ-RINCON et al., 2018).

Em cânceres, pacientes com o mesmo status de doença podem ter respostas de tratamentos e desfechos clínicos diferentes (ZHANG et al., 2018). Os **cânceres** são, frequentemente, causados por diferentes mutações genéticas e apresentam considerável **heterogeneidade** fenotípica (LIANG et al., 2015) - pacientes com uma mesma classificação de tipo tumoral podem apresentar variações de: características moleculares; comportamento clínico; aparência morfológica; e respostas a terapias (SUN; WANG; LI, 2018).

Verificou-se que muitos cânceres incluem vários **subtipos**, em termos de diferentes desfechos clínicos e respostas a terapias (GUO; SHANG; LI, 2018). A identificação desses subtipos de câncer é vital para proporcionar avanços em terapia personalizada e diagnósticos precisos em oncologia (GUO; SHANG; LI, 2018; LIANG et al., 2015).

A predição do prognóstico de câncer com maior precisão poderia ajudar os médicos a tomar decisões informadas e orientar a terapia apropriada (SUN; WANG; LI, 2018). A identificação de subgrupos de sobrevida de cânceres, determinando subtipos mais agressivos, poderia melhorar significativamente o atendimento ao paciente (CHAUDHARY et al., 2017); uma vez que o tempo e a intensidade de terapia em câncer deve ser ajustada pela estimativa de prognóstico (normalmente, predita a partir de informações patológicas e clínicas). Mais recentemente, tem-se investigado o uso de assinaturas de expressão gênica para a definição do protocolo de terapia a ser adotado (SHARIFI-NOGHABI et al., 2018).

Por exemplo, a determinação precisa de risco de um paciente desenvolver uma doença metastática ou da agressividade do subtipo do tumor, permite determinar: pacientes que devem ser tratados com terapias mais intensivas; e pacientes que requerem uma menor intensidade de tratamento, de forma a evitar os sérios efeitos colaterais associados de uma terapia intensiva (SHARIFI-NOGHABI et al., 2018).

Ressalta-se que devido à heterogeneidade do tumor e sub-clones intra-tumorais, a previsão precisa de resposta a fármacos e a identificação de novos fármacos anticâncer ainda permanecem sendo tarefas desafiadoras (CHIU et al., 2018).

1.3 Objetivo Geral

Este trabalho visa investigar e estabelecer **abordagens computacionais de Aprendizagem de Máquina** para analisar dados de expressão gênica em câncer, tratando problemas de **classificação de amostras e seleção de genes (features)** relevantes que possam ser usados como potenciais biomarcadores ou alvos de fár-

macos.

1.4 Objetivos Específicos

Os objetivos específicos deste trabalho são listados a seguir.

- Estabelecimento de uma abordagem que permita mapear dados organizados como um conjunto de variáveis relacionadas para um multiarray, de forma que variáveis relacionadas fiquem próximas umas das outras; contribuindo para a análise desse tipo de dados por Convolutional Neural Networks (CNNs).
- Avaliação de diferentes técnicas de Deep Learning, especialmente as Redes Neurais Profundas Convolutional Neural Networks (CNNs) e o framework Generative Adversarial Network (GAN), e de métodos de Aprendizagem Rasa para classificação de dados de expressão gênica.
- Determinação das melhores abordagens para classificação de amostras de expressão gênica que possa ser escalável a diferentes datasets; com adequada capacidade de generalização, de forma a ter desempenho em dados sujeitos à maldição de dimensionalidade (número de features muito maior que o número de amostras).
- Avaliação e determinação das melhores abordagens para seleção de features (genes relevantes) que considere a alta redundância nos dados.

1.5 Organização do Trabalho

Neste capítulo, foi contextualizado o escopo do presente trabalho, destacando os desafios envolvidos em análises de dados de expressão gênica e ressaltando sua importância em estudos relacionados a cânceres. Então, apresentaram-se os objetivos do presente trabalho.

O Capítulo 2 apresenta o referencial teórico, são explicados alguns conceitos-chave relacionados a Redes Neurais Profundas; abordam-se as *Convolutional Neural Networks (CNNs)* e o *framework Generative Adversarial Networks (GAN)*. Seguindo, apresenta-se a técnica usada para mapeamento de dados organizados como um conjunto de variáveis relacionadas para um *multiarray*, visando favorecer a análise desse tipo de dados por CNNs.

Então, o Capítulo 3 apresenta os trabalhos relacionados, fazendo uma revisão sobre a aplicação de *Deep Learning (DL)* em problemas tratados em Bioinformática.

Após, são comentados artigos mais relacionados ao tema do presente trabalho - que aplicam DL em análises envolvendo dados de expressão gênica e câncer.

O Capítulo 4 apresenta a metodologia adotada para investigação dos métodos de Aprendizagem de Máquina. São descritos os datasets empregados e as abordagens originais propostas neste trabalho. Também, são descritos o procedimento para a escolha dos hiperparâmetros, comparação e execução dos métodos.

No Capítulo 5, propõe-se uma abordagem que utiliza o mecanismo de *dropout* para explorar um modelo de Rede Neural como um *ensemble* de modelos, objetivando aproveitar a alta capacidade de generalização normalmente obtida em *ensembles* de classificadores.

O Capítulo 6 explica as metodologias estabelecidas para a seleção de genes relevantes nas tarefas de classificação.

Então, os Capítulos 7 e 8 apresentam e discutem os resultados obtidos relacionados às abordagens de classificação e seleção de genes, respectivamente. Conforme expomos os resultados, discutimos as percepções alcançadas.

Finalizando, o Capítulo 9 resume os principais aspectos do trabalho, discutindo as contribuições e apresentando os trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 Aprendizagem de Máquina

Os métodos de **Aprendizagem de Máquina - *Machine Learning (ML)*** buscam aprender automaticamente a reconhecer padrões complexos com base em dados. O objetivo desses métodos é habilitar um algoritmo a aprender, com dados do passado ou presente, e usar esse conhecimento para fazer predições ou decisões para eventos futuros desconhecidos (LIN; LANE, 2017).

Esses métodos podem ser utilizados para resolver problemas que não sabemos como programar o computador para resolvê-los. Alguns dos algoritmos mais conhecidos em métodos de aprendizagem de máquinas incluem (BISHOP, 2006): árvore de decisão, redes neurais artificiais, máquina de vetor de suporte (SVM), k-Means, k-vizinhos mais próximos (kNN) e regressão.

Os métodos de ML podem ser classificados basicamente nas seguintes classes (GHEISARI; WANG; BHUIYAN, 2017).

- Métodos de **Aprendizagem Supervisionada** - aprendem um mapeamento entre entradas e saídas e requerem um conjunto de treinamento com dados rotulados (exemplos de entrada e saída esperada, ou seja, requerem atributos e rótulos). Quanto ao tipo de saída, esses métodos são classificados em: categóricos (classificação) quando a saída é um rótulo; e regressão quando a saída é um valor numérico real.

- Métodos de **Aprendizagem Não-supervisionada** - realizam redução dimensional; selecionam ou extraem atributos; agrupam dados; ou, ainda, descobrem relações entre variáveis.

Os métodos de ML também podem ser classificados quanto ao tipo de aprendizagem que pode ocorrer das seguintes formas (MAMOSHINA et al., 2016).

- **Aprendizagem rasa (*Shallow Learning*)** - forma mais convencional, por exemplo, através de métodos de Redes Neurais com poucas camadas de neurônios escondidos, Máquinas de Vetor Suporte (*Support Vector Machines - SVM*), etc.

- **Aprendizagem profunda (*Deep Learning*)** - envolve métodos com várias camadas hierárquicas de processamento não linear; por exemplo, utilizando Redes Neurais Profundas com muitas camadas de processamento.

Uma diferença a se considerar entre os dois tipos de aprendizagem (Rasa e Profunda) é que a aprendizagem rasa não trata bem dados brutos e, portanto, requer extensivo trabalho humano para configuração dos métodos e manutenção dos mesmos. Enquanto que a Aprendizagem Profunda aprende padrões em dados brutos de alta dimensionalidade com pouca necessidade de supervisão humana (LECUN; BENGIO; HINTON, 2015).

Essa questão é tratada por (BENGIO; COURVILLE; VINCENT, 2012) como otimização do *trade-off* entre largura e profundidade. Definindo que apenas um circuito profundo pode executar tarefas exponencialmente complexas sem requerer um número infinito de elementos.

O tipo de aprendizagem requerido em algumas tarefas complexas como reconhecimento de imagens e linguagem (ou até mesmo a replicação de estilos de pintura e a composição de músicas) envolve o aprendizado de uma representação a partir de dados brutos. Nesse caso, a Aprendizagem Profunda apresenta vantagens, especialmente, quando os dados são de estrutura hierárquica (MAMOSHINA et al., 2016).

O reconhecimento de imagem, por exemplo, envolve a aprendizagem de uma hierarquia progressiva de abstração, começando por bordas, formas, até o objeto por inteiro (LECUN; BENGIO; HINTON, 2015). As representações são formadas através de associações simples, usando, por exemplo, pixeis como dados brutos de entrada dos métodos. As camadas de uma *Deep Neural Network* extraem padrões mais sofisticados gradualmente.

Atualmente, os computadores podem identificar diferentes objetos em milhões de imagens distintas; podem distinguir imagens de dois objetos quase idênticos; ou completar uma frase. Esses e outros desenvolvimentos recentes em *Deep Learning* têm impulsionado o entusiasmo na comunidade de *Machine Learning*, com desempenho sem precedentes em muitas tarefas desafiadoras (MAMOSHINA et al., 2016).

2.2 Deep Neural Networks

Os vários tipos de **Redes Neurais Profundas - *Deep Neural Networks (DNNs)*** são os métodos predominantes em Aprendizagem Profunda - *Deep Learning (DL)*. Uma DNN é uma Rede Neural Artifical composta de várias camadas computacionais de processamento não-linear que analisam dados de forma hierárquica (MAMOSHINA et al., 2016; RAMPASEK; GOLDENBERG, 2016).

As Redes Neurais Artificiais são modelos computacionais inspirados no proces-

samento realizado pelo cérebro com várias unidades de processamento (neurônios) interconectadas (sinapses) (KHAZE; MASDARI; HOJJATKHAH, 2013).

Cada camada de uma DNN recebe uma entrada e produz uma saída, calculada como uma função não linear de uma combinação linear ponderada dos valores de entrada (RAMPASEK; GOLDENBERG, 2016).

A saída de uma camada torna-se a entrada da próxima camada de processamento, criando, através do 'empilhamento' de várias camadas, uma arquitetura profunda. Em cada camada sucessiva, os dados vão sendo representados de forma cada vez mais abstrata (RAVÌ et al., 2017). Esses modelos profundos podem, portanto, aprender uma abstração de dados altamente complexa (RAMPASEK; GOLDENBERG, 2016).

Apenas recentemente as DNNs mostraram resultados expressivos devido ao: aumento de poder computacional disponível (destaque ao processamento paralelo em GPU); maior quantidade de dados para treinamento de DNNs; e ao desenvolvimento de melhores algoritmos para treinamento das DNNs (LECUN; BENGIO; HINTON, 2015).

Uma consideração importante das DNNs são seus muitos hiperparâmetros (MAMOSHINA et al., 2016). Esses são arquitetônicos (como tamanhos de camadas e funções de transferência), tipos de otimização (como taxas de aprendizado e valores de *momentum*) ou de regularização, como as probabilidades de *dropout* (SRIVASTAVA et al., 2014) ou o nível de ruído injetado nas entradas. Portanto, o *fine-tuning* (aprimoramento) dos numerosos hiperparâmetros é uma das tarefas mais difíceis na implementação de soluções de DNN (MAMOSHINA et al., 2016).

2.3 Arquiteturas de DNNs

As DNNs podem ser construídas com uma variedade de arquiteturas distintas. As arquiteturas mais utilizadas são listadas a seguir (ZHANG et al., 2018).

- ***Redes Neurais Convolucionais - Convolutional Neural Network (CNN)*** (ALOM et al., 2018) - foram projetadas de forma a aproveitarem as relações estruturais (espaciais/temporais) em dados representados na forma de *multiarray*. As CNNs consideram que elementos próximos nessa representação são relacionados.

- ***Autoencoders*** (MENG; DING; XUE, 2017) - rede neural não-supervisionada cujo treinamento é feito para que a mesma aprenda a reproduzir, na saída, a entrada apresenta a ela.

- ***Rede Neural de Propagação de Crença - Deep Belief Network (DBN)*** (HUA; GUO; ZHAO, 2015) - é um modelo gráfico generativo e, também, uma classe

de rede neural profunda, composta por múltiplas camadas de variáveis latentes (variáveis escondidas, não observáveis), com conexões entre as camadas, mas sem conexões entre neurônios de mesma camada.

- ***Redes Neurais Recorrentes - Recurrent Neural Network (RNN)*** (LIPTON; BERKOWITZ; ELKAN, 2015) - visam representar comportamentos dinâmicos através de conexões em *loop* - realimentação de informação. Assim, é possível criar representações internas (memória) e, portanto, tratar informações temporais e sequenciais.

2.4 Rede Neural Convolucional - CNN

As Redes Neurais Convolucionais - *Convolutional Neural Networks* (CNNs) têm sido aplicadas em diversos problemas envolvendo imagem, vídeo, voz, texto e, até mesmo, em dados biológicos (LECUN; BENGIO; HINTON, 2015). As redes CNNs foram inspiradas em trabalhos sobre o córtex visual (NIEPERT; AHMED; KUTZKOV, 2016) e seu predecessor era denominado de *Neocognitron* (FUKUSHIMA; MIYAKE, 1982).

Essas redes neurais foram projetadas de forma a aproveitarem as relações estruturais locais (espaciais/temporais) em dados representados em forma de *multiarray* (normalmente 2D ou 3D). As CNNs consideram que elementos próximos nessa representação são relacionados entre si.

Existem várias arquiteturas distintas de CNNs: ResNet, AlexNet, Inception, VGG, entre outras. Mas todas essas arquiteturas apresentam camadas convolucionais. A maioria, também, apresenta camadas de *pooling* intercaladas entre as camadas convolucionais e, no final da rede neural, também, normalmente, apresenta camadas *fully-connected*. A Figura 1 mostra um esquema de CNN para classificação de imagens.

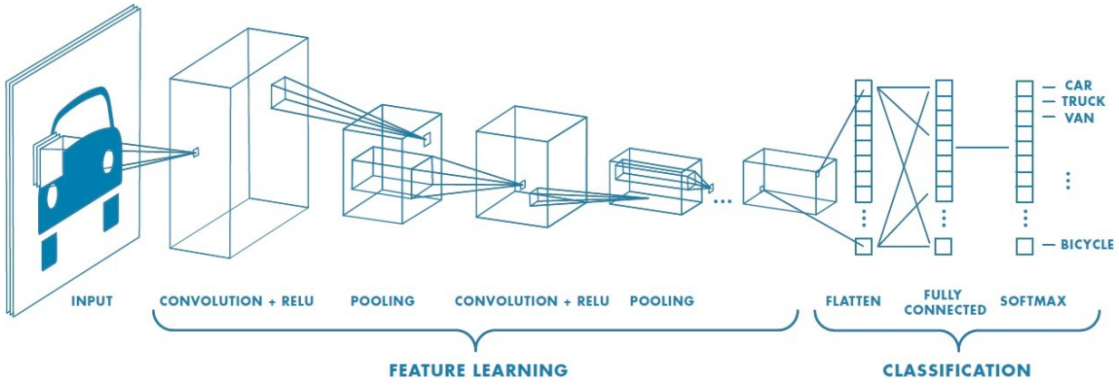

Figura 1 – Esquema de *Convolutional Neural Networks - CNNs* para classificação de imagens. Fonte (PRABHU, 2018).

O propósito de uma camada convolucional é a extração de padrões comuns encontrados dentro de regiões locais no conjunto de dados de entrada. Normalmente, cada camada convolucional apresenta um conjunto de N filtros que devem ser aprendidos.

As camadas convolucionais realizam operações de convolução em seus dados de entrada. A convolução (de cada filtro da camada) é implementada por uma janela deslizante com pesos (parâmetros que devem ser aprendidos). Para cada posição em que a janela deslizante é posicionada, os pesos do filtro são multiplicados pelos dados de uma região local da entrada. A convolução de um filtro aprendido produz, como resultado, tensores cuja profundidade é o número de filtros aplicados.

Assim, cada filtro de uma camada convolucional realiza transformações, com o mesmo conjunto de parâmetros, para as variadas regiões do *multiarray* dos dados de entrada. Dessa forma, as CNNs permitem que sejam executadas muitas transformações não lineares em dados de alta dimensionalidade sem requerer um enorme conjunto de parâmetros (a serem aprendidos) quanto os que seriam necessários se empregássemos as tradicionais redes *Multilayer Perceptron* (ao invés das CNNs).

As camadas de *pooling* são responsáveis por realizarem uma redução dimensional dos dados de entrada, favorecendo a redução do número de parâmetros da rede e, portanto, permitindo uma maior generalização na aprendizagem das CNNs.

As camadas de *pooling* operam em regiões locais na imagem, tal como as camadas convolucionais; entretanto, elas não aprendem filtros, não tendo parâmetros a serem aprendidos. Essas camadas apenas fazem determinada operação matemática em cada posição local dos dados de entrada.

Os tipos mais comuns de camadas de *pooling* são: a *Average Pooling* que realiza uma redução dimensional tomando apenas o valor da média dos elementos de cada região local da entrada; e a *Max Pooling* que toma apenas o valor do maior elemento de cada região local da entrada.

As camadas *fully-connected* são, normalmente, posicionadas ao final da rede e são responsáveis por interligar a informação dos diferentes *feature maps* obtidos (saídas dos filtros da camada anterior da rede). As camadas *fully-connected* apresentam vários neurônios, sendo que cada neurônio é conectado com cada elemento da saída da camada anterior.

2.4.1 ResNet

A ResNet foi proposta por (HE et al., 2016), essa arquitetura emprega módulos residuais (o termo ResNet50, comumente encontrado na literatura, refere-se a arquitetura ResNet com 50 camadas de neurônios).

Embora a ResNet seja muito mais profunda que a VGG de 16 ou 19 camadas, o número de parâmetros é menor. A redução de parâmetros é obtida devido ao uso do

global average pooling ao invés das camadas *fully-connected*.

Os autores (HE et al., 2016) tiveram uma percepção importante sobre as DNNs: eles notaram que, intuitivamente, quanto mais profunda fosse uma rede neural, melhor deveria ser seu aprendizado, ao menos em analisar o conjunto de treinamento (a acurácia não é diminuida pelo *overfitting*) (XU, 2017).

Por exemplo, uma rede com n camadas alcança uma certa precisão. No mínimo, uma rede com $n + 1$ camadas deveria conseguir exatamente a mesma precisão, se a última camada apenas passasse diretamente as suas entradas (um mapeamento identidade de suas entradas). Da mesma forma, redes com $n + 2$, $n + 3$ e $n + 4$ camadas poderiam continuar a executar mapeamentos identidade e obter a mesma precisão.

No entanto, na prática, as redes mais profundas degradavam o desempenho. Os autores da ResNet, então, pensaram na seguinte hipótese: os mapeamentos diretos são difíceis de aprender.

Então, (HE et al., 2016) propuseram o seguinte: ao invés de determinada camada tentar aprender um mapeamento de uma entrada x para o valor y, ela deve aprender a diferença entre y e o valor devolvido pela camada anterior, ou seja, o resíduo (diferença entre valor alvo e o que as camadas anteriores conseguiram aprender), que deve ser muito mais próximo de zero. Então, para se obter o valor alvo y na última camada, pode-se simplesmente adicionar o resíduo ao valor de entrada dessa camada.

Assim, (HE et al., 2016) definiram o bloco da ResNet (ou bloco de "rede residual") esquematizado na Figura 2 a seguir.

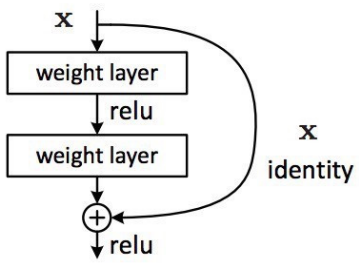

Figura 2 – Bloco da ResNet (fonte (SIMONYAN; ZISSERMAN, 2014) - adaptada).

Cada bloco da ResNet consiste em uma série de camadas e uma conexão de ‘atalho’ que adiciona a entrada do bloco à sua saída.

Essa ideia funciona de forma surpreendente na prática. Anteriormente, as redes neurais profundas, muitas vezes, sofriam do problema de fuga de gradientes (*vanishing gradients*) (XU, 2017), em que os sinais de gradiente da função de erro diminuíam exponencialmente à medida que eles retropropagavam para as camadas anteriores. A medida que os sinais de erro eram retropropagados de volta para as camadas iniciais, eles se tornavam tão pequenos que a rede não conseguia aprender.

Na ResNet, o sinal de gradiente pode retropropagar diretamente para camadas iniciais através das conexões de 'atalho' e, dessa forma, conseguem-se construir redes de 50 camadas, 101 camadas, 152 camadas e até 1000 camadas que ainda funcionam bem.

2.5 Generative Adversarial Network - GAN

O framework *Generative Adversarial Network* (GAN) foi proposto por (GOODFELLOW et al., 2014) e consiste em um modelo generativo não supervisionado. A GAN objetiva estimar um modelo generativo através de um processo adversarial, na qual são treinados, simultaneamente, duas redes neurais (ver Figura 3):

- uma rede neural geradora G que visa capturar a distribuição dos dados de entrada de um *dataset* de treinamento; e
- uma rede neural discriminadora D que estima a probabilidade de cada amostra apresentada vir de dados do conjunto de treinamento ou do gerador G.

Dessa forma, G estará sendo treinada para que maximize a probabilidade de D cometer erros.

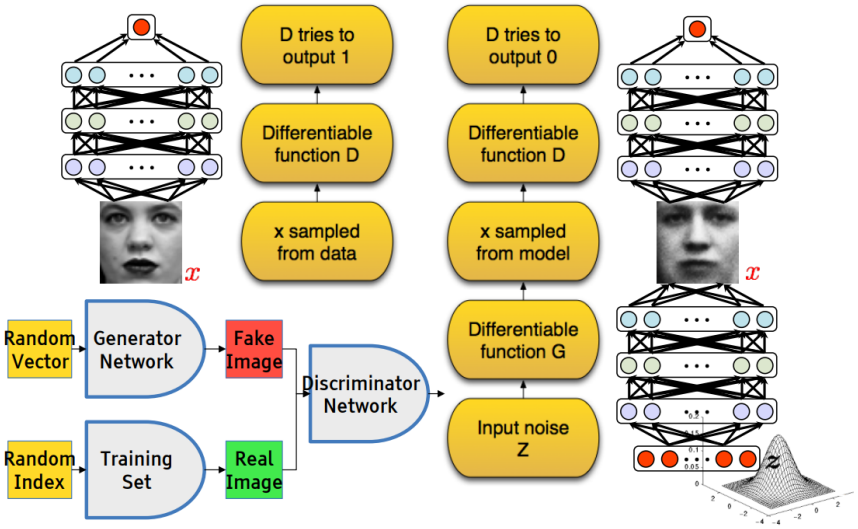

Figura 3 – Esquema da *Generative Adversarial Network - GAN*. Fonte (GOODFELLOW, 2016).

O gerador G aprende o mapeamento entre um vetor de variáveis $\boldsymbol{z}$ de uma distribuição definida a priori $P_{pri}(\boldsymbol{z})$ para uma saída $\boldsymbol{x} = G(\boldsymbol{z})$. Na formulação original da GAN (GOODFELLOW et al., 2014), o vetor de variáveis z que constituem a entrada da rede geradora G é um ruído aleatório. O discriminador $D : \boldsymbol{x} \rightarrow [1, 0]$ aprende a discriminar entre amostras produzidas pelo gerador (amostras falsas - rótulo 0) e amostras reais obtidas do *dataset* de treinamento (amostras verdadeiras - rótulo 1).

Para que o gerador produza amostras da distribuição $P_{in}(\boldsymbol{x})$ (*in-distribution*) dos dados de entrada (dados do *dataset* modelado pela GAN), segue-se uma estratégia de otimização minimax da seguinte função objetivo:

$$\min_{G}\max_{D}\mathbb{E}_{P_{in}(\boldsymbol{x})}[logD(x)] + \mathbb{E}_{P_{pri}(\boldsymbol{z})}[log(1 - D(G(z)))] \tag{1}$$

- $\mathbb{E}_{P_{in}(\boldsymbol{x})}[logD(x)] \rightarrow$ esse termo representa a média de acertos do discriminador em classificar corretamente uma amostra verdadeira do *dataset* como pertencentes à distribuição dos dados de entrada.

- $\mathbb{E}_{P_{pri}(\boldsymbol{z})}[log(1 - D(G(z)))] \rightarrow$ esse termo representa as chances do discriminador em classificar corretamente uma amostra falsa (produzida pelo gerador) como não pertencente à distribuição dos dados de entrada.

Portanto, o gerador G busca minimizar (min) a função objetivo acima, ou seja, diminuir os acertos do discriminador. O discriminador D, por sua vez, tenta maximizar (max) a função objetivo, ou seja, os seus acertos.

Essa estratégia pode ser comparada a otimização Minimax para jogos de dois jogadores competindo entre si. Como explicado por (GOODFELLOW et al., 2014), ambos G e D melhoram com o treinamento, havendo apenas uma única solução para os modelos G e D, na qual G aprende a distribuição dos dados de entrada e D devolve, para todas as amostras vindas de G ou do conjunto de treinamento, a probabilidade de 0.5 (metade) de ser dos dados de treinamento e metade de ser de G.

2.5.1 MetaGAN

Métodos de *few-shot learning* buscam realizar a aprendizagem de modelos a partir de poucas amostras. Os métodos de *few-shot learning* podem se basear:

- na aprendizagem de uma métrica compartilhada entre tarefas similares, ou seja, aprendem uma métrica de distância entre classes, em tarefas similares, que pode ser utilizada para aprender uma nova tarefa a partir de poucas amostras (SUNG et al., 2018); ou

- na aprendizagem de uma forma de aumentar um classificador treinado em um conjunto de tarefas similares para uma nova tarefa, adaptando os gradientes desse modelo (FINN; ABBEEL; LEVINE, 2017).

(ZHANG et al., 2018) propuseram um *framework*, denominado de MetaGAN, baseado no *framework Generative Adversarial Network* (GAN), para gerar amostras *fakes* que podem ser úteis para a aprendizagem de tarefas com *few-shot learning* (com poucas amostras). O *framework* MetaGAN deve ser integrado com algum método

classificador tradicional de *few-shot learning*, que ocupará o papel do discriminador da GAN (ver Seção 2.5). O classificador tradicional de *few-shot learning* tem o número de classes ampliadas em um elemento, para representar as amostras detectadas como *fake*.

A ideia por traz do MetaGAN é que os modelos geradores da GAN são imperfeitos e que, portanto, podem fornecer amostras *fake* de muitas classes reais que envolvam tarefas similares às usadas no treinamento dos modelos da GAN. Dessa forma, essas amostras *fake* podem ser empregadas para explorar o espaço amostral de muitas classes similares.

2.6 Transfer Learning

Transfer Learning (Transferência de Aprendizagem) é uma técnica em *machine learning* que visa 'transferir' o conhecimento de um domínio fonte para um domínio alvo no qual podem-se ter poucas amostras.

Essa técnica envolve a inicialização dos pesos de um modelo a partir de um modelo pré-treinado em outra base de dados (não alvo), como por exemplo a ImageNet. O modelo inicializado pode ser utilizado como um extrator de atributos (*feature extractor*), ou ainda, pode-se empregar *fine-tuning* (ajuste de parâmetros) das últimas camadas do modelo, utilizando as amostras do domínio alvo.

2.6.1 Bases de Dados ImageNet

ImageNet (LAB, 2018a) é um projeto que visa a anotação (classificação) manual de imagens de quase 22.000 categorias distintas de objetos para fins de pesquisa em visão computacional.

O termo ImageNet também é utilizado, no contexto de Deep Learning, para se referir ao *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) (LAB, 2018b). O ILSVRC é uma competição (um desafio) na área de classificação de imagens cujo objetivo é o treinamento de um modelo capaz de classificar imagens de 1.000 categorias distintas.

Nesse desafio, os modelos são treinados com cerca de 1.2 milhões de imagens de treinamento e 50.000 imagens de validação. Após, os modelos são avaliados com um conjunto de teste de 100.000 imagens.

As categorias de objetos do ImageNet são objetos do cotidiano como espécies de cães, gatos, tipos de veículos etc. Assim sendo, o ImageNet (ILSVRC) é um *benchmark* para o desenvolvimento de algoritmos de classificação.

Desde de 2012, os modelos vencedores do desafio foram treinados por CNNs. As CNNs pré-treinadas no ImageNet demonstram uma forte capacidade de generalização

de forma que podem ser empregadas para classificar novas imagens de objetos de novas classes (distintas daquelas existentes no ImageNet) através de *Transfer Learning* (Transferência de Aprendizagem), com abordagens como extrações de atributos (*feature extraction*) e *fine-tuning*.

2.7 Regularização - Dropout

A técnica Dropout visa evitar o *overfitting* (especialização do modelo) e consiste em, a cada passo do treinamento, escolher aleatoriamente um conjunto de neurônios para serem ignorados. Dessa foma, evita-se a inter-dependência entre neurônios: cada neurônio deverá realizar a aprendizagem sem depender da entrada recebida de outros neurônios.

2.8 Transcriptograma

O ordenamento do Transcriptograma (*Transcriptogramer*) é proposto para mapeamento de dados organizados como um conjunto de variáveis relacionadas para uma representação de *multiarray*.

Transcriptograma (*Transcriptogramer*) (MORAIS; ALMEIDA; DALMOLIN, 2019) (SILVA et al., 2014) (RYBARCZYK-FILHO et al., 2010) é uma técnica proposta para ser utilizada em bioinformática para possibilitar a comparação de dados de expressão gênica (transcriptômicos) entre duas classes fenotípicas distintas (como por exemplo, amostras cancerosas versus amostras de tecido normal).

Essa técnica permite evidenciar processos biológicos que apresentam maiores diferenças de expressão entre duas classes fenotípicas e possibilita aumentar a razão sinal-ruído, mitigando o problema de ruído inerente as técnicas de estimativas de expressão gênica, como RNAseq e microarray.

O Transcriptograma é dividido em duas etapas: i) ordenamento de genes/proteínas - visa agrupar genes relacionados, utilizando informações de redes biológicas (de interação proteína-proteína ou de interação gênica); ii) cálculo de médias - funciona como um filtro de média móvel obtido por uma janela deslizante que visa diminuir os ruídos associados aos dados.

2.8.1 Ordenamento

Primeiramente, é realizada a etapa de ordenação dos genes de forma a posicionar genes relacionados próximos uns com os outros. Para isso, é empregada a informa-

ção de dados de redes de interação gênica e Interação Proteína-Proteína (PPI) de *databases* como o STRING (STRING, 2019) e o BioGRID (BIOGRID, 2019).

Esses *databases* organizam a informação de relação entre proteínas / genes como um grafo, cujas arestas representam a confiabilidade da relação entre duas proteínas ou genes adjacentes.

O grafo contido no *database* STRING é montado a partir de mineração de informações publicadas na literatura; databases de experimentos de interação; dados de co-expressão; e transferência sistemática de evidências de interação de um organismo para outro (SZKLARCZYK et al., 2021). Enquanto que, o grafo contido no BioGRID é curado manualmente, as interações analisadas são sugeridas por algoritmos de mineração de texto e submissões diretas de usuários (OUGHTRED et al., 2019).

Quando se está trabalhando com redes PPI, as proteínas são mapeadas para os seus respectivos genes de forma que cada nodo do grafo seja correspondente a um gene específico.

Para o ordenamento, as arestas com certo grau de confiabilidade são mantidas e aquelas abaixo de certo limiar são desconsideradas. O grafo resultante é representado por uma matriz de adjacências contendo 1's (presença de aresta) e 0's (sem aresta entre os respectivos pares de genes).

O ordenamento pode ser feito de forma linear 1D (RYBARCZYK-FILHO et al., 2010) - organização dos genes ordenadamente em uma linha; ou ainda, bidimensional 2D (PERRONE, 2013), dispondo os genes em uma matriz. O processo do ordenamento é feito através do método de Monte Carlo com *Simulated Annealing*.

Primeiramente, é atribuído a cada gene uma posição aleatória (por exemplo, um índice numa lista - para o ordenamento 1D). Iterativamente, o método seleciona aleatoriamente dois genes e os troca de posição e, consecutivamente, é obtida uma nova matriz de adjacências do grafo da rede de interações (obtida pela troca de linhas e colunas referentes aos genes cujas oposições foram trocadas).

Então, calcula-se uma função de energia referente a nova configuração da matriz de adjacências. Essa função de energia é tanto maior quanto forem as interfaces entre 1's e 0's na matriz de adjacências e quanto maior as distâncias das arestas à diagonal da matriz de adjacências. Assim, ao se minimizar a energia, formam-se grupos de genes relacionados (com arestas entre si) próximos uns dos outros. A seguir, na Equação 2, é apresentada a função de energia E para uma matriz de adjacências com elementos $a_{i,j}$ e N nodos. α é um parâmetro que modela a influência (na composição da energia) da distância da aresta à diagonal.

$$E = \sum_{i=1}^{N} \sum_{j=1}^{N} a_{i,j} |i-j|^{\alpha} (4 - a_{i,j+1} - a_{i,j-1} - a_{i+1,j} - a_{i-1,j}) \tag{2}$$

A nova configuração da matriz de adjacências é aceita caso a energia for menor

do que a anterior. Caso contrário, a nova configuração é aceita com a probabilidade $e^{-\delta E/T}$, onde T é a temperatura virtual do *Simulated Annealing*.

A seguir, a Figura 4 mostra a configuração de uma matriz de adjacências após o ordenamento, os pontos pretos representam as arestas (1's na matriz) para valores de $\alpha = 1$ e $\alpha = 10$.

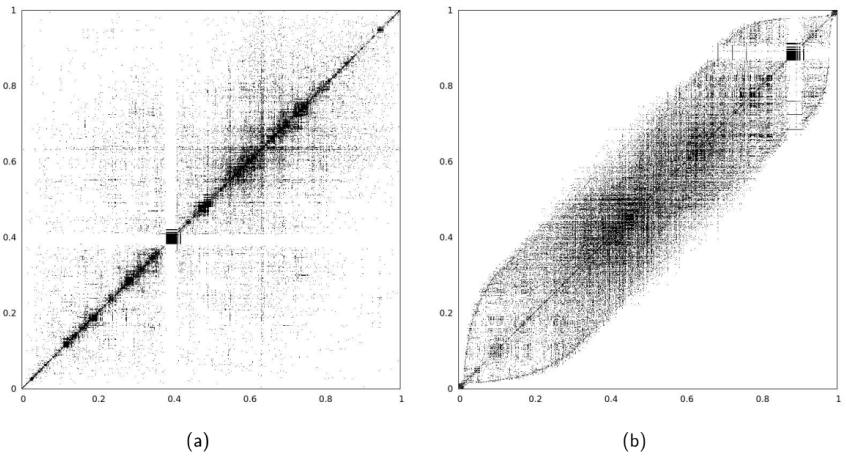

Figura 4 – Configuração final de uma matriz de adjacências, após o ordenamento, os pontos pretos representam as arestas (1's na matriz). a) $\alpha = 1$ b) $\alpha = 10$. Fonte (KUENTZER, 2014).

2.8.2 Cálculo de Médias

Tendo uma posição de ordem para todos os genes, no qual os relacionados estejam próximos entre si, pode-se realizar uma filtragem para diminuir ruídos; bem como, podem-se realizar comparações em relação a processos biológicos (uma vez que os genes que participam de um mesmo processo biológico devem estar agrupados nessa ordem).

Para diminuir o ruído (etapa de cálculos de médias) utiliza-se um filtro passa baixa: filtro de média móvel. Uma janela deslizante é passada por cada amostra de expressão gênica: para cada posição que a janela ocupa, o valor correspondente ao centro dessa janela é substituído pela média do conjunto de genes na janela.

2.8.3 Comparação entre Classes

Tendo os genes relacionados próximos e removido os ruídos, podem-se realizar comparações entre duas classes fenotípicas. A média e o desvio padrão para cada gene de cada uma das classes é calculado e dispostos em um gráfico.

As regiões em que ocorre grande disparidade entre as duas classes - média de expressão muito diferente, considerando o valor desvio padrão para aquela região - podem ser utilizados para destacar as vias biológicas cuja expressão gênica está mais diferenciada entre as duas classes.

A Figura 5, a seguir, mostra o exemplo de um gráfico gerado pelo Transcriptograma para comparação entre duas classes, as médias de expressão por posição são diminuídas pela média de expresão da classe controle, de forma que a expressão da classe controle possa ser visualizada como uma linha (preta) no zero e a linha cinza mostra a diferença de expressão da classe comparada em relação a controle. Os pontos em colorido que correspondem a picos da diferença podem ser associados a processos biológicos como demonstrado.

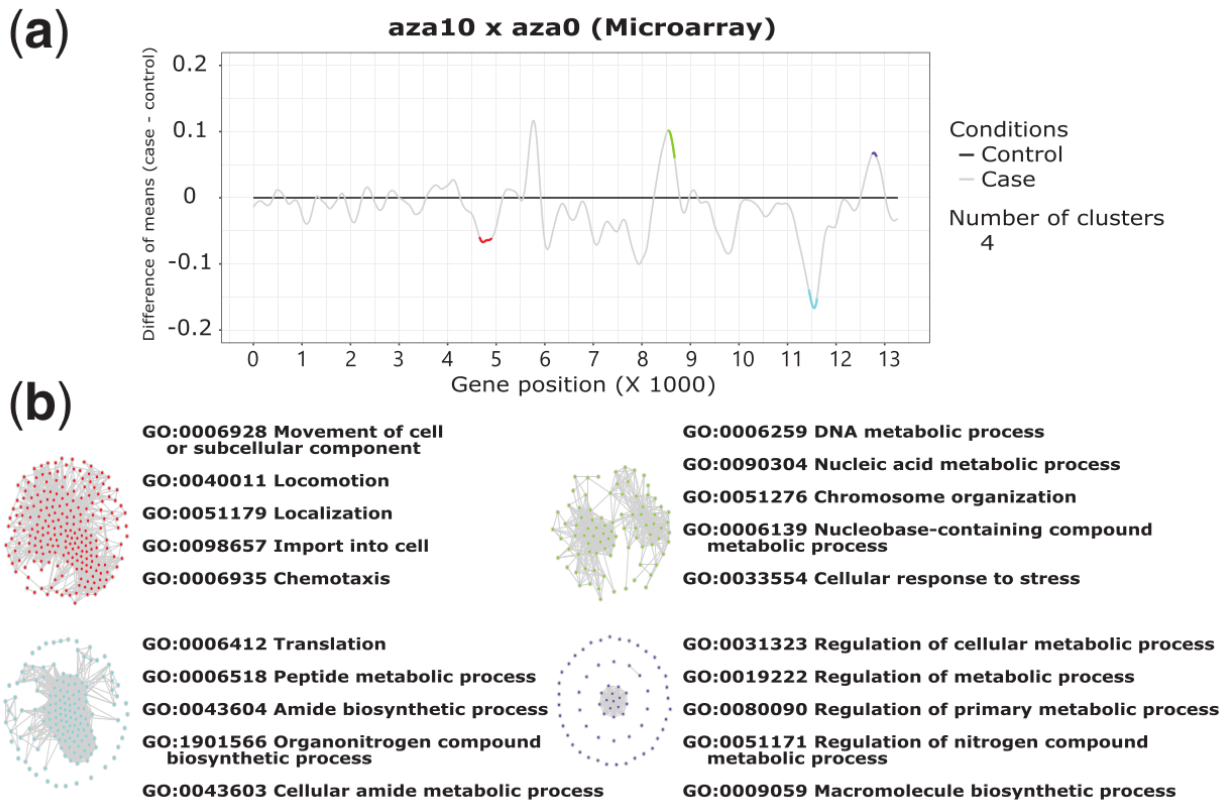

Figura 5 – Gráfico montado pelo Transcriptograma - diferença de expressão entre a classe controle (preto) e a classe comparada (cinza). Os picos coloridos são associados a processos biológicos. Fonte (MORAIS; ALMEIDA; DALMOLIN, 2019).

2.8.4 Utilização do Transcriptograma para Estabelecer a Entrada de CNNs

O ordenamento do Transcriptograma fornece uma ordem: posição para cada um dos genes em uma lista ordenada, na qual genes com funções relacionadas estão próximos uns dos outros.

Essa ordem pode ser utilizada para montar as entradas de uma CNN. Nas CNNs 1D, a expressão dos genes é, então, disposta em uma linha, em que a expressão de cada gene é colocada na posição determinada pelo Transcriptograma. Nas CNNs 2D, por sua vez, a lista ordenada de expressões gênicas é disposta em zigzag, condensada em uma imagem.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta uma revisão sobre a aplicação de **Deep Learning (DL)** em problemas tratados em **Bioinformática**, com o intuito de traçar um panorama geral das pesquisas na área. Após, foca-se nos artigos mais relacionados ao tema do presente trabalho - que apliquem DL em análises envolvendo dados de **expressão gênica** e **câncer**.

Nesse trabalho, realizou-se a seleção de uma lista de artigos que aplicam de *Deep Learning* (DL) em problemas de bioinformática nas áreas ômicas (genômica, transcriptômica, proteômica), a partir do ano de 2012 a 2018. A seguinte *query* foi pesquisada no *Google Scholar*: *"deep learning"AND ("autoencoder"OR "autoencoders"OR "neural network"OR "neural networks"OR "generative adversarial network"OR "generative adversarial networks"OR "lstm"OR "lstms"OR "dbn"OR "dbns") AND (protein OR rna OR gene OR dna).*

Após, realizou-se a análise de título, *abstract* e leitura superficial (quando necessária) do artigo, de forma a determinar se cada artigo realmente pertencia à área de interesse.

A fim de refinar o escopo da revisão bibliográfica, foram retirados, da seleção obtida, trabalhos cujo foco principal era a análise de: imagens (histopatológicas, ressonância, tomografia); sinais de eletrocardiograma, eletroencefalograma e eletromiografia; sinais e prontuários clínicos; mineração de texto e determinação de entidade-relacionamento através de textos (literatura). Considera-se que tais trabalhos têm maior proximidade com a área de visão computacional, processamento de imagens, mineração de texto, processamento de linguagem natural e informática médica do que com a bioinformática propriamente dita.

Observa-se que nos anos de 2012 e 2013, tiveram-se poucos trabalhos, muitos dos quais visavam revisar métodos de DL e especular possíveis aplicações desses métodos em problemas biológicos. Percebe-se que a DL demorou um pouco mais para despontar em aplicações de bioinformática do que em outras áreas como visão computacional e processamento de linguagem. Entretanto, a partir do ano de 2016, o número de artigos sendo publicados na área se torna expressivo, observa-se um

crescimento exponencial de trabalhos publicados.

Atualmente, os métodos de DL podem ser vistos como um ferramental para abordar diversificados problemas. Assim, encontram-se trabalhos, na literatura, mais focados no tratamento dos dados biológicos para serem empregados na DL; como, também, existem trabalhos que tratam mais da parte computacional de desenvolvimento de novos métodos de DL com propósito direto de serem aplicados à bioinformática; e ainda existem trabalhos que apenas utilizam *datasets* biológicos como *benchmarks* para validar sua metodologia (sem abordar o problema biológico propriamente dito).

A seleção de artigos está disponibilizada nos arquivos suplementares. Este trabalho não se propõe a descrever toda a lista de trabalhos selecionados na revisão (total de 1210 artigos); são abordados apenas artigos mais recentes e aqueles relacionados à área do presente trabalho: análise de expressão gênica em câncer.

3.1 Aplicações

Nos últimos anos, muitos trabalhos foram publicados sobre a aplicação de *Deep Learning* em uma variedade de problemas biológicos. A Figura 6 organiza as principais aplicações investigadas pelos trabalhos selecionados na revisão, citando algumas referências recentes. Os trabalhos foram classificados nas áreas Genômica (DNA), Transcriptômica (RNA), Proteômica (proteínas) e Fármacos de acordo com o principal objetivo da aplicação e dos tipos de dados analisados. Entretanto, ressalta-se que vários trabalhos empregam mais do que um tipo de dados (de mais do que uma dessas áreas).

Figura 6 – Aplicações de Deep Learning em Bioinformática (CUI et al., 2018; FIORAVANTI et al., 2018; HUANG; LIAO; WU, 2018; CHUAI et al., 2018; WU; WEI, 2018; KISHAN et al., 2018; MÜLLER; HISS; SCHNEIDER, 2018; AMIDI et al., 2018; MERK et al., 2018; KIM et al., 2018; ONIMARU; NISHIMURA; KURAKU, 2018; NGUYEN et al., 2018; LEI et al., 2018; WANG; CHENG, 2018; ZHOU et al., 2018; GUAN et al., 2018; GUO et al., 2018; REN et al., 2018; ZHOU et al., 2018; KAMARUDIN et al., 2018; GAO et al., 2018; YADAV; GUPTA; BIST, 2018; JING et al., 2018; SALEKIN, 2018; OUYANG et al., 2018; BAEK et al., 2018; YANG et al., 2018; CAO et al., 2018; PAN et al., 2018; RYU; LIM; KIM, 2018; JIMENEZ-CARRETERO et al., 2018; SUNSERI et al., 2018; JACOBS et al., 2018; SERRANO et al., 2018; FENG et al., 2018; DU et al., 2018; GAO et al., 2018; CHANG et al., 2018; ÖZTÜRK; OZKIRIMLI; ÖZGÜR, 2018; CHENG et al., 2018; WICK; JUDD; HOLT, 2018; KIM et al., 2018; LOPEZ-RINCON et al., 2018; KARIMI et al., 2018; WANG et al., 2018; MIRANDA; HU, 2018; NAZARI; CHONG, 2018; HAMID; FRIEDBERG, 2018; SHARIFI-NOGHABI et al., 2018; GAO; YANG; ZHOU, 2018; WANG; HOEKSEMA; LIANG, 2018; BRETSCHNEIDER et al., 2018; DIZAJI; WANG; HUANG, 2018; PFEIFFENBERGER; BATES, 2018; RENHULDT, 2018).

Um trabalho que merece ser destacado é o sistema denominado de AlphaFold (JUMPER et al., 2021) que alcançou resultados sem precedentes na predição de estrutura terciária de proteínas, recebendo reconhecimento na comunidade científica em geral. O AlphaFold é capaz de predizer a estrutura terciária de proteínas a partir das suas sequências de aminoácidos e do alinhamento com sequências homólogas.

Esse sistema foi desenvolvido pela DeepMind Google e alcançou o primeiro lugar no concurso *14th Critical Assessment of protein Structure Prediction (CASP14)*, produzindo predições com acurácia acima de uma larga margem da segunda melhor abordagem. O AlphaFold utiliza um conjunto de diferenciais: uma nova arquitetura de *Deep Neural Network* constituída de blocos com e sem mecanismo de atenção em um esquema inspirado nas redes neurais *Transformers*; uma abordagem que possibilita um refinamento iterativo das predições; *self-distillation* do conhecimento; entre outros.

O Apêndice A relaciona alguns dos trabalhos recentes encontrados pela revisão sistemática, especificando as técnicas de DL utilizadas. A próxima seção comenta trabalhos mais relacionados ao tema do presente trabalho.

3.2 Métodos Relacionados - Expressão Gênica

Para restringir o número de artigos a serem analisados mais detalhadamente, selecionou-se, do conjunto de trabalhos obtidos pela revisão, 25 artigos que: i) empregavam, como entrada de suas abordagens de DL, dados de expressão gênica, expressão de microRNAs (isoformas de microRNAs) ou que realizavam inferência de níveis de expressão gênica a partir de outros dados transcriptômicos; e ii) buscavam obter informações biológicas que claramente possam ser associadas ao estudo de cânceres.

Não se esgotou a lista de todos os artigos que entram nesses critérios, mas acredita-se que esse conjunto forneça uma boa representatividade dos artigos existentes relacionados ao presente trabalho. A Figura 7 lista os artigos selecionados, que são detalhados, logo a seguir. Os trabalhos são comentados em ordem cronológica, seguida, de ordem alfabética. A cor disposta, na figura, para cada trabalho, se mantém a mesma nos esquemas das demais figuras.

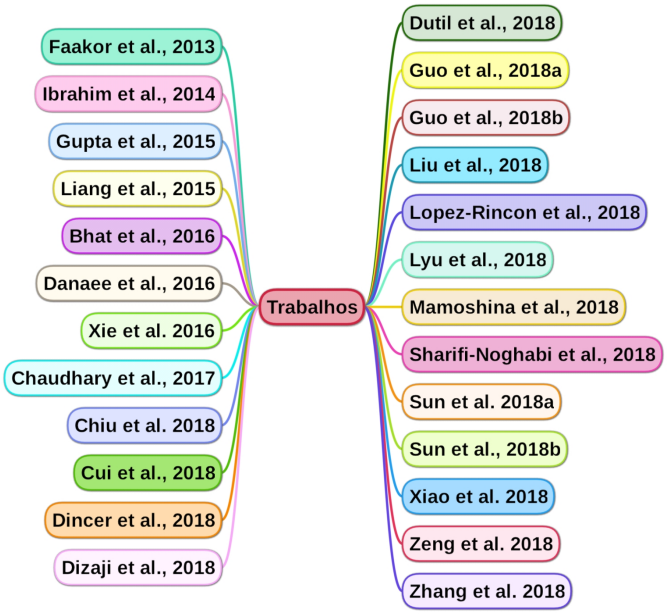

Figura 7 – Trabalhos selecionados para serem detalhados, ordenados em ordem cronológica, seguida da alfabética. Para efeitos de organização, a cor é mantida a mesma para cada trabalho, em todos os esquemas das figuras a seguir - (FAKOOR et al., 2013; IBRAHIM et al., 2014; GUPTA; WANG; GANAPATHIRAJU, 2015; LIANG et al., 2015; BHAT; VISWANATH; LI, 2016; DANAEE; GHAEINI; HENDRIX, 2016; XIE et al., 2016; CHAUDHARY et al., 2017; CHIU et al., 2018; CUI et al., 2018; DINCER et al., 2018; DIZAJI; WANG; HUANG, 2018; DUTIL et al., 2018; GUO et al., 2018; GUO; SHANG; LI, 2018; LIU et al., 2018; LOPEZ-RINCON et al., 2018; LYU; HAQUE, 2018; MAMOSHINA et al., 2018; SHARIFI-NOGHABI et al., 2018; SUN et al., 2018; SUN; WANG; LI, 2018; XIAO et al., 2018; ZENG et al., 2018; ZHANG et al., 2018).

A Figura 8 esquematiza as técnicas de *Deep Learning* usadas nos trabalhos selecionados.

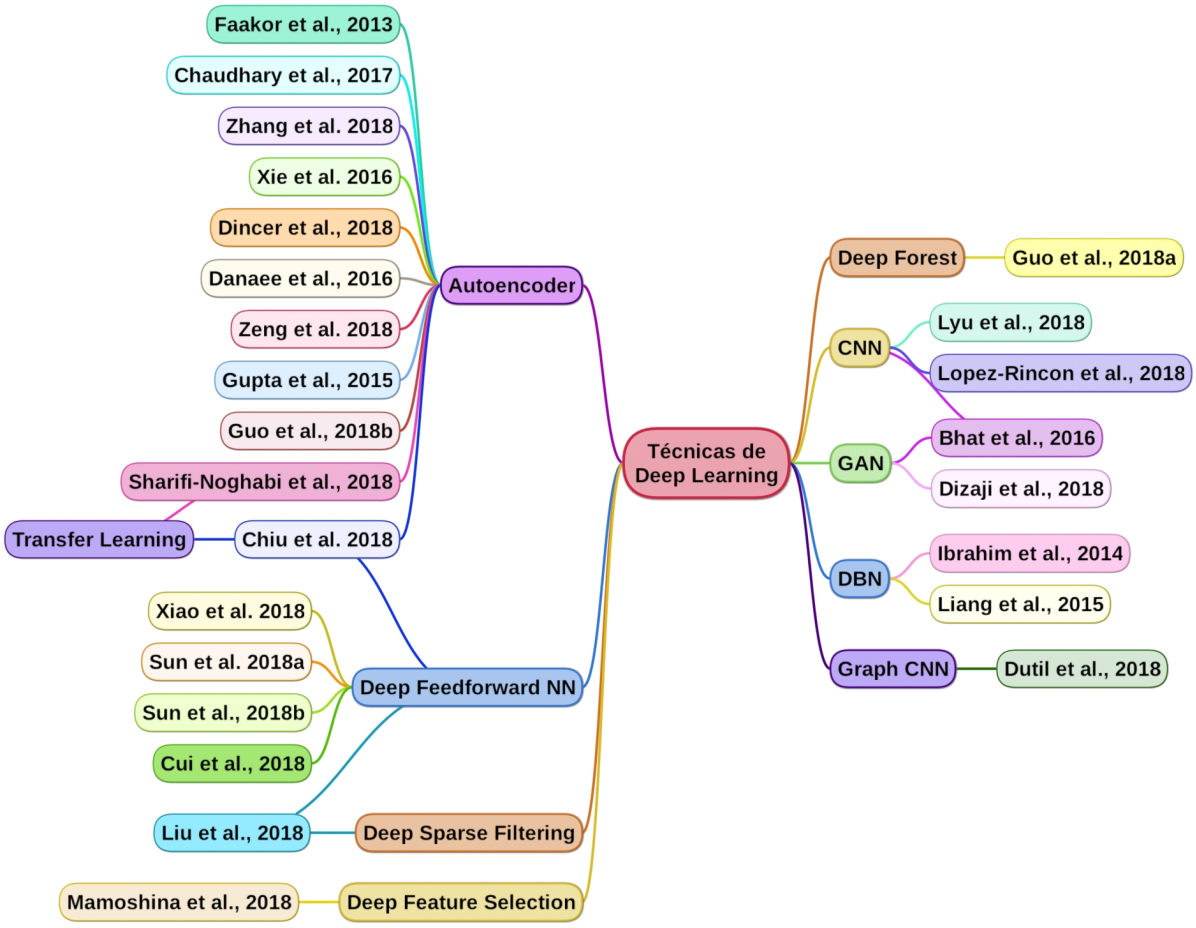

Figura 8 – Técnicas de *Deep Learning* usadas nos trabalhos selecionados.

A maioria dos trabalhos (16 artigos - (FAKOOR et al., 2013), (IBRAHIM et al., 2014), (GUPTA; WANG; GANAPATHIRAJU, 2015), (LIANG et al., 2015), (BHAT; VISWANATH; LI, 2016), (DANAEE; GHAEINI; HENDRIX, 2016), (CHAUDHARY et al., 2017), (CHIU et al., 2018), (DINCER et al., 2018), (DIZAJI; WANG; HUANG, 2018), (GUO; SHANG; LI, 2018), (LIU et al., 2018), (SHARIFI-NOGHABI et al., 2018), (XIAO et al., 2018), (ZENG et al., 2018), (ZHANG et al., 2018)) empregou abordagens não supervisionadas - *Autoencoder, Generative Adversarial Network (GAN), Deep Belief Network (DBN)* e *Deep Sparse Filtering*.

Enquanto que, 9 artigos (CUI et al., 2018), (DUTIL et al., 2018), (GUO et al., 2018), (LOPEZ-RINCON et al., 2018), (LYU; HAQUE, 2018), (MAMOSHINA et al., 2018), (SUN et al., 2018), (SUN; WANG; LI, 2018) ,(XIAO et al., 2018) usaram apenas abordagens supervisionadas - *Deep Feedforward Neural Network, Convolutional Neural Network (CNN), Graph Convolutional Neural Network* e *Deep Forest*.

A técnica de *Transfer Learning* pode ser, também, aplicada em métodos de Aprendizagem Rasa, mas, como se tem mostrado importante em *Deep Learning*, foi destacada no esquema da figura.

Percebe-se a grande importância dada a abordagens não supervisionadas, devido aos fatores listados a seguir.

- Muitas vezes, têm-se amostras não rotuladas, que podem ser aproveitadas por essas abordagens (CHIU et al., 2018; DIZAJI; WANG; HUANG, 2018). A falta de rótulo pode ser por causa de: perdas de acompanhamento de pacientes; falta de execução de análises adicionais para todas amostras, que podem ser caras de serem feitas em larga escala.
- Para alguns problemas, não se tem o conhecimento biológico para determinar rótulos associados às amostras - o objetivo do próprio método de *Machine Learning* pode ser adquirir esse conhecimento.

  Por exemplo, pode-se desenvolver abordagens não supervisionadas para: encontrar subtipos de determinado tumor (LIANG et al., 2015; GUO; SHANG; LI, 2018) (estabelecendo pacientes que poderiam ser tratados por uma mesma terapia, para propiciar estudos de medicina personalizada e previsão de prognóstico); definir genes relevantes (*feature selection*) (LIU et al., 2018; IBRAHIM et al., 2014), para a obtenção de biomarcadores e determinação de possíveis alvos para o desenvolvimento de novos fármacos.
- Mesmo que se tenham rótulos para as amostras, pode ser interessante utilizar métodos não supervisionados para: redução dimensional; aprendizagem de *features*, para facilitar o tratamento de determinado problema; remoção de ruídos e generalização da distribuição do *dataset* de treinamento (FAKOOR et al., 2013), (GUPTA; WANG; GANAPATHIRAJU, 2015), (BHAT; VISWANATH; LI, 2016), (DANAEE; GHAEINI; HENDRIX, 2016), (XIE et al., 2016), (CHAUDHARY et al., 2017), (CHIU et al., 2018), (DINCER et al., 2018), (DIZAJI; WANG; HUANG, 2018), (GUO; SHANG; LI, 2018), (SHARIFI-NOGHABI et al., 2018), (ZENG et al., 2018), (ZHANG et al., 2018).

Os métodos não supervisionados como *Autoencoder*, *Deep Belief Network* e, mais recentemente, a GAN podem ser valiosos para essas tarefas. Muitos autores ressaltaram a importância de redução dimensional para análises de expressão gênica (FAKOOR et al., 2013; DANAEE; GHAEINI; HENDRIX, 2016; CHIU et al., 2018; DINCER et al., 2018), uma vez que a dimensionalidade de cada amostra é muito alta em relação ao número de amostras disponíveis.

A Figura 9 esquematiza os tipos de dados usados, como entrada, nos métodos apresentados pelos artigos selecionados. Todos os trabalhos empregaram, como entrada, dados de expressão (gênica ou de microRNAs), com exceção de (XIE et al.,

2016), que inferiram expressão gênica a partir de dados de SNPs (*Single Nucleotide Polymorphisms*).

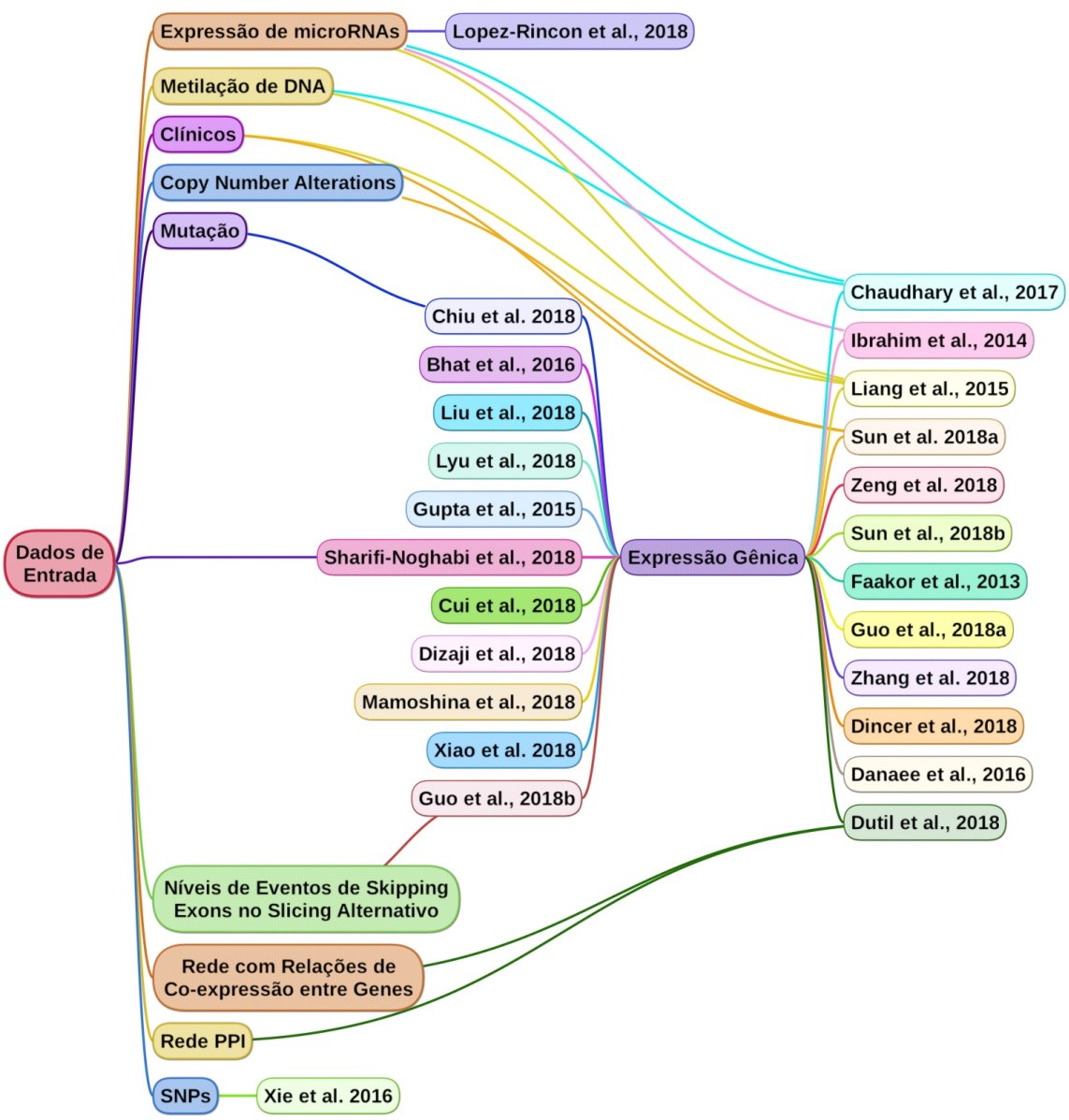

Figura 9 – Dados de entrada dos métodos propostos pelos trabalhos selecionados.

A Figura 10 esquematiza os trabalhos segundo a aplicação foco das abordagens propostas pelos trabalhos selecionados. A maioria dos trabalhos executou uma tarefa de classificação ou regressão; dentre esses trabalhos, alguns estabeleceram, ainda, metodologias para a determinação de assinatura gênica ou de microRNAs, ou seja, seleção de *features* com poder de discriminação na tarefa executada.

Além desses trabalhos, (GUPTA; WANG; GANAPATHIRAJU, 2015; LIANG et al., 2015; GUAN et al., 2018) executaram tarefas de agrupamento. (ZENG et al., 2018) investigaram uma abordagem para selecionar amostras de tecido saudável para comparação com amostras de câncer (amostras de diferentes *datasets* e indivíduos). (LIU

et al., 2018) propuseram um método não supervisionado para selecionar assinatura gênica, sem executar uma etapa prévia de classificação.

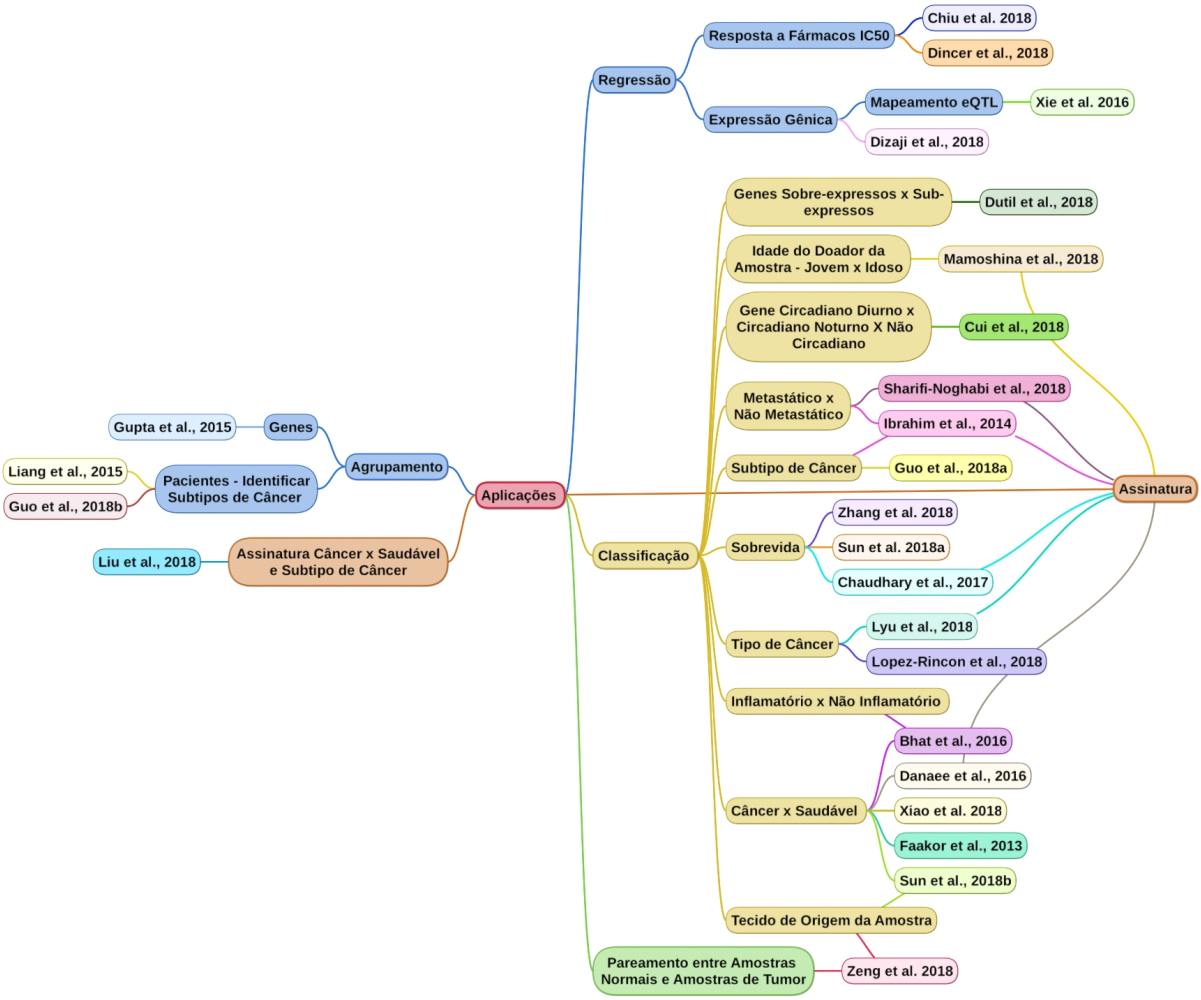

Figura 10 – Aplicações tratadas pelos trabalhos selecionados.

No restante desta seção, os 25 artigos selecionados são comentados em maiores detalhes, ordenados em ordem cronológica, seguida da alfabética.

(FAKOOR et al., 2013) propuseram uma abordagem que emprega *Deep Autoencoder Network* para a aprendizagem de *features*. A *Deep Autoencoder Network* permite realizar uma redução dimensional e, ao mesmo tempo, aprender *features* representativos dos dados de entrada de forma não supervisionada.

Os *features* aprendidos foram, então, utilizados em um classificador *Softmax Regression*, para a predição se amostras de perfis de expressão gênica correspondiam a células cancerosas ou saudáveis.

Os autores aproveitaram o potencial de generalização das *Deep Autoencoder Networks* para empregar dados de diferentes tipos de tumores (diferentemente de

prévios trabalhos que não integravam dados de diferentes tipos de tumores).

(IBRAHIM et al., 2014) propuseram uma abordagem para *feature selection* multinível, denominada de MLFS (*Multi-Level Feature Selection*). Essa abordagem foi aplicada para selecionar genes relevantes, em tarefas de classificação binária (subtipos de tumor e metastático versus não metastático), a partir de dados de expressão gênica. A MLFS emprega *Deep Belief Networks* (DBN) e uma extensão não supervisionada de *active learning*.

Primeiramente, a DBN foi treinada de forma não-supervisionada. O modelo obtido foi empregado para obter uma representação de mais alto nível dos dados de expressão gênica. Então, foram aplicados métodos de *feature selection* convencionais - testes estatísticos *t-test* e *relief-f* - para a seleção de genes relevantes na formação dessa representação de mais alto nível.

Após, os autores empregaram um esquema, nomeado de *unsupervised active learning*, para tentar reduzir o número de genes relevantes selecionados na etapa anterior. Nesse esquema, a acurácia de um classificador SVM (em uma tarefa de classificação binária - subtipos de tumor e metastático versus não metastático) foi utilizada para avaliar a *performance* de um subconjunto de genes selecionados.

Por fim, o grupo de genes selecionados foram usados em um classificador *Random Forest* para obter a *performance* final da metodologia. Também, foi apresentado um esquema para usar a abordagem proposta para *feature selection* de microRNAs, usando relações biológicas entre microRNAs e seus respectivos genes alvos.

(GUPTA; WANG; GANAPATHIRAJU, 2015) utilizaram um *Denoising Autoencoder* (DAE) para auxiliar o treinamento de métodos de *clustering* (k-means e algoritmo *Spectral Clustering*) para agrupar genes segundo os seus níveis de expressão. Ou seja, amostras re-geradas pelo DAE foram utilizadas como entrada dos métodos de *clustering*.

O objetivo dos autores foi demonstrar a eficácia do emprego de um DAE em tarefas de agrupamento, destacando sua capacidade de generalização de propriedades da distribuição das amostras de treinamento.

(LIANG et al., 2015) propuseram uma abordagem que emprega um modelo de *multimodal Deep Belief Network* (DBN) para agrupar pacientes, de forma a identificar subtipos de câncer, a partir da integração de dados de diferentes modalidades (expressão gênica, expressão de microRNA, metilação de DNA e dados clínicos).

Os autores destacaram, como diferencial de sua abordagem, o fato de que a multimodal DBN consegue explorar, tanto as propriedades estatísticas de uma mesma modalidade de dados (intra-modalidade), quanto as correlações entre as distintas mo-

dalidades - diferentemente da maioria dos métodos prévios.

O modelo multimodal DBN foi treinado, de forma não supervisionada, com o algoritmo Contrastive Divergence, que busca minimizar a Divergência de Kullback-leibler entre a distribuição dos dados de entrada e a distribuição gerada pelo modelo.

As primeiras camadas do modelo multimodal DBN constituem *Restricted Boltzmann Machines* (RBMs) separadas. Cada RBM toma, como entrada, uma modalidade de dados distinta. As camadas escondidas dessas RBMs aprendem uma representação latente sobre as características dos dados de uma mesma modalidade.

As últimas camadas do modelo fazem, então, a fusão das representações aprendidas pelas RBMs e obtêm uma representação conjunta dos dados. As variáveis escondidas são binárias e, portanto, cada combinação de configuração das variáveis da última camada (do topo) pode ser considerada como um grupo (*cluster*) distinto.

Os autores aplicaram a abordagem em dados de câncer de ovário e mama. Eles, também, realizaram algumas análises adicionais, como testes t para a comparação de médias entre dois grupos, de forma a verificar se a expressão de certos genes e microRNAs estavam diferentemente expressos entre os grupos (subtipos de cânceres) determinados pela multimodal DBN.

(BHAT; VISWANATH; LI, 2016) propuseram uma abordagem de *Deep Convolutional Generative Adversarial Network*, ou seja, um *framework GAN* no qual tanto a rede geradora quanto a discriminadora são *Convolutional Neural Networks (CNNs)*.

Essa abordagem foi denominada de DeepCancer e visa utilizar um *framework GAN* para a aprendizagem, a partir de dados de expressão gênica de um conjunto relativamente pequeno em número de amostras, das seguintes tarefas de classificação de: i) amostras de câncer de mama entre tecido inflamatório e não-inflamatório; e ii) amostras de mama e próstata entre tecido canceroso e saudável.

Nessa abordagem, a entrada da CNN discriminadora é uma matriz 2D, contendo os valores de expressão gênica de uma amostra. A entrada da CNN geradora, também, é uma matriz 2D de ruído desestruturado.

Os autores comparam os resultados alcançados pelo DeepCancer com os obtidos com o emprego de *Restricted Boltzmann Machine* seguido de classificadores tradicionais (*Suppor Vector Machine* - SVM e regressão logística) e demonstraram que a metodologia proposta proporciona melhores resultados.

(DANAEE; GHAEINI; HENDRIX, 2016) apresentaram uma abordagem de DL para detecção de câncer em amostras de RNA-Seq de mama (classificação câncer ou saudável), obtidas do repositório TCGA (TCGA, 2019); e identificação de genes relevantes - possíveis biomarcadores de câncer de mama.

A abordagem proposta emprega, primeiramente um *Stacked Denoising Autoenco-*

der (SDAE) para extrair *features* funcionais e obter uma representação dos dados de menor dimensionalidade. Após, a *performance* dessa representação é avaliada com o emprego de classificadores tradicionais (Redes Neurais Artificiais Rasas - com poucas camadas e SVM) que preveem o rótulo de cada amostra (câncer ou saudável) a partir da representação extraída.

Além disso, os autores identificaram um conjunto de genes relevantes, pela análise das matrizes de conectividade (pesos) do SDAE - cálculo da influência de cada gene na determinação da representação extraída pelo SDAE. Após, realizaram *pathway analysis* e enriquecimento em Gene Ontology (GO) dos genes selecionados (fortemente conectados no SDAE).

(XIE et al., 2016) com o intuito de contribuir para a obtenção de melhores mapeamentos *Expression quantitative trait locus* (eQTL), propuseram a aprendizagem de um modelo preditivo capaz de inferir a expressão genética a partir de dados de variação genotípica. O mapeamento eQTL é importante no estudo da influência de variações genéticas na expressão de genes.

Os autores realizaram as predições a partir de dados de variação genotípica de *Single Nucleotide Polymorphisms* (SNPs), mas comentaram que a abordagem é flexível e que dados epigenéticos, metabólicos e de fatores ambientais, também, poderiam ser usados, como entrada, do modelo preditivo.

A abordagem, proposta por (XIE et al., 2016), emprega *Stacked Denoising Autoencoder* para a aprendizagem de *features* úteis para o treinamento de um modelo de regressão. A saída desse modelo é usada, como entrada, para uma rede neural *Multilayer Perceptron* que prediz a expressão gênica de amostras de dados de variação genotípica.

(CHAUDHARY et al., 2017) treinaram um modelo de DL, um autoencoder, para analisar subgrupos de pacientes com carcinoma hepatocelular, de forma a diferenciar subpopulações de pacientes de acordo com a sobrevida.

Os autores empregaram dados multi-ômicos - RNA-seq, miRNA-seq e metilação - de seis coortes. Os dados de cada paciente foram agrupados em uma única matriz, para ser utilizada como uma única amostra nas análises.

As amostras foram, então, utilizadas para o treinamento de um autoencoder. Os *features* da camada de gargalo desse autoencoder mais ativos foram passados, como entrada, para um método K-means para agrupar (*clustering*) as amostras.

Os rótulos determinados pelo K-means foram, então, empregados para o treinamento de um classificador *Support Vector Machine* (SVM); esse classificador foi utilizado para a avaliação do método em um conjunto de validação. Os *top* N *features* mais correlacionados aos rótulos dos subgrupos (*clusters* obtidos K-means)

foram identificados usando ANOVA F-value. Os autores, também, fizeram análise de expressão gênica diferencial e enriquecimento funcional.

(CHIU et al., 2018) investigaram a predição de resposta de tumores a fármacos. Os autores relataram que recentemente foi publicado, por (IORIO et al., 2016), um estudo que avaliava a resposta de linhagens celulares de canceres humanos a uma ampla quantidade de fármacos anti-câncer. Entretanto, os autores, também, destacaram que, devido às diferenças essenciais entre as linhagens celulares e amostras de tumores, a predição de resposta a fármacos em tumores ainda é desafiadora.

Assim, no sentido de contribuir com a predição de resposta de tumores a fármacos, os autores propuseram uma abordagem de DNN, que emprega um pré-treinamento de autoencoders com uma boa quantidade de amostras não rotuladas (perfis de mutação e expressão gênica de células cancerígenas), obtidas do repositório TCGA (TCGA, 2019), para a aprendizagem de *features*. Após, utiliza os dados produzidos por (IORIO et al., 2016) (rotulados), para treinamento de uma rede neural preditiva de resposta a fármacos.

O modelo proposto contém três sub-redes neurais:

- um autoencoder para a aprendizagem de *features* de dados de mutação - pré-treinado em um grande conjunto de dados de câncer (obtidos do TCGA (TCGA, 2019)) para resumir representações centrais de dados de mutação de alta dimensionalidade;
- um autoencoder para a aprendizagem de *features* de dados de expressão gênica - também, pré-treinado em dados do TCGA (TCGA, 2019); e
- uma *Deep Feedforward Neural Network* para a predição de resposta a fármacos - que recebe, como entrada, as saídas da camada de gargalo das duas primeiras sub-redes. Dado um par de perfis de mutação e expressão gênica, o modelo é capaz de prever os valores de *half maximal inhibitory concentration* (IC_{50}) de 265 fármacos.

Segundo os autores, a abordagem proposta superou métodos no estado da arte.

(CUI et al., 2018) propuseram uma metodologia para a predição de genes circadianos, a partir de dados de séries temporais de expressão gênica, obtidos de amostras de sangue de um conjunto de indivíduos.

A abordagem proposta emprega uma *Deep Feedforward Neural Network* treinada para discriminar genes em aperiódicos e dois tipos distintos de genes circadianos (genes com maior expressão no turno diurno e genes com maior expressão durante a noite, após a liberação da melatonina). Genes circadianos são expressos, periodicamente, em um período de aproximadamente 24 horas e desempenham papéis

importantes de regulação.

Foram analisados dados de expressão de 18541 genes de 24 indivíduos, tomados em intervalos de três horas, durante um período de 34.5 horas. Primeiramente, os dados foram preprocessados, sendo transformados em dados categóricos: cada medida de expressão de cada gene foi transformada em três categorias $(-1, 0, 1)$, para indicar a tendência dos níveis de expressão: expressão diminuindo, estável ou aumentando.

Após, para estabelecer um *dataset* confiável para o treinamento da rede neural, os autores selecionaram 2000 genes e verificaram manualmente se cada um deles exibia comportamento periódico com período de 24 horas (genes circadianos). Após, os autores realizaram agrupamento dos genes identificados manualmente como circadianos, pelo método de *clustering* hierárquico de Ward, e separaram os genes em dois *clusters* - um associado aos genes de maior expressão noturna e outro, aos genes de maior de expressão diurna.

Então, o *dataset* rotulado obtido foi usado para o treinamento da rede neural. Cada amostra de entrada da rede neural é constituída pelos dados da série temporal de expressão de um gene específico de todos os 24 indivíduos e a saída é a classe desse gene.

Tendo o modelo treinado, os autores o usaram para estabelecer as classes do restante dos genes ainda não analisados e encontraram novos genes circadianos (ainda não identificados por métodos prévios). Os autores analisaram manualmente a série temporal de alguns desses novos genes circadianos e perceberam que a maioria deve ser de verdadeiros circadianos.

Os autores comparam a abordagem proposta com quatro classificadores tradicionais (k-NN, regressão logistica, *naive* Bayes e *Support Vector Machines*) e com os resultados de dois trabalhos anteriores de identificação de genes circadianos. Além disso, realizaram enriquecimento funcional dos genes circadianos encontrados.

(DINCER et al., 2018) propuseram o DeepProfile *framework* que: emprega um *Variational Autoencoder* (VAE), para aprender a codificar uma representação de menor dimensionalidade de dados de expressão gênica, usando uma grande quantidade de amostras não rotuladas de pacientes com leucemia mieloide aguda, obtidas do GEO (GEO, 2019); e, após, utiliza a representação aprendida para o treinamento de um modelo de predição de resposta a quimioterápicos.

Os autores empregaram dados de duas plataformas distintas. Para integrar os dados das plataformas, os autores mantiveram apenas genes que foram medidos em ambas plataformas. Os autores, também, normalizaram os dados, de forma que os níveis de expressão de cada gene, em cada *dataset*, ficasse com média zero e variância unitária, para garantir que os diferentes *features* (níveis de expressão dos genes) ficassem sob a mesma escala.

Para diminuir a dimensionalidade das amostras, antes do treinamento do VAE, os autores empregam *agglomerative clustering*, para agrupar genes com níveis de expressão similares em 300 *clusters*. Os centroides de todos *clusters* encontrados (valores de média dos níveis de expressão dos genes de cada grupo) foram usados, como entrada, do VAE.

Após, um conjunto de dados rotulados, com os respectivos valores de resposta a fármacos, foi empregado para o treinamento de um modelo de regressão linear, para a predição de resposta a quimioterápicos de pacientes com leucemia mieloide aguda. Esse preditor empregou, como entrada, as saídas da camada de gargalo do VAE treinado anteriormente.

Os autores, também, demonstraram que a metodologia é generalizável para ser empregada em outras aplicações, apresentando, também, resultados da abordagem para a classificação de: invasão de tumor de pacientes com câncer de ovário (infiltrativo ou expansivo); e subtipos de câncer de mama.

(DIZAJI; WANG; HUANG, 2018) propuseram uma nova abordagem de DL que emprega frameworks GANs, nomeada de SemiGAN, para a aprendizagem semi-supervisionada (utilizando tanto dados não rotulados, quanto rotulados). O método proposto foi utilizado para a inferência de expressão gênica de um conjunto de genes alvos, a partir de níveis de expressão de uma menor quantidade de genes (genes de referência). Foram utilizados dados obtidos por microarray e RNA-Seq.

Os autores explicaram que, como existe correlação entre padrões de expressão gênica, é possível realizar a inferência de expressão de genes alvos, a partir de um pequeno conjunto de genes de referência. Essa inferência pode auxiliar análises genéticas, uma vez que medir a expressão de poucos genes é menos custosa do que realizar medições para o transcriptôma completo.

A SemiGAN treina duas redes neurais geradoras: i) uma primeira G_x que gera amostras *fake* de expressão de genes de referência; ii) e uma segunda G_y que gera amostras *fake* de expressão de genes alvos. Ambas redes geradoras recebem um mesmo vetor de ruídos aleatórios como entrada, ou seja, usam uma entrada compartilhada.

Além disso, a SemiGAN possui três redes discriminadoras: i) a primeira D_x separa amostras *fake* das verdadeiras (vindas do *dataset* de treinamento) de dados de genes de referência; ii) a segunda D_y separa amostras *fake* das verdadeiras de dados de genes alvo; e iii) a terceira D_{xy} separa amostras *fake* das verdadeiras de dados de expressão de genes alvos, em conjunto com os correspondentes valores de expressão dos genes de referência (gerados a partir da mesma entrada compartilhada).

A SemiGAN, também, possui uma rede *Deep Feedforward Network* para a inferência da expressão dos genes alvos y, a partir dos genes de referência x (aprende

o mapeamento $F : x \rightarrow y$), que é treinada tanto com amostras *fake* geradas pelas GANs, quanto amostras rotuladas de um *dataset* de treinamento.

O treinamento da rede de inferência e das GANs é feito de forma cooperativa. Os autores, também, estabeleceram duas funções de *loss* - *reconstruction loss* e *translation loss* - para as redes geradoras das GANs, que visam evitar o problema de *Mode Collapse*.

Para tanto, os autores utilizaram uma rede neural I treinada para realizar o mapeamento inverso: receber uma amostra de valores de expressão de genes de referência x e gerar o vetor de ruídos aleatórios correspondentes (que ao ser aplicado nas redes geradoras obteria a amostra x de expressão de genes de referência e os respectivos níveis de expressão de genes alvo y).

A *reconstruction loss* é empregada para amostras não rotuladas e mede a diferença entre: uma amostra verdadeira x de genes referência; e uma amostra *fake*, gerada a partir do vetor de ruídos aleatórios, obtido pela rede de mapeamento inverso, quando essa recebe, como entrada, a amostra verdadeira x.

A *translation loss* é empregada para amostras rotuladas e mede a diferença entre uma amostra verdadeira y de genes alvos e uma amostra *fake*, gerada a partir do vetor de ruídos aleatórios, obtido pela rede de mapeamento inverso, quando essa recebe, como entrada, a amostra verdadeira x de genes de referência correspondente à amostra y.

Essa abordagem permite aproveitar dados não rotulados. As amostras não rotuladas (que possuem apenas valores para os genes de referência) são usadas no treinamento da: i) rede discriminadora D_x, que gera amostras *fake* de expressão de genes de referência; ii) rede discriminadora D_{xy}, que analisa a expressão tanto de genes de referência quanto de genes alvos - para obter uma estimativa dos valores de expressão para os genes alvos, a rede neural de inferência F é empregada; iii) no treinamento da rede neural de mapeamento inverso I.

Os autores, também, verificaram o papel de cada gene de referência na predição de expressão dos genes alvos, usando *Layer-wise Relevance Propagation* (LRP) (BACH et al., 2015). Os autores, também, empregaram k-means para agrupar genes e perfis transcriptômicos e analisar a contribuição de predição entre os *clusters*.

(DUTIL et al., 2018) destacaram os desafios para se aplicar *Deep Learning* na análise de dados de expressão gênica. Os autores utilizaram *Graph Convolutional Neural Networks*, para aproveitar informações de redes de interação (Interação Proteína-Proteína (PPI), interação gênica ou de vias metabólicas) e analisar dados de expressão gênica de RNA-Seq.

A rede neural foi treinada para predizer se determinado gene está sobre ou superexpresso (over or under-expressed) (em relação à média de valores de expressão

desse gene em todas as amostras do conjunto de treinamento), a partir da informação de expressão de genes vizinhos no grafo de relações entre genes.

Os experimentos realizados demonstraram dificuldades de analisar uma vizinhança de maior grau (que considere não apenas os vizinhos diretos), possivelmente, devido à falta de consideração pelo método de força, tipo e direção das interações (arestas no grafo).

(GUO et al., 2018) propuseram um novo método, denominado de *Boosting Cascate Deep Forest (BCDForest)* para a classificação de subtipos de câncer, a partir de dados de expressão gênica. O BCDForest foi baseado em um modelo de *Deep Forest* recentemente proposto - o gcForest.

O gcForest emprega uma cascata (várias camadas) de *ensembles* de florestas de árvores de decisão. Cada camada do modelo recebe *features* gerados pela camada inferior (estimativas de classificação dos ensembles da camada inferior) concatenados com o conjunto de *features* dos dados originais. Espera-se que a última camada do modelo consiga aprender uma estrutura mais complexa dos dados e fornecer predições com maior acurácia. O gcForest, também, possui um esquema de janela deslizante (*grained-scanning*) para que a aprendizagem de *features* seja feita analisando o contexto local dos dados.

O BCDForest adapta o gcForest, visando favorecer a generalização do modelo, de forma que seja possível uma melhor análise de *datasets* não balanceados e com poucas amostras. O BCDForest envolve três diferenciais, listados a seguir.

- É proposto um método, denominado de *multi-class-grained-scanning*, para encorajar a diversidade nos *ensembles* de florestas aleatórias e, consecutivamente, favorecer a generalização do modelo. Nesse método, a diversidade é encorajada pela utilização de classes de treinamento distintas para as várias florestas - ao invés de todas as florestas tratarem uma mesma tarefa de classificação de múltiplas classes, transforma se essa tarefa em múltiplas tarefas de classificação binárias (uma para cada classe) e cada floresta aprende a classificar uma classe específica.
- É utilizado um esquema, para estimar e considerar a acurácia dos diferentes *ensembles* do modelo, na determinação da predição final das amostras.
- É proposta a estratégia *Boosting Cascate*, para impulsionar *features* mais importantes na cascata de florestas, propagando os filtros mais discriminativos para as camadas superiores. A importância de cada *feature* é estimada, analisando os nodos das árvores de decisão.

Os autores avaliaram o BCDForest em um conjunto de *datasets* de dados obtidos por microarray e RNA-Seq, os resultados obtidos demonstraram que o BCDForest

supera o gcForest e quatro outros classificadores convencionais.

(GUO; SHANG; LI, 2018) propuseram um framework hierárquico de aprendizagem não supervisionada, nomeado de HI-SAE, que emprega *Stacked autoencoders* (SAE), autoencoder tradicional e k-means para: integrar dados transcriptômicos de diferentes tipos (dados de expressão gênica e de *splicing* alternativo); e agrupar amostras em subtipos de câncer.

Primeiramente, os autores estimaram os níveis de eventos de *skipping* de diferentes exons (pular o exon na formação do mRNA, durante o processo de *splicing*), a partir das leituras de RNA-Seq, analisando o número de ocorrências de junções de exons.

Após, os autores empregam dois SAEs, cada um sendo treinado separadamente para aprender uma representação latente de cada um dos dois tipos de dados (níveis de eventos de *skipping* exons e expressão gênica). Então, um autoencoder tradicional foi usado para integrar as representações.

Por último, o método de agrupamento K-means foi usado para separar as amostras em subtipos de câncer. Foram analisados dados de câncer de mama e empregou-se o número de grupos k, definido a priori, como o mesmo número de subtipos de câncer de mama determinado por (PRAT et al., 2015).

A *performance* do método foi avaliada pela verificação de significância da diferença de tempo de sobrevida (dados clínicos) entre os *clusters* encontrados pelo método. A análise de sobrevida revelou que os subtipos identificados pelo HI-SAE apresentavam diferenças mais significativas em tempo de sobrevida do que os subtipos identificados com o emprego de uma representação obtida por PCA e, também, de representações obtidas por cinco métodos de integração de dados convencionais.

Os autores, também, avaliaram a contribuição das informações de *splicing* alternativo na formação dos *clusters*, analisando as diferenças (entre *clusteres*) de níveis de eventos de saltos (*skipping*) de exons no *splicing* alternativo e de expressão de genes associados a fatores de *splicing* (papel importante na regulação do *splicing* alternativo); e concluíram que os dados de *splicing* forneceram informação discriminativa sobre os diferentes subtipos de câncer e que os fatores de *splicing* podem ser potenciais biomarcadores para separarem subtipos de câncer.

(LIU et al., 2018) propuseram uma metodologia não-supervisionada, que adota *Sample Learning* com *Deep Sparse Filtering*, para identificar genes característicos em câncer (ou seja, selecionar um pequeno conjunto de genes que participam de processos biológicos específicos de diferentes tipos de câncer), pela análise de dados de expressão gênica (microarray e RNA-Seq).

Os autores ressaltaram que a maioria dos métodos de seleção de genes realiza

feature learning, na qual o espaço de *features* original dos dados é transformado para uma representação, normalmente, de menor dimensionalidade. Porém, a *feature learning* tem uma desvantagem para esse tipo de aplicação: cada *feature* do espaço original dos dados é o nível de expressão de um gene específico e, portanto, ao se transformar o espaço de *features*, perde-se a correspondência de valores a cada gene específico - o quê dificulta a seleção de genes.

Considerando tal fato, os autores propuseram adaptar o método *Deep Sparse Filtering* (NGIAM et al., 2011) para usar *Sample Learning*, de forma que, após a aprendizagem realizada pelo método, ainda se possa associar um valor à cada gene específico. Ou seja, nesse tipo de aprendizagem, o espaço de amostras é transformado e o espaço de *features* (genes) é mantido.

O *Deep Sparse Filtering* é uma extensão do *Sparse Filtering* e emprega *Feedfoward Network* para realizar uma filtragem em múltiplas camadas e aprender estruturas mais complexas. O *Sparse Filtering* é um método não supervisionado que busca otimizar a esparsidade dos *features*, de forma a aprender *features* discriminativos (com alto potencial para distinguir as diferentes amostras).

Os autores executaram a abordagem proposta em dados de cinco tipos de câncer separadamente. Para cada tipo de câncer, as amostras analisadas envolviam subtipos e tecido normal. Portanto, os genes selecionados deviam ser aqueles que estavam diferentemente expressos entre tecido normal e canceroso e entre os subtipos do câncer considerados.

Após, os autores realizaram enriquecimento funcional, usando ToppFun, para encontrar funções biológicas que os genes selecionados tinham em comum. O método proposto foi comparado com quatro outros.

(LOPEZ-RINCON et al., 2018) investigaram o uso de *Convolutional Neural Networks* (CNN) (com convoluções 1D) para a classificação de 29 tipos de tumores, a partir de dados de expressão de microRNAs, obtidos do repositório TCGA (TCGA, 2019); visando contribuir para o desenvolvimento de metodologias de determinação de assinaturas de tipos específicos de tumores, para facilitar o diagnóstico de subtipos tumorais e determinação de tecido de origem.

Os autores empregaram algoritmos evolucionários para otimizar os hiperparâmetros da CNN (número de canais de saída de camadas convolucionais, largura de janela das camadas convolucionais e largura de janela das camadas de pooling).

Os autores, também, compararam a abordagem proposta com 21 classificadores no estado da arte e mostraram que o método proposto permitiu obter melhores resultados.

(LYU; HAQUE, 2018) propuseram uma abordagem que emprega uma *Convoluti-*

onal Neural Network (CNN) (com três camadas convolucionais e três camadas *fully connected*) para a classificação de amostras de 33 tipos de tumores, a partir de dados de expressão gênica de RNA-Seq, obtidos do TCGA (TCGA, 2019).

Primeiramente, foi feita uma pré-filtragem: genes que apresentavam pouca variação entre todas as amostras foram desconsiderados nas análises. Após, cada amostra foi organizada como uma matriz 2D (como as imagens tradicionalmente usadas nas CNNs). Essas matrizes foram, então, usadas para treinar uma CNN para predizer a classe de uma amostra (tipo de tumor entre os 33 analisados).

Após, os autores utilizaram o método Guided Grad-Cam, que busca analisar os gradientes da CNN, para determinar regiões da amostra de entrada que foram mais relevantes para a obtenção da classificação retornada pela CNN. O Guided Grad-Cam foi empregado para obter um *heatmap* para cada amostra. Por fim, os autores realizaram enriquecimento funcional dos top-genes nos *heatmaps*. Os autores comentaram que os top-genes podem ser possíveis biomarcadores de tipos específicos de tumores.

(MAMOSHINA et al., 2018) empregaram um conjunto de métodos de Aprendizagem de Máquina, incluindo *Deep Feedforward Neural Networks*, para a predição de idade de amostras de tecido muscular esquelético humano, a partir de dados de expressão gênica de indivíduos saudáveis, obtidos do GEO (GEO, 2019) e ArrayExpress (ARRAYEXPRESS, 2019). A *performance* foi avaliada com amostras obtidas do GTEx (GTEX, 2019). Os autores ressaltaram que biomarcadores de idade podem ser empregados para identificar novos alvos moleculares para o desenvolvimento de terapias anti-idade.

Os autores obtiveram acurácia de 80% na predição do grupo de idade (jovem – de 16 a 30 anos e idoso – acima de 60 anos). Os autores, também, realizaram análises de expressão gênica diferencial e *pathway analysis* para comparar as assinaturas genéticas entre os grupos (jovem $\times$ idoso).

Os autores empregaram *Deep Feature Selection* (DFS) – uma *Deep Feedforward Neural Networks*, proposta por (LI; CHEN; WASSERMAN, 2016), que apresenta uma camada de neurônios com funções de ativação linear um-para-um (cada neurônio é ligado apenas a um neurônio da próxima camada), logo após a camada de entrada da rede neural. Os pesos dessa camada linear podem ser interpretados como a importância de cada atributo de entrada na classificação realizada pela DFS.

Os autores usaram o algoritmo iPANDA para comparar as assinaturas genéticas dos grupos. Os métodos de Machine Learning usados nas predições de idade foram: *ElasticNet*, *Support Vector Machines* (SVM), *k-Nearest Neighbors*, *Random Forests* e *Feed-forward Neural Networks*.

Para a identificação de genes relevantes na classificação, os autores utilizaram

quatro abordagens: i) realizaram um *rank* dos genes de acordo com o valor absoluto dos coeficientes de regressão do modelo ElasticNet; ii) empregaram o algoritmo *Random Forest Feature Importance*; iii) utilizaram os valores de importância relativa de cada gene designado pelo modelo do DFS; iv) utilizaram o método de *wrapper feature selection Permutation Feature Importance* (PFI) (PUTIN et al., 2016) em conjunto com o classificador SVM. Como todas essas quatro abordagens para a seleção de atributos retornam valores de importância diferentes, os autores empregaram o algoritmo *Borda count* para obter um valor de importância final de cada gene.

(SHARIFI-NOGHABI et al., 2018) propuseram o *Deep Genomic Signature (DGS)*: uma abordagem para investigar assinaturas, em dados de expressão gênica, que sirvam para detectar precocemente metástase em cânceres de próstata.

Essa abordagem visa permitir o aproveitamento de dados de coortes que não possuem informações sobre os desfechos clínicos, ou seja, o DGS emprega tanto dados rotulados com poucas amostras, quanto dados não-rotulados com uma maior quantidade de amostras.

O DGS utiliza *Denoising Autoencoders* (DAEs) para a extração e seleção de *features* em dados de expressão gênica. São empregados dois DAEs treinados separadamente. Primeiramente, é treinado um DAE que emprega as amostras não-rotuladas. Após, os parâmetros da primeira *hidden layer* desse DAE são transferidos para um segundo DAE, esses parâmetros ficam congelados e o restante da rede neural é treinada com o conjunto de amostras rotuladas, ou seja, é empregado *transfer learning* entre os dois DAEs.

Após, o conjunto de pesos que ligam cada neurônio de entrada (correspondente a cada gene) aos neurônios da camada de gargalo do segundo DAE (treinado com dados rotulados) são analisados, para a seleção de um conjunto de genes a serem usados no treinamento de um modelo de regressão logística, para a predição de metástase: calcula-se uma distribuição de pesos considerando todos os genes, então, determinam-se genes cujo respectivo vetor de pesos esteja nas caldas da distribuição de pesos.

Os genes selecionados são, então, usados para o treinamento do modelo de regressão logística. Os genes com coeficientes desse modelo não nulos são considerados como parte da assinatura genética para detectar metástase.

(SUN; WANG; LI, 2018) investigaram o emprego de *Deep feedforward neural networks* com quatro camadas escondidas, para a classificação de prognóstico em câncer de mama em duas classes: sobrevida menor de 5 anos e sobrevida maior de 5 anos.

O método realiza a predição a partir de dados multi-dimensionais: dados clíni-

cos, de expressão gênica e *Copy Number Alterations* – CNAs (para medir duplicações ou deleções de genes). Primeiramente, os dados passam por uma etapa de *feature selection* que emprega o método mRMR, visando mitigar o problema de '*curse of dimensionality*' (número de amostras disponíveis bem menor do que a dimensionalidade dos dados - número de atributos por amostra).

Após, os dados passam por três *Deep feedforward neural networks*, uma para cada tipo de dado (clínicos, expressão genética e CNAs), que são treinadas individualmente. A saída dessas redes neurais são combinadas por um média ponderada, com pesos sendo estabelecidos para que se maximize a *performance* do método em um conjunto de validação.

Os resultados obtidos pela abordagem proposta foram melhores que os obtidos por métodos tradicionais (*Support Vector Machines* - SVM, Random Forest e Regressão Logística).

(SUN et al., 2018) desenvolveram o GeneCT (*Generalizable Cancerous-status and Tissue-of-origin classifier*), uma abordagem para classificar determinada amostra de expressão gênica de RNA-Seq como cancerosa ou não e o tipo de tecido de origem das amostras. GeneCT possui duas redes neurais profundas (uma para classificar o status de existência de câncer e outra, o tipo de tecido).

(XIAO et al., 2018) propuseram um *ensemble* de classificadores para a predição se determinada amostra de RNA-Seq é derivada de um tecido canceroso ou saudável. As análises foram feitas em três tipos de tumores.

Na abordagem proposta, primeiramente, se realiza uma etapa de *feature selection* de seleção de genes diferentemente expressos entre as duas classes (câncer $\times$ saudável), visando a diminuição da dimensionalidade dos dados a serem analisados.

Após, emprega-se uma primeira etapa de classificação usando técnicas tradicionais (*k-Nearest Neighbors (k-NN), Support Vector Machines - SVMs, Decision Trees, Random Forests* e *Gradient Boosting Decision Trees*). As predições dos cinco classificadores foram, então, utilizadas como entrada de uma rede neural *Deep feedforward* que realiza o *ensemble* dos classificadores.

Os resultados foram levemente melhores do que com o emprego de *Majority voting* para a combinação da saída dos classificadores.

(ZENG et al., 2018) explicaram que, para que comparações possam ser feitas entre cânceres e tecido normal, é interessante ter pares de amostras: amostra de câncer com correspondente amostra de tecido normal adjacente ao tumor e livre de células cancerosas. Os autores destacaram, também, a dificuldade de se obter, em repositórios públicos de dados de expressão gênica de cânceres (como o TCGA (TCGA, 2019)

e o TARGET (NCI, 2019)), amostras de tecido normal (saudável) correspondente às amostras cancerosas.

Os autores, também, notaram que o projeto recente *Genotype-Tissue Expression* (GTEx) (GTEX, 2019) disponibiliza um conjunto de amostras de indivíduos saudáveis. (ZENG et al., 2018), então, analisaram a viabilidade de se empregar essas amostras do GTEx para a comparação com amostras cancerosas obtidas de outros repositórios (de outro grupo de indivíduos).

Para tornar essa tarefa possível, (ZENG et al., 2018) propuseram uma abordagem que: i) emprega autoencoders para aprender uma representação latente dos dados de expressão gênica; ii) e, após, usa correlações, na representação aprendida, entre as amostras normais e as amostras de câncer, para determinar um conjunto de amostras de tecido normal (saudável), vindas do GTEx, adequadas para serem empregadas como referência (comparação) das amostras de câncer.

Para a avaliação do método, os autores compararam: as assinaturas genéticas de cânceres, determinadas pela análise de genes diferentemente expressos entre pares de amostras de câncer e de tecido normal, selecionadas pela abordagem proposta; em relação às assinaturas genéticas determinadas com o emprego de um pequeno número de amostras do TCGA que apresentavam amostras correspondentes de tecido normal subadjacente ao tumor.

(ZENG et al., 2018) concluíram que amostras do GTEx podem ser usadas como referência para a análise de amostras cancerosas, especialmente quando não se possui amostras de tecido normal adjacente ao tumor.

(ZHANG et al., 2018) apresentaram um framework não-supervisionado, que integra *Principal Component Analysis* (PCA) e autoencoders, para a aprendizagem de *features* em dados de expressão gênica obtidos por microarray.

Um *ensemble* de classificadores baseado no algoritmo AdaBoost foi, então, empregado para predizer desfechos clínicos em dados de câncer de mama, a partir dos *features* aprendidos.

Os resultados obtidos com a abordagem proposta foram comparados com uma metodologia semelhante, mas sem o emprego dos autoencoders; (ZHANG et al., 2018) perceberam que o emprego dos autoencoders proporcionam melhores resultados.

3.3 Considerações sobre os Trabalhos Relacionados

Muitos dos trabalhos relacionados (FAKOOR et al., 2013; GUPTA; WANG; GANAPATHIRAJU, 2015; DANAEE; GHAEINI; HENDRIX, 2016; CHAUDHARY et al., 2017; DINCER et al., 2018; GUO; SHANG; LI, 2018; SHARIFI-NOGHABI et al., 2018; ZHANG et al., 2018; IBRAHIM et al., 2014; BHAT; VISWANATH; LI, 2016) que reali-

zaram uma tarefa de classificação ou agrupamento de amostras empregaram métodos de *Deep Learning* apenas em uma etapa inicial, de forma não supervisionada, para a aprendizagem de uma representação embutida de menor dimensionalidade, utilizando *Deep Autoencoders*, *Generative Adversarial Networks* ou *Deep Belief Networks*. E, após, utilizaram essa representação como entrada de métodos de Aprendizagem Rasa.

Embora essa etapa inicial possa favorecer a generalização de classificadores (aplicados posteriormente), isso dificulta a interpretabilidade dos modelos e, consequentemente, a seleção de genes relevantes; pois cada *feature* da representação embutida não estará mais associada a um único ou a poucos genes.

Dessa forma, no presente trabalho, escolhemos por não empregar essa abordagem de utilização de representações embutidas como entrada de classificadores. Nossa proposta, neste trabalho, é aplicar os dados de expressão gênica diretamente em classificadores, explorando a característica ressaltada na literatura de que métodos de *Deep Learning* não requerem engenharia de *features*, podendo extraí-los automaticamente de dados brutos (MIN; LEE; YOON, 2016).

(LIANG et al., 2015) não realizaram tarefas de classificação; os autores empregaram as *Deep Belief Networks* para realizar agrupamento de amostras. (XIAO et al., 2018) utilizaram uma rede neural profunda *Deep feedforward* apenas para combinar a predição obtida por cinco classificadores de Aprendizagem Rasa.

Mais próximos ao presente trabalho têm-se: (SUN et al., 2018), (SUN; WANG; LI, 2018), (GUO et al., 2018), (LYU; HAQUE, 2018), (MAMOSHINA et al., 2018) e (LOPEZ-RINCON et al., 2018).

(SUN et al., 2018) utilizou redes neurais *Deep feedfoward* para separar amostras normais x cancerosas e tipo de tecido das amostras. (SUN; WANG; LI, 2018) empregou rede neural *Deep feedforward* para predição de classe de sobrevida (acima ou abaixo de 5 anos). (GUO et al., 2018) utilizou uma abordagem de florestas aleatórias em muitas camadas - designada, portanto, como método Profundo - para classificação de subtipos tumorais. (LYU; HAQUE, 2018) utilizaram uma *Deep Convolutional Neural Network* (CNN) 2D para classificação de tipo (tecido) de amostras tumorais.

(MAMOSHINA et al., 2018) utilizaram *Deep Feedfoward Networks* em conjunto com métodos de Aprendizagem Rasa. Dentre os trabalhos selecionados, esse é o único trabalho que realizou a combinação de diferentes modelos de Aprendizagem de Máquina para determinar a relevância de cada gene.

(LOPEZ-RINCON et al., 2018) utilizaram CNNs 1D para classificação de tipos tumorais e compararam com 21 classificadores. Segundo os autores, seus resultados foram melhores com a CNN do que com os demais métodos. Entretanto, os autores utilizaram uma abordagem evolucionária para a determinação dos hiperparâmetros da CNN e não comentam se verificaram diferentes hiperparâmetros dos demais métodos.

Nossos resultados, embora aplicados em outros *datasets*, se mostraram discordantes aos obtidos por (LOPEZ-RINCON et al., 2018). (LOPEZ-RINCON et al., 2018), por exemplo, reportou que o SVM obteve apenas 13.58% de acurácia contra 96.60% do método de CNN proposto pelos autores.

Esse resultado também está discordante ao do trabalho de (GRISCI et al., 2019) que analisou um conjunto de classificadores de Aprendizagem Rasa em dados de expressão gênica e obteve o SVM como o melhor classificador na maioria dos *datasets* experimentados.

Deve-se destacar, porém, que a separação de tipos tumorais de diferentes tecidos, como realizado por (LOPEZ-RINCON et al., 2018), é uma tarefa mais simples do que a classificação de subtipos tumorais como é avaliada no presente trabalho e em (GRISCI et al., 2019).

Não encontramos trabalhos que realizassem comparações mais criteriosas de métodos de classificação de Aprendizagem Rasa e Profunda com vários *datasets* de expressão gênica em problemas de diferentes dificuldades e treinamento de vários modelos por método. Além disso, dentre os trabalhos selecionados, apenas o trabalho de (MAMOSHINA et al., 2018) realizou uma combinação de modelos de classificadores de Aprendizagem de Máquina para extração dos *features* (genes) mais importantes.

4 METODOLOGIA

Neste capítulo é explicada a metodologia adotada para a investigação dos diferentes métodos de Aprendizagem de Máquina. São listados os datasets empregados. Então, são descritas, em detalhes gerais, as duas abordagens originais do trabalho: i) a utilização do **Transcriptograma** em conjunto com CNNs; e ii) a **ML-PEns**. Seguindo, é abordada a forma como foi feita para definir os melhores conjuntos de hiperparâmetros dos métodos. Após, comenta-se a forma como é realizada a comparação e excução dos métodos. Então, esquematizam-se as soluções adotadas para a seleção dos genes relevantes.

4.1 Esquema Geral

As Figuras 11 e 12 esquematizam a abordagem adotada para a investigação dos diferentes métodos de Aprendizagem de Máquina.

Na **primeira etapa** do trabalho são investigados os seguintes dois métodos.

- As Redes Neurais Profundas Convolucionais - Convolution Neural Networks (CNNs), experimentando alternativas de entrada e diferentes arquiteturas, otimizadores, regularização e hiperparâmetros.
- A Máquinas de Vetores de Suporte - Support Vector Machine (SVM) como método *baseline* para comparação com as acurácias obtidas com a CNN. O SVM foi escolhido por ter demonstrado maior acurácia entre os métodos de aprendizagem rasa experimentados no trabalho de (GRISCI et al., 2019).

Para esta etapa, adotou-se um único dataset de amostras de expressão gênica - o dataset com maior número de amostras ($n = 763$) que obtivemos - durante o decorrer deste texto, o tratamos como dataset piloto. O problema tratado nesse dataset é a separação de quatro subgrupos moleculares de meduloblastoma.

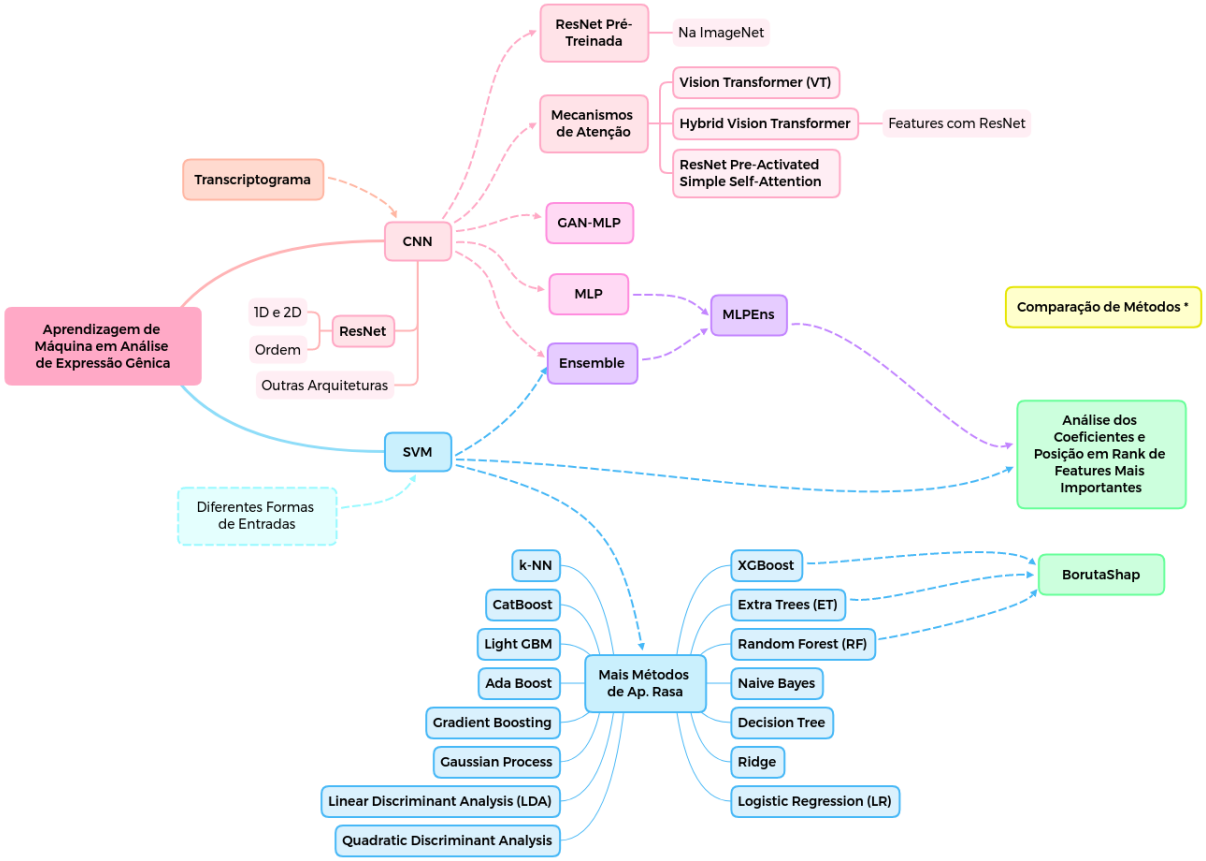

Figura 11 – Esquema da abordagem empregada.

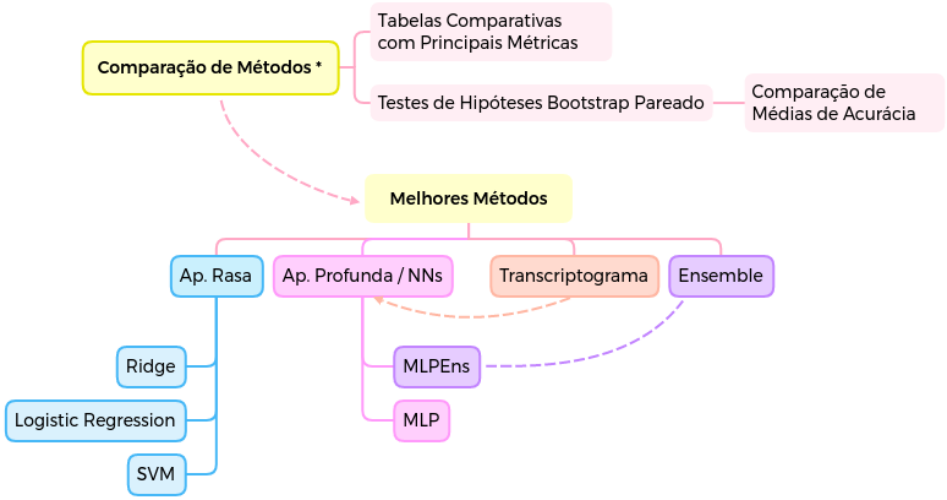

Figura 12 – Esquema da abordagem empregada - Comparação de Métodos de Aprendizagem de Máquina

Um enfoque maior é dado às *Convolution Neural Network Deep Learning* (CNN), especialmente no início do trabalho. Então, seguiu-se para alternativas que pareciam ser o caminho mais adequado no momento: a investigação de *ensemble* de modelos; o *framework Generative Adversarial Network* - GAN (GAN-MLP) (ZHANG et al., 2018); a MLP; metologias de Redes Neurais Profundas com mecanismo de atenção;

e a ResNet Pré-treinada na ImageNet. Como abordagens de Redes Neurais Profundas com Mecanismo de atenção, adotaram-se o *Vision Transformer* (DOSOVITSKIY et al., 2020), *Hybrid Vision Transformer* (DOSOVITSKIY et al., 2020) e a *ResNet Pre-Activated Simple Self-Attention* (ZHANG et al., 2019).

Então, propõe-se a arquitetura de redes neurais MLPEns que emprega o *dropout* para tratar um modelo de redes neurais como um *ensemble* de modelos. Também, experimenta-se um conjunto de classificadores de Aprendizagem Rasa.

Para a seleção de genes relevantes, analisam-se o método BorutaShap; os coeficientes do SVM; e os pesos da primeira camada da MLPEns.

Realizamos experimentos em diferentes datasets para comparação dos métodos estudados.

4.2 Datasets Empregados

A Figura 13 esquematiza os datasets empregados no trabalho; e a Figura 14 relaciona os experimentos realizados com os datasets. Nas subseções a seguir explicamos usos desses datasets.

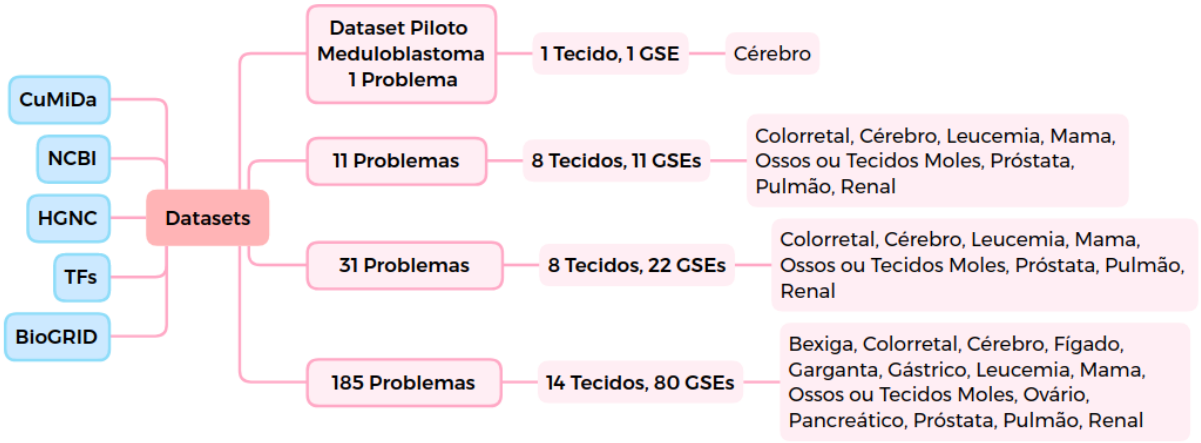

Figura 13 – Datasets empregados.

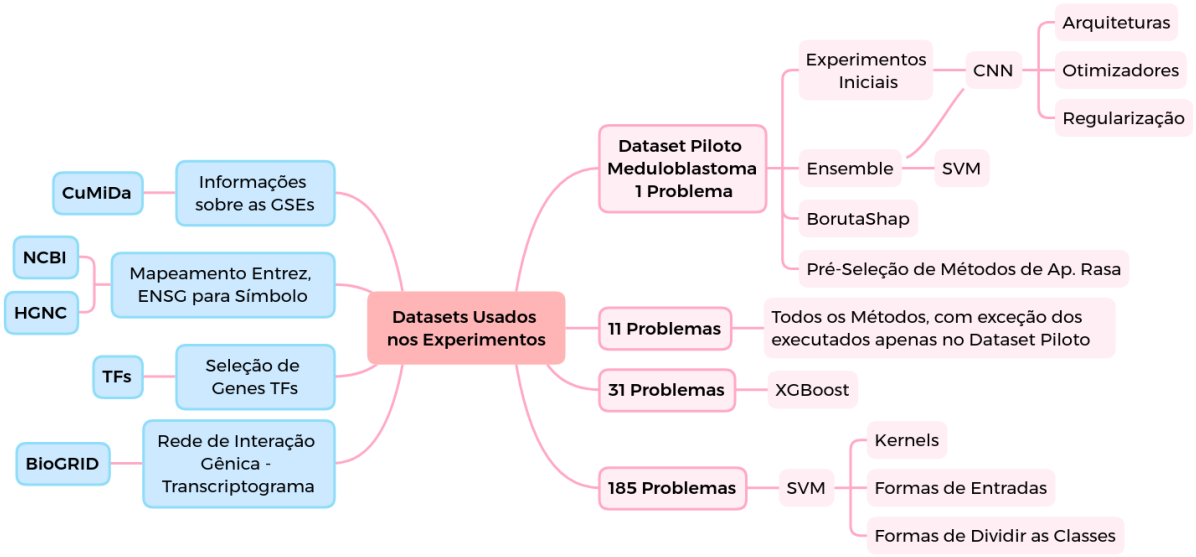

Figura 14 – Relação dos datasets e experimentos.

4.2.1 Amostras de Expressão Gênica em Cânceres

Para a maioria dos experimentos, utilizam-se 11 datasets de *microarray*, obtidos no GEO (GEO, 2019), com uma separação de classes por dataset, constituindo 11 problemas. Para o classificador SVM, realizaram-se mais experimentos empregando 80 GSEs e 185 problemas, com diferentes formas de dividir as classes para um mesmo dataset (por exemplo, uma primeira divisão pode ser os subtipos tumorais e uma segunda tumor x normal), de forma que um mesmo dataset é avaliado para problemas mais simples (tumor x normal) e mais complexos (divisão de subtipos tumorais). O XGBoost é avaliado em 22 GSEs e 31 problemas.

Todos os datasets (GSEs) e divisões de classes empregadas, bem como, os experimentos realizados são especificados nas tabelas do Apêndice B. A seguir, apresentamos apenas os 11 datasets empregados na maioria dos experimentos.

Tabela 1 – 11 Problemas Selecionados.

	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(1, 4)	GSE2553	Bone_or_soft_tissue	Ewing_Sarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 162 Ewing_Sarcoma: 19
(8, 2)	GSE85217	Brain	grupos moleculares	763	4	Group4: 326 SHH: 223 Group3: 144 WNT: 70
(13, 2)	GSE26910	Breast_prostate	normal_breast x cancer_breast	12	2	Normal_breast_stroma: 6 Stroma_associated_to_breast_invasive_tumor: 6
(18, 1)	GSE70947	Breast	tumor x normal	296	2	normal: 148 tumor: 148
(42, 1)	GSE28497	Leukemia	8 subtipos de ALL, normal	288	9	7_B-with-miscellaneous-karyotype: 59 3_ETV6_RUNX1: 56 1_Hyperdiploid: 52 8_T-ALL: 45 2_TCF3-PBX1: 22 4_MLL: 18 5_Ph: 16 6_Hypo: 16 9_CD10CD19: 4
(51, 4)	GSE6919_U95B	Prostate	tumor x normal	143	2	normal: 77 primary_prostate_tumor: 66
(59, 4)	GSE15824	Brain	glioblastoma (sem secundário, com cell line), astrocitoma, oligodendrioglioma, normal	42	4	glioblastoma: 22 astrocytoma: 8 oligodendrioglioma: 7 normal: 5
(62, 2)	GSE7670	Lung	adenocarcinoma x normal adj adenocarcinoma (sem cell line)	54	2	normal_adjacent_adenocarcinoma: 27 cancer_adenocarcinoma: 27
(71, 1)	GSE6344_U133B	Renal	tumor x normal	20	2	Normal: 10 Tumor: 10
(77, 5)	GSE77953	Colorectal	câncer x normal (removendo amostras de metástase)	47	2	cancer: 34 normal: 13
(82, 4)	GSE21510	Colorectal	câncer (juntando LCM e homogenizado) x normal homogenizado	148	2	cancer: 123 normal: 25

4.2.2 Auxiliares

Além dos datasets de amostras de expressão gênica, são utilizados os seguintes dataset: CuMiDa, NCBI, HGNC, TFs e BioGRID. O dataset **CuMiDa** (GRISCI et al., 2019)é empregado como um referencial para estabelecer os datasets a serem usados. Neste trabalho, usou-se todos os GSEs disponibilizados, com a adição de mais algumas de interesse do nosso grupo de pesquisa.

Tanto o **NCBI** (NCBI, 2019) quanto o **HGNC** (HGNC, 2019) são usados para obter informações sobre os genes e mapear as informações de identificadores (entrez e ensg) estabelecidos nas GPLs de cada GSE para símbolo, tarefa que facilita a visualização de questões relacionadas com a interpretabilidade dos modelos, ou seja, para que se possa obter uma lista de símbolos oficiais de cada gene selecionado como importante.

O dataset de **TFs** (LAMBERT et al., 2018) é utilizado para seleção apenas do conjunto de genes que são Fatores de Transcrição (TFs). Alguns experimentos com SVM utilizam apenas valores de expressão dos TFs para realizar as classificações.

Tal abordagem, possibilita a redução do número de features tratados pelo classificador e facilita a obtenção de interpretabilidade da resposta fornecida pelo classificador, uma vez que genes TFs normalmente possuem relativa alta quantidade de informação descrita na literatura; contrastando com alguns genes que quase não se possui informação disponível sobre a sua funcionalidade.

O dataset **BioGRID** (BIOGRID, 2019) estabelece uma rede de interação entre genes: um grafo, na qual os nodos são genes e cada aresta ligando dois genes informa que existe informação na literatura sobre esses dois genes apresenta relação. Esse grafo é utilizado pelo método Transcriptograma para a obtenção de uma lista ordenada de genes, na qual genes relacionados estão posicionados perto uns dos outros.

4.3 Propostas Originais

As duas principais propostas apresentadas neste trabalho são descritas a seguir.

O ordenamento do **Transcriptograma** (MORAIS; ALMEIDA; DALMOLIN, 2019) para estabelecer uma ordem dos *features* (genes) que permite montar amostras para serem utilizadas em redes neurais CNNs na qual exista uma relação de estrutural de posição que possa ser aproveitada por essas redes neurais nas execuções das convoluções. O Capítulo 2.8 apresenta em detalhes o método. Neste trabalho, para determinar a contribuição que este tipo de método pode trazer, investigamos modelos de ResNet com entrada de features expostos em ordem aleatória e em ordem dada pelo Transcriptograma, expostos na Seção 7.5.4.

A segunda contribuição é a proposição de uma metodologia para tratar **um modelo de rede neural como ensemble de modelos** e uma arquitetura para empregá-la, nomeada de **MLPEns**. Esta abordagem é apresentada no Capítulo 5. Para verificar o desempenho da abordagem, comparamos seus resultados com os obtidos por abordagens de arquiteturas de CNN ResNet e Multi-layer Perception MLP, apresentados na Seção 7.10.

4.4 Definição dos Melhores Conjuntos de Hiperparâmetros

4.4.1 Métodos de Redes Neurais

Realizar fine-tuning dos hiperparâmetros em cada dataset analisado demandaria alto custo de processamento (tempo) e também dificulta na comparação de métodos, pois teríamos um melhor conjunto de hiperparâmetros por dataset. Dessa forma, para a escolha do melhor conjunto de hiperparâmetros, realizamos o procedimento esquematizado na Figura 15, a seguir.

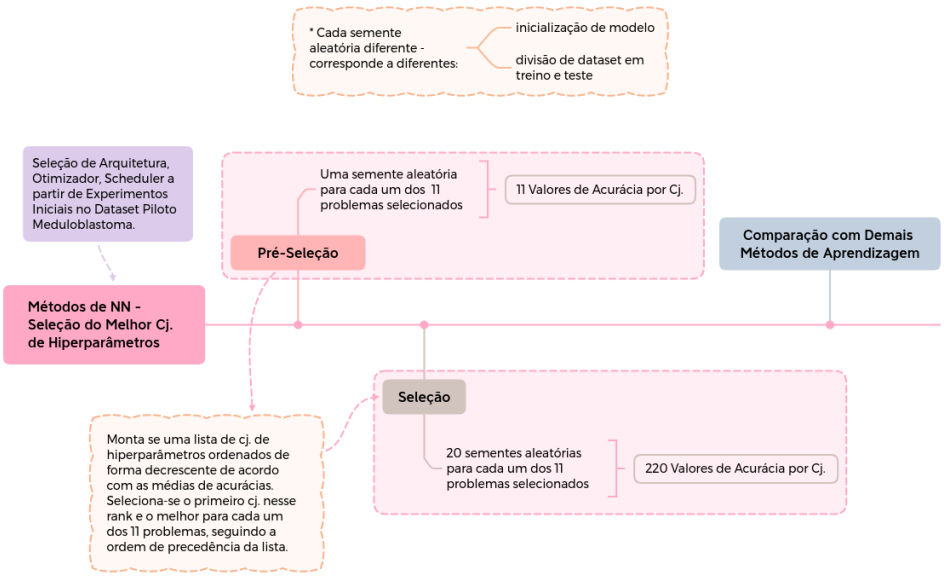

Figura 15 – Processo adotado para obtenção do melhor conjunto de hiperparâmetros por método de rede neural.

Primeiramente, realiza-se um treinamento de modelo (apenas uma semente aleatória) por problema e conjunto de hiperparâmetros investigados. Após, selecionam-se os melhores conjuntos de hiperparâmetros: o com maior mé- dia de acurácia entre todos problemas e aqueles selecionados como melhores em cada problema individual. Então, é executado treinamento de modelos com outras sementes aleatórias, completando 20 execuções por problema e alcançando $11 * 20 = 220$ execuções por conjunto de hiperparâmetros.

Então, usando essas 220 define-se o melhor conjunto de hiperparâmetros por método, que é empregado para comparação do método de Aprendizagem de Máquina em questão com os demais.

4.4.2 Métodos de Aprendizagem Rasa

Para os métodos de Aprendizagem Rasa, adotam-se, preferencialmente, os parâmetros *default* dos pacotes empregados.

Para as Máquinas de Vetor Suporte (SVM) investigou-se quatro tipos de kernels: Linear L1, Linear L2, RBF e Poly. Foram empregadas as implementações e demais hiperparâmetros *default* dos pacotes sklearn (módulos linearSVC para os kernels lineares e SVC para os demais kernels) e PyCaret.

Para o XGBoost, varia-se o hiperparâmetro max_depth que modela a máxima profundidade e alteram-se alguns hiperparâmetros para favorecer a generalidade como será descrito no Capítulo cap-resultados. Para a implementação, foram empregados o sklearn e o PyCaret.

Os demais métodos de aprendizagem Rasa são executados através do pacote

PyCaret, usando hiperparâmetros *default*.

4.5 Comparação dos Métodos de Aprendizagem de Máquina para Classificação

Após a obtenção do melhor conjunto de hiperparâmetros por método de Rede Neural usando uma semente aleatória, realizam-se mais treinamentos de modelos até alcançar 20 execuções (20 sementes aleatórias).

Para os métodos de Aprendizagem Rasa, para cada conjunto de hiperparâmetros investigado, também, obtêm-se 20 execuções (com 20 sementes aleatórias). Cada semente aleatória, define uma inicialização diferente dos modelos e uma divisão distinta dos datasets em conjuntos de treino e teste. Empregamos 70% das amostras para treino e 30% para teste, com divisão estratificada das diferentes classes.

Analisa-se a acurácia, então, desses conjuntos de execuções, tendo 20 execuções por cada um dos problemas analisados. As comparações envolvendo as Redes Neurais e as comparações de todos os métodos de classificação em conjunto são executadas em 11 problemas (em 11 GSEs distintas), tendo portanto $11 * 20 = 220$ valores de acurácia por método. Então, apresentam-se tabelas de principais métricas, considerando cada problema individualmente e em conjunto.

Para saber se as diferenças encontradas realmente são significativas, empregam-se testes de comparação de médias bootstrap pareado (KABACOFF, 2015; BRUCE; BRUCE, 2019). Uma vez que cada semente aleatória define uma divisão do dataset em treino e teste e, para cada divisão tem se um valor de acurácia por método; para comparar dois métodos temos 220 pares de acurácia, logo um teste pareado é o mais adequado. A vantagem dos métodos bootstrap é que não requer suposições sobre a distribuição dos dados.

4.6 Sistema de Execução dos Métodos

As redes neurais profundas foram executadas na nuvem no sistema Google Colaboratory (Colab) que possibilita a execução de notebooks jupyter em python - foram mantidas 6 contas profissionais (pro) nesse ambiente. Os arquivos de resultados dos modelos foram salvos no google drive e, após, baixados para a realização das análises.

As contas pro permitem acesso a GPUs mais modernas (normalmente são disponibilizadas a GPU T4 ou P100 com 16GB de memória) que as contas livres e, também, acesso em tempo integral às GPUs (sem pausas). Além disso, as contas pro permitem a utilização de um terminal de comandos que facilita a realização de conferências

dos arquivos e status dos processos em execução.

A fim de agilizar as execuções, abrimos um arquivo (notebook) por conta em um ambiente com a opção de alta RAM marcada (cerca de 25GB ficam a disponibilização do usuário) e empregamos o pacote *multiprocessing* do python para abrir vários processos concorrentes na mesma conta: cada processo rodando uma opção de hiperparâmetro e semente aleatória.

Dependendo do dataset empregado e do número de canais ou neurônios usados na rede neural profunda foi possível abrir de 2 a 6 processos concorrentes por conta - o número de processos abertos foi limitado pela quantidade de memória da GPU utilizada.

Os métodos de Aprendizagem Rasa, por sua vez, foram executados em máquina local sem grandes requisitos de hardware.

4.7 Interpretabilidade - Seleção de Genes Relevantes

A Figura 16 esquematiza as abordagens adotadas para seleção de genes relevantes, que serão tratadas, em mais detalhes, no Capítulo 6.

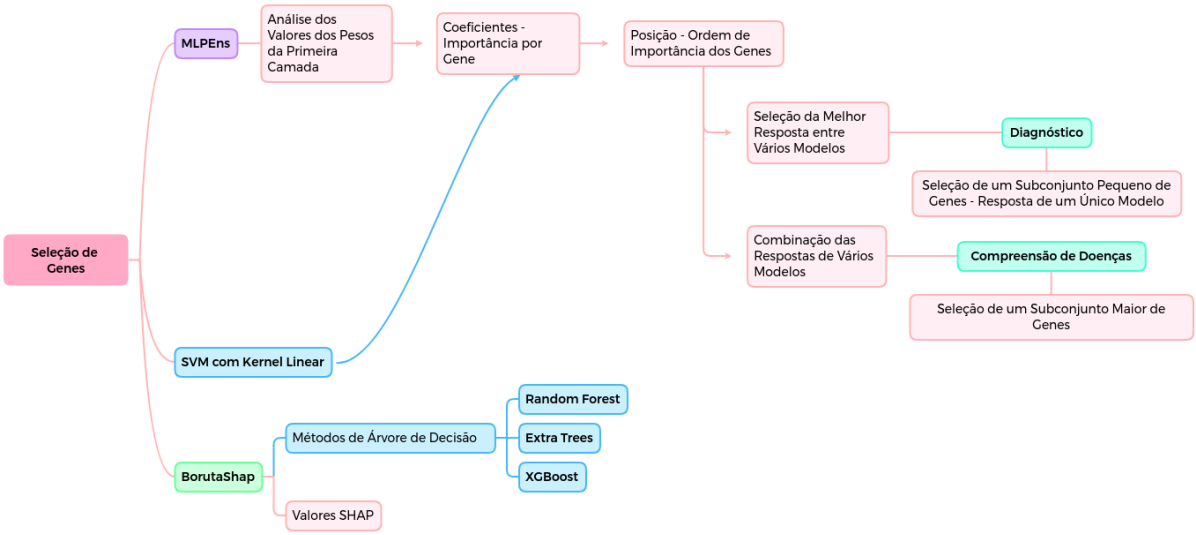

Figura 16 – Esquema das abordagens de seleção de genes relevantes adotadas.

Nós adotamos uma abordagem de *wrapper feature selection*, o BorutaShap (KEANY, 2020), que emprega várias execuções de classificadores de Aprendizagem de Máquina baseados em árvores de decisão e valores de Shap para avaliar a importância dada aos *features* pelos classificadores. Como métodos de classificação em conjunto com o BorutaShap são investigados as Random Forest; Extra Trees; e o XGBoost.

Nós também utilizamos duas metodologias de *embedded feature selection*, analisando as importâncias que os modelos de Máquinas de Vetor Suporte (SVM) e da

rede neural MLPEns atribuem aos genes. Nessas duas últimas metodologias, analisamos dois critérios para obter um rank de genes mais importantes e combinar os resultados de diferentes modelos:

- usando os coeficientes, ou seja, os valores de importância computados pelos classificadores;
- estabelecendo um rank de ordem de genes mais importantes por modelo e atribuindo, como valor de importância, a posição que cada gene assume nesse rank.

Ressaltamos que a análise combinada de vários modelos, mesmo que de um mesmo tipo de classificador, é de suma importância devido à alta redundância e correlações nos dados de expressão gênica, especialmente, quando a tarefa de interesse for a compreensão de doenças.

4.8 Comparação dos Métodos de Aprendizagem de Máquina para Seleção de Genes Relevantes

A avaliação dos conjuntos de genes selecionados é feita da seguinte forma. Para o BorutaShap, na qual se emprega apenas uma execução do método, compara-se a acurácia de classificadores, usando apenas os genes selecionados pelo método, em um conjunto de teste não empregado na escolha dos genes importantes.

Para ambas as abordagens, apresentam-se *heatmaps* (agrupamentos hierárquicos) e t-SNEs para inspeção visual da separabilidade das classes, utilizando apenas os *features* selecionados como importantes. Para o dataset de Sarcoma de Ewing, apresenta-se, também, base biológica, disponível na literatura, corroborando a relevância dos genes selecionados.

5 MLPENS

Neste capítulo, propõe-se uma abordagem que utiliza o mecanismo de dropout para explorar um modelo de Rede Neural como um ensemble de modelos, objetivando aproveitar a alta capacidade de generalização normalmente obtida em *ensembles* de classificadores.

5.1 Rede Neural Vista como um Ensemble de Caminhos

Inspirando-se em conhecimentos biológicos sobre **Potenciais de Ação**, propomos uma nova perspectiva sobre a técnica de *dropout*, visando fornecer uma boa capacidade de generalização em *datasets* com poucas amostras.

A ideia por trás da nossa abordagem está em perceber que os neurônios, assim como as células cardíacas, possuem um **período de refratário** após a ocorrência de um Potencial de Ação, devido à necessidade de reacomodação dos canais iônicos na membrana plasmática e retorno ao potencial de repouso da célula (GARCIA, 2015). Ou seja, um neurônio não é capaz de produzir Potenciais de Ação em tempo permanente. Esse aspecto biológico é muito semelhante à técnica **dropout**: em determinados momentos, alguns neurônios estariam desativados.

Ao analisar esse ponto, pensamos o seguinte: uma dada informação, seja uma imagem, ou um texto, parece percorrer os caminhos neuronais de formas distintas, em diferentes instantes de tempo, enquanto pensamos. Caminhos esses determinados conforme os neurônios estão em latência ou não. As decisões do nosso cérebro devem, portanto, ser tomadas pela concordância das conclusões obtidas nos diferentes caminhos pelos quais a informação passa.

Pensando sobre o ponto de vista de Aprendizagem de Máquina, podemos imaginar essa perspectiva sobre o cérebro como um ***ensemble* de modelos dos diversos caminhos pelo qual a informação pode passar**. Como *ensembles*, em geral, essa perspectiva favorece a obtenção de soluções que evitem o *overfitting*.

Para possibilitar a adoção da Rede Neural como um *ensemble*, propomos:

- o emprego da **predição** como uma combinação das diferentes saídas de uma

mesma rede neural com *dropout* ativado;

- uma forma de determinar os **melhores conjuntos de pesos** (tratando a rede neural como um *ensemble*);
- e uma forma simples de realizar ***feature selection*** nessa rede neural, para a seleção de genes relevantes.

Ainda, estendendo mais essa ideia de *ensemble* dos caminhos pelo qual a informação passa, propomos uma **arquitetura** simples baseada na rede neural *Multi-layer Perceptron* (MLP), nomeada de **MLPEns**, que realiza o *ensemble* de três caminhos distintos e estabelecemos formas de computar ***loss*** para essa arquitetura.

Nós aplicamos a abordagem proposta para tratar a rede neural como um *ensemble* na MLPEns, mas a mesma pode ser estendida a outras arquiteturas.

5.2 Arquitetura com Ensemble de Caminhos

Avançando a ideia de *ensemble* de caminhos pelo qual a informação passa, propomos a adoção de uma extensão da MLP (nomeada de **MLPEns**), que combina a saída de três caminhos: após cada camada escondida da rede e antes da aplicação de *dropout*, dividimos a informação em mais um novo caminho. A Figura 17 esquematiza a **arquitetura proposta**:

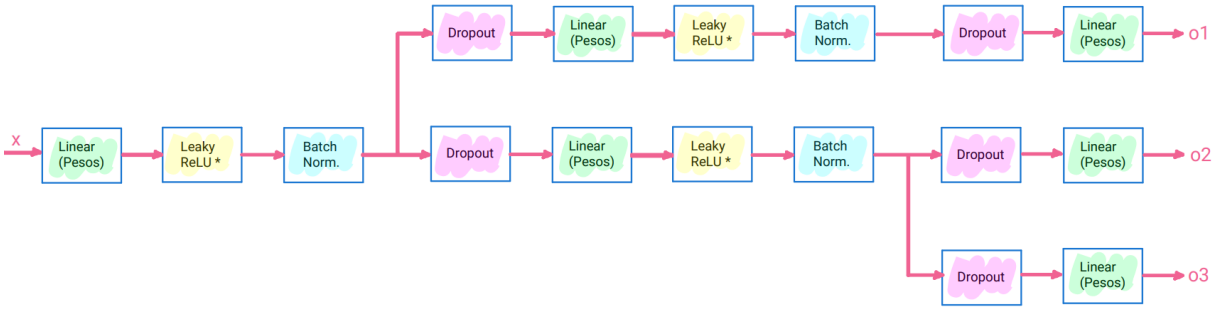

Figura 17 – Arquitetura da MLPEns proposta.

Como, antes do início de cada novo caminho de informação, coloca-se um *dropout* distinto, estamos desativamos diferentes neurônios em cada caminho. Em uma rede neural com três camadas escondidas, portanto, temos três caminhos distintos, fornecendo três saídas $o1$, $o2$ e $o3$ para cada entrada x propagada. Essas saídas podem ser combinadas usando regras simples de combinação de classificadores, com votação, produto, soma, máximo ou outras regras.

Neste trabalho, nós tratamos a saída individual de cada caminho como probabilidades (passando a saída pela função *softmax*) e a combinação das probabilidades dos diferentes caminhos, portanto, é o produtório das probabilidades individuais. Para evitar tratar produtórios envolvendo números muito pequenos e, consecutivamente,

problemas de incapacidade de representação de muitas casas decimais, adotamos a soma dos logaritmos das probabilidades (soma de funções *log_softmax*).

5.2.1 Loss para a Arquitetura MLPEns

Para o treinamento dessa arquitetura, precisamos utilizar uma ***loss*** que penalize o erro dos diferentes caminhos. Pensamos em três formas de computá-la:

- $individual$ - a loss é a soma da *cross-entropy* de cada um dos três caminhos - cada caminho deve por si só apresentar uma boa solução;
- $combinada$ - antes do cálculo da *cross-entropy*, combinam-se as saídas dos caminhos (como, por exemplo, usando a soma do logaritmo das probabilidades obtidas por cada caminho, soma de $log\_softmax$) - nesse caso, pensa-se que a combinação dos três caminhos deve ser a melhor possível;
- $individual\_combinada$ - realizamos uma soma das duas *loss* citadas acima - ou seja, tanto os caminhos individuais, quanto a combinação dos mesmos devem fornecer bons resultados.

5.3 Predição - Tratando a Rede Neural como um *Ensemble*

Para obter a predição para amostras de um conjunto de teste, nós propagamos os *features* de cada amostra $nr\_propagacoes\_predicao$ vezes, com *dropout* ativado com alta probabilidade $prob\_dropout\_predicao$.

A saída do ensemble é, então, obtida por alguma regra de combinação de saídas de modelos. Neste trabalho, nós adotamos a regra de votação e a soma dos logaritmos da função *softmax* das saídas individuais.

Ressalta-se que, na etapa de predição, não é necessário computar os gradientes da rede neural, o quê reduz consideravelmente o tempo de executar propagações pela rede neural.

5.4 Determinando o Melhor Conjunto de Pesos

Quando o número de amostras disponíveis é pequeno, um problema a ser enfrentado é determinar o melhor conjunto de pesos. Duas estratégias são as mais usadas:

- selecionar o conjunto de pesos com menor loss no conjunto de treinamento, o quê poderia causar a seleção de pesos com *overfitting* no conjunto de treino;
- selecionar o conjunto de pesos que obtém menor *loss* para um conjunto de validação, o quê diminui o número de amostras disponíveis para treino e ainda pode causar especialização no conjunto de validação.

Sendo assim, preferimos empregar todas as amostras do conjunto de treino para o treinamento e como uma forma de evitar o *overfitting*, selecionamos o conjunto de pesos que fornece melhor resposta para $nr\_propagacoes\_escolha\_pesos$ com *dropout* ativado com probabilidade $prob\_dropout$ - ou seja, selecionamos o conjunto de peso cuja soma da *loss* de $nr\_propagacoes\_escolha\_pesos$ é o menor possível. Os valores de *loss* são sempre computados no conjunto de treinamento.

Deve-se destacar que, para determinar o melhor conjunto de pesos, não é necessário computar os gradientes da rede neural, reduzindo consideravelmente o tempo de teste se um novo conjunto de peso é melhor do que o anterior. Ainda, para diminuir o tempo de processamento, se a soma que está computando se tornar superior a soma do melhor conjunto anterior, pode-se encerrar as propagações.

5.5 Interpretabilidade da Rede Neural - *Feature Selection*

Uma forma valiosa de interpretabilidade em modelos é determinar quais *features* são mais importantes para a decisão tomada. Especialmente nos casos de análises de dados de expressão gênica, é desejável uma forma de determinar quais genes são os mais importantes para, por exemplo, definir estratégias de diagnóstico, bem como, para definir possíveis alvos para investigação de terapias.

Na nossa abordagem, a utilização de altas taxas de *dropout* possibilita a seguinte percepção: os *features* mais importantes devem ser aqueles que estão associados aos **neurônios de entrada que mais dispersam informação**. Pois, se determinado *feature* for importante é desejável que ele tenha alta probabilidade de passar sua informação para as próximas camadas da rede neural. Portanto, analisando a dispersão dos *features* - pesos que partem dos neurônios da camada de entrada para a primeira camada escondida, poderemos perceber quais *features* são os mais importantes.

Para medir essa dispersão, nós computamos, para cada neurônio da camada de entrada da rede neural x_i, a soma de todos pesos w_{ij} que partem desse neurônio para a próxima camada. Como cada *feature* (gene) é associado a um único neurônio da camada de entrada, essa soma fornece o valor de importância do gene em questão.

6 SELEÇÃO DE GENES RELEVANTES

Este capítulo explica as metodologias estabelecidas para a seleção de genes relevantes nas tarefas de classificação.

6.1 Seleção de *Features*

Feature selection é um problema de otimização que envolve a busca no espaço de possíveis atributos por um subconjunto de atributos que otimizam algum critério (exemplo, acurácia) (KHALID; KHALIL; NASREEN, 2014). As abordagens de *feature selection* podem ser classificadas em quatro categorias (FONTES, 2020), citadas a seguir.

- Filtros - não se baseiam em métodos de aprendizagem de máquina, apresentam baixo custo computacional, mas menor acurácia em relação às abordagens *wrappers* (ex de método: Fisher score).
- *Wrappers* - resolvem o problema de seleção de atributos otimizando o resultado alcançado pelo classificador, são melhores que as abordagens de filtros, no entanto, apresentam maior custo computacional (ex de método: Algoritmos Genéticos com algum classificador).
- *Embedded* - emprega uma métrica intrínseca do método de Aprendizagem de Máquina usado como classificador (ex de método: Árvores de Decisão).
- Híbrida - combinam abordagens baseadas em filter ou *embedded* e *wrappers*. Em geral, o processamento dos dados de alta dimensionalidade é uma tarefa difícil para um método *wrapper* (FONTES, 2020), pois este precisa treinar vários classificadores para tomar decisões sobre uma quantidade elevada de atributos. Nas abordagens híbridas, então, apenas os *features* que obtiverem um maior valor de importância dado por método de filtro ou *embedded* são fornecidos ao método *wrapper*.

Neste trabalho, nós empregamos o **BorutaShap** (KEANY, 2020) que é um método de *wrapper feature selecion*. Bem como, empregam-se as análises dos coeficientes de Máquinas de Vetor Suporte (SVM) e dos pesos da primeira camada da rede

neural MLPEns. Essas duas metologias podem ser classificadas como abordagens de *feature selection embedded*, pois empregam métricas intrínsecas aos métodos de Aprendizagem de Máquina usados como classificadores.

6.2 Seleção de Genes Relevantes

É importante notar que dados de expressão gênica (com alta dimensionalidade) apresentam alta **redundância** para classificadores. Ou seja, muitos features (genes) poderiam ser usados para uma mesma tarefa de classificação, pois um gene interage com vários outros; de forma que, para um classificador, muitos genes poderiam escolhidos de forma equivalente.

Tendo essa observação em mente, podemos perceber duas formas distintas de selecionar um subconjunto de genes relevantes, adequada a dois tipos diferentes de aplicações, listadas a seguir.

- Aplicação 1 - Tarefa de **diagnóstico** (classificação). Nessa aplicação, é interessante selecionar um subconjunto pequeno de genes, o menor possível, que permita diagnosticar (separar classes) com precisão.
- Aplicação 2 - Fornecer **compreensões** e *insights* sobre **doenças** / fenômenos. Nesse caso, é interessante que todos os genes importantes no fenômeno (doença) que se está analisando sejam selecionados. Em outras palavras, é útil que todos features que possam ser equivalentemente tomados pelo classificador sejam determinados.

6.2.1 Seleção de Biomarcadores para Diagnóstico

Para obter uma solução para a primeira aplicação (diagnóstico), basta executar uma vez um classificador e analisar quais genes o mesmo selecionou como mais importantes e teremos um subconjunto pequeno de features que pode separar as classes analisadas.

Alternativamente, podemos executar várias vezes os classificadores, usando sementes aleatórias distintas e, assim, obteremos vários subconjuntos distintos de features, sendo que cada subconjunto separa adequadamente as classes.

Então, escolhemos, como biomarcadores, o menor subconjunto tomado entre todos classificadores. Por exemplo, escolhendo o subconjunto de genes, cujos $nr\_biomarcadores$ primeiros da lista ordenada apresentem um maior valor de soma de importâncias (tendo o cuidado de escalar primeiro os valores de importância para que todos classificadores tenham um mesmo valor final de soma de importância para todos os features).

6.2.2 Seleção de Genes Relevantes para a Compreensão de um Fenômeno / Doença

Para a segunda aplicação (compreensão de doenças), pode-se executar classificadores várias vezes, usando sementes aleatórias distintas. E, a seguir, combinamos os subconjuntos de features selecionados por cada classificador, para obter uma única lista ordenada de features segundo a sua importância.

Uma possível forma de realizar essa combinação é calcular a média de importâncias de cada *feature* calculadas pelos classificadores, tendo o cuidado de, primeiramente, escalar os valores de importância dos *features* para que cada classificador contribua de igual forma no valor de média calculado.

6.3 SVM e MLPEns

Nos modelos de **Máquina de Vetores Suporte (SVM)** com kernel linear, temos um valor de importância dado a cada atributo (*feature*) - um coeficiente.

De forma semelhante, pode-se obter um valor de importância dado a cada *feature* (gene) por um modelo de **MLPEns**, analisando os pesos da rede neural. Nós empregamos a soma de todos os pesos conectados a cada neurônio da camada de entrada. Como cada um dos neurônios da primeira camada é relacionado a um gene, pode-se assumir a soma dos pesos saindo desse neurônio como um valor de importância dado ao gene.

6.3.1 Combinando Valores de Importância para Vários Modelos

Para combinar os valores de importância de vários modelos dos métodos **SVM** e **MLPEns** temos duas formas (critérios) para selecionar os genes:

- através dos **coeficientes** propriamente ditos (ou através da média de coeficientes para a combinação dos modelos);
- estabelecendo um rank de *features* (genes) mais importantes dos modelos e, após, realizando uma média das **posições** nos ranks dos diferentes modelos a serem combinados.

Como mencionado anteriormente, a combinação de vários modelos é importante especialmente ao se analisar dados com alta redundância como os dados de expressão gênica.

6.4 BorutaShap

O BorutaShap (KEANY, 2020) é um método de *wrapper feature selection* que combina o algoritmo de seleção de *features* Boruta com valores Shapley. O BorutaShap

pode ser utilizado em conjunto com qualquer classificador baseado em Árvore de Decisão.

Esse método é iterativo: em cada iteração, o método executa um classificador baseado em Árvores de Decisão, usando, como entrada, os *features* reais do *dataset* e *features* criados embaralhando os valores dos *features* reais. Então, avalia-se a importância que o classificador deposita em cada um dos *features* de entrada com os valores Shapley - contrastando a importância dada aos *features* reais com os criados pelo embaralhamento. Após, cada iteração, o método realiza um teste Binomial bilateral para verificar se a importância dada a cada um dos *features* reais difere da fornecida ao melhor *feature* obtido pelo embaralhamento.

Esse método tem a vantagem de se basear em um formalismo matemático forte, entretanto, em dados de expressão gênica, com alto número de *features*, o tempo de processamento que o BorutaShap requer para tomar decisões sobre todos os *features* é elevado.

6.5 Visualização - t-SNE

Uma forma de avaliar visualmente a seleção de genes realizada é através do emprego de *Heatmaps* (Agrupamentos Hierárquicos) e t-SNEs.

t-Distributed Stochastic Neighbor Embedding (t-SNE) (MAATEN; HINTON, 2008) é uma técnica não-linear que emprega estatística para redução de dimensionalidade. O t-SNE é bem adequado para visualização de dados de alta dimensionalidade. Cada amostra dos dados de alta dimensionalidade é colocada em um espaço de menor dimensão, geralmente 2D ou 3D.

A similaridade entre cada par de amostras x_i e x_j, no espaço de alta dimensionalidade, é modelada como uma probabilidade condicional $p_{j|i}$, que representa a probabilidade de que x_i escolha x_j como seu vizinho, se os vizinhos fossem escolhidos proporcionalmente a uma densidade de probabilidade gaussiana centrada em x_i.

Também, modela-se uma probabilidade condicional $q_{j|i}$ para representar a similaridade entre y_i e y_j, que são as correspondentes a x_i e x_j no espaço de menor dimensionalidade. Para $q_{j|i}$, o t-SNE emprega a distribuição t-Student, ao invés da gaussiana.

O t-SNE visa encontrar uma representação dos dados em um espaço de baixa dimensionalidade que minimize as diferenças entre $p_{j|i}$ e $q_{j|i}$. Para isso, o t-SNE minimiza a soma da divergência de Kullback-Leibler entre $p_{j|i}$ e $q_{j|i}$ de todas as amostras, usando o método de gradiente descendente.

7 RESULTADOS E DISCUSSÕES - CLASSIFICAÇÃO

Este capítulo apresenta os resultados obtidos com os métodos de classificação investigados. Primeiramente, são descritos os experimentos iniciais no dataset piloto com os métodos CNN e SVM. Posteriormente, apresentam-se os resultados da combinação de saídas de diferentes modelos desses dois métodos (ensemble de modelos). Então, apresentam-se os resultados do SVM com diferentes formas de entrada em 185 problemas e do XgBoost em 31 problemas.

A seguir, descrevemos os resultados usando 11 datasets selecionados. São apresentados os resultados com a ResNet, bem como, o estabelecimento de ordem dos features pelo Transcriptograma. Então, abordam-se os resultados da MLP e os da ResNet18 Pré-treinada na ImageNet. Após, descrevem-se os resultados das abordagens com mecanismos de atenção: ResNet Pre-Activated Self-Attention, Vision Transform e Hybrid Vision Transformer. Então, apresentam-se os resultados obtidos com a GAN-MLP e MLPEns.

Demonstra-se, também, os resultados obtidos com 15 métodos de aprendizagem rasa no dataset piloto e de 10 métodos nos 11 problemas selecionados. Por fim, apresenta-se uma comparação dos melhores métodos.

7.1 Experimentos Iniciais - CNN x SVM

As primeiras análises de métodos foram executadas no **dataset piloto de Medulloblastoma** GSE85217 com $N = 763$ amostras, para classificação dos quatro subgrupos moleculares: *Group 3, Group 4, SHH e WNT*. Nesses cânceres, existe alta diferença de prognóstico entre os diferentes grupos, especialmente entre o Group3 (pior prognóstico) e WNT (melhor prognóstico), sendo que a obtenção de marcadores capazes de diferenciá-los ou ainda fornecer *'insights'* sobre esse tipo de câncer seria de grande valia.

A escolha desse dataset como piloto, portanto, se deveu ao fato dessa separação de subgrupos ser um problema biológico relevante e devido ao fato desse ser um dataset com relativo maior número de amostras em comparação com a maioria dos

datasets de *microarray* sobre um tipo específico de câncer. Destaca-se que embora o mesmo tenha alto número de amostra em relação a outros datasets, ainda assim, o número de features ($nr\_genes = 21.641$ genes) é muito maior que o número de amostras, permitindo a avaliação dos métodos analisados em problemas com maldição da dimensionalidade (*curse of dimensionality* - número de features muito maior que o número de amostras), que é o foco deste trabalho.

Os primeiros métodos de Aprendizagem de Máquina analisados foram as Redes Neurais Convolucionais (CNNs) e a Máquina de Vetores Suporte (SVM). As **CNNs** foram escolhidas devido aos pontos destacados no Capítulo 1 que motivaram um enfoque deste trabalhor maior nas Redes Neurais Profundas, devido à percepção da investigação do uso desses métodos em dados de expressão ser um problema computacional relevante.

O **SVM**, por outro lado, foi escolhido como abordagem de Aprendizagem Rasa *baseline*, devido aos resultados relatados no dataset CuMiDa (GRISCI et al., 2019) que o mostrou superior aos métodos *Multi-layer Perceptron* (MLP), Árvores de Decisão, *Naive Bayes*, *Random Forest* (RF), *Hierarchical Clustering* e *k Nearest Neighbor* (k-NN).

Nesses experimentos iniciais, antes da utilização dos classificadores, executou-se um **controle de qualidade de amostras**, para verificar se existiam amostras mais ruidosas que deveriam ser removidas. Para isso, empregou-se o pacote *arrayQualityMetrics* do *Bioconductor* (R). Todas as amostras que foram destacadas como possível *outlier*, por alguma das análises executadas pelo pacote, foram removidas (35 amostras) e foram mantidas 728.

Nos experimentos da CNN, foram investigadas as seguintes **arquiteturas**: ResNet com diferentes números de camadas, AlexNet, EfficientNet, SparseNet e Xception. A arquitetura **ResNet** apresentou melhores resultados e considerável menor tempo de execução. Portanto, nos demais experimentos usamos apenas essa arquitetura de CNN. As implementações foram feitas utilizando o *PyTorch*.

Como **otimizadores** experimentaram-se o Adam, RAdam e SGD. O Adam e o RAdam apresentaram resultados similares e superiores ao SGD. Portanto, nos demais experimentos fixamos o otimizador como o Adam. Como função de perda se fixou a **cross entropy**. Como **scheduler** para a *learning rate* foi escolhido o **OneCycleLR**.

Para mecanimos de **regularização** foi experimentado a regularização Post-synaptic potential (PSP) (TARTAGLIONE; PERLO; GRANGETTO, 2019), o dropout na learning rate (LIN et al., 2022), max-norn (GÉRON, 2019), L1 e L2; mas não conseguimos verificar melhorias nos resultados. Assim, nos experimentos posteriores, para regularização, empregou-se apenas o **dropout** e a **Batch Normalization**.

Como **condições de parada** para o treinamento das redes neurais, adotou-se número de épocas máximo e saída por alcance do valor de mínima loss igual a $1 * 10^{-8}$.

Nos experimentos iniciais testaram-se diferentes números máximo de épocas e após fixou-se em 500.

Para a escolha do **melhor conjunto de pesos** testaram-se três abordagens, nos experimentos iniciais:

- i) útimos pesos, obtidos ao final do treinamento;
- ii) conjunto com menor valor de *loss* avaliada no conjunto de treinamento;
- iii) melhor acurácia e menor *loss* avaliada em um conjunto de validação - antes do início do treino, as amostras do conjunto de treinamento são separadas em um conjunto de validação (testado 30%, 50% e 70% do tamanho do conjunto de treino), mantido fora da computação dos gradientes no treino e usado apenas para avaliar os pesos.

Não foi possível notar grandes diferenças entre as abordagens (i) e (ii). Por outro lado, a abordagem (iii) não demonstrou bons resultados, o treinamento alcançava valores de acurácia elevada para o conjunto de validação rapidamente e, normalmente, os pesos escolhidos não eram os melhores para o conjunto de teste.

Concluímos que isso deve ter ocorrido por termos muito poucas amostras, em relação ao número de *features*, e essa condição de escolha de pesos estava fazendo com que fossem escolhidos pesos especializados no conjunto de validação, mas não os melhores para o conjunto de teste e, além disso, diminuía o número de amostras disponíveis para o treinamento. Assim, nos experimentos posteriores, usamos apenas as abordagens (i) e (ii).

Os dados (expressões por genes) foram escalados usando **Min-Max Feature Scaling** no intervalo de 0 e 1, com os valores de mínimo e máximo obtidos usando apenas o conjunto de treinamento. Os dados do conjunto de teste que, após serem escalados, continuaram fora do intervalo especificado foram cortados para que também variassem entre 0 e 1.

A implementação usada do **SVM** foi a disponível no módulo linearSVC da sklearn (python); nestes experimentos iniciais empregou-se apenas o SVM com kernel linear L1.

O **dataset foi dividido** em 70% das amostras para treino e 30% para teste. Cada **semente aleatória** utilizada estabeleceu uma divisão diferente do dataset e uma inicialização diferente dos modelos. Divisões diferentes do dataset foram empregadas para que as médias de acurácias avaliadas, para cada método de Aprendizagem de Máquina, fossem o menos possível sujeitas a algum viés causado por uma divisão de amostras específica.

7.1.1 Resultados e Curva de Aprendizagem

Nesses experimentos iniciais, o **SVM apresentou melhores resultados** que ambas as arquiteturas de CNN no dataset piloto. Considerando três sementes aleatórias, o SVM com kernel linear L1 alcançou média de acurácia de **98.646%** e a CNN, com arquitetura ResNet de 18 camadas, 64 canais por camada e escolha de pesos pela menor *loss* avaliada no conjunto de treinamento, obteve **98.384%**. Embora seja pouca diferença, esses resultados foram desmotivadores.

Visto que as Redes Neurais Profundas, como as CNNs, costumam exigir grande número de amostras, analisou-se, então, a **curva de aprendizagem** (*learning curve*) (ver Figura 18) (NG, 2017; HARRISON, 2019; GRUS, 2019). Essa curva foi montada fixando um único conjunto de teste com 30% das amostras disponíveis e tomando parcelas distintas das 70% amostras restantes como conjunto de treinamento. Iniciando com um conjunto de treinamento de apenas 10% das amostras e aumentando até a utilização dos 70% das amostras para treino. Para cada tamanho de dataset de treino tomado, realizaram-se 10 execuções (com 10 sementes aleatórias).

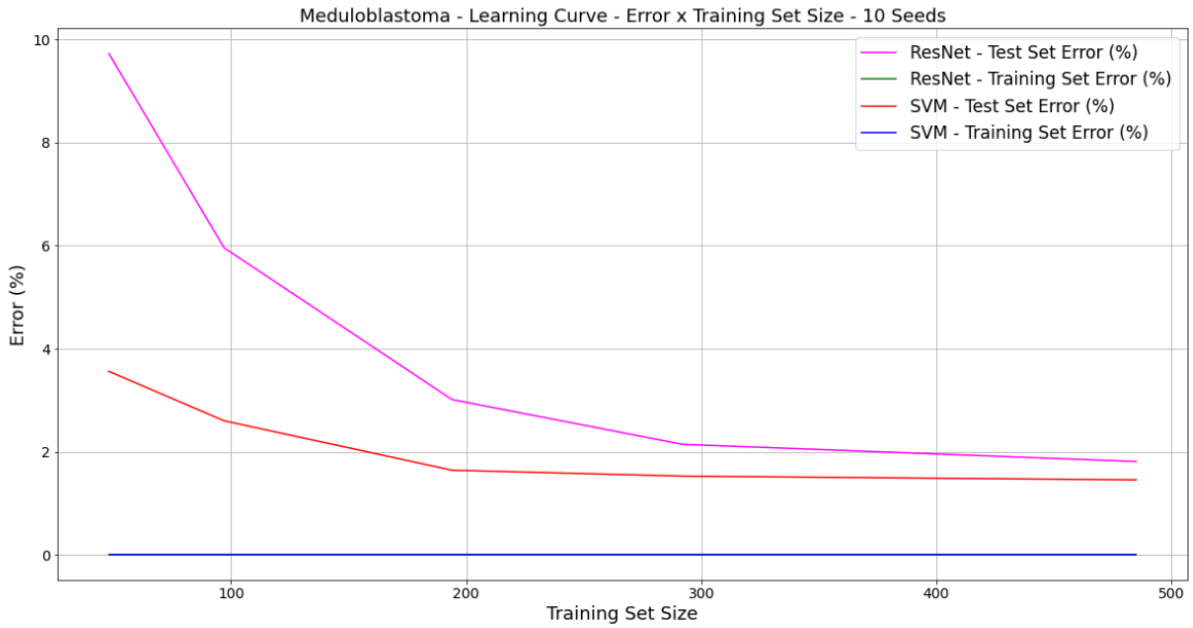

Figura 18 – Curva de aprendizagem do SVM e CNN para comparação de tendência de caimento da curva e verificação da polarização (bias) versus variância. A curva mostra valores de erro de acurácia, conforme se aumenta o número de amostras usadas no conjunto de treinamento para um mesmo conjunto de teste.

Essa curva permite verificar a **tendência de caimento** do erro de acurácia, conforme se aumenta o número de amostras usadas no treino. Se for percebido um alto caimento, indica que mais dados poderiam ser usados para melhorar o desempenho do método.

Além disso, a análise da curva de aprendizagem permite verificar o compromisso entre **polarização (bias)** e **variância** dos métodos, como explicado a seguir.

- Se existir uma variabilidade na acurácia do conjunto de treino e/ou um erro maior do que o esperado nesse conjunto indica que está havendo uma **subadequação (*underfitting*)** do modelo aos dados disponíveis para treino; ou seja, dizemos que o modelo está sofrendo com erro de **polarização (viés, *bias*)**.
  Nesse caso, o modelo não apresenta expressividade o suficiente para aprender todas as características dos dados disponíveis. Para resolver esse problema, deveríamos aumentar a expressividade do modelo, por exemplo, acrescentando mais neurônios em uma rede neural.

- Por outro lado, se o desempenho do conjunto de teste for muito menor do que a do conjunto de treino indica que houve **superadequação (*overfitting*)** do modelo às amostras de treino e o mesmo está sofrendo de **variância**.
  Ou seja, o modelo está se especializando demais no conjunto de treinamento e perdendo capacidade generalização. Nesse caso, a adição de mais amostras no conjunto de treinamento ou a diminuição da expressividade do modelo podem melhorar o desempenho do método.

Esperava-se obter uma tendência de caimento maior para CNN em relação ao SVM - devido ao conhecimento da literatura que Redes Neurais Profundas precisam de alto número de amostras para treino. Essa diferença de caimento esperada não ficou tão evidenciada na curva (Figura 18).

Mas pela verificação que o erro no conjunto de treinamento de ambos os métodos estão em zero, podemos perceber que **o problema da CNN é perda de generalidade**, ou seja, o método está sofrendo de **variância** ou superadequação (*overfitting*) do modelo às amostras do treinamento.

Tal conclusão vem ao encontro do que se conhece na literatura, de que as **Redes Neurais Profundas apresentam alta expressividade** (muitos parâmetros para serem aprendidos); e a alta dimensionalidade dos dados (muitos *features*) também dificulta a generalidade dos modelos a serem aprendidos.

Com essa observação em mente e sabendo que não conseguiríamos mais amostras para treino, o próximo experimento realizado visou favorecer a generalidade dos modelos aprendidos, usando combinação (*ensemble*) de modelos com regras simples, como explicado na próxima Seção.

7.2 Ensemble de CNN x Ensemble de SVM

Nesse experimento comparamos *ensemble* de CNNs com *ensemble* de SVMs. Usamos CNNs com menor expressividade (em uma tentativa de diminuir o *overfitting*), adotamos apenas 8 canais em cada camada, utilizamos uma ResNet de 18 camadas

e um número reduzido de épocas de treinamento, apenas 30 épocas (*early stopping*); e empregamos SVM com kernel linear L1.

Realizamos experimentos indo de *ensembles* com $nr\_modelos = 3$ modelos até $nr\_modelos = 10$ modelos de um mesmo método (CNN ou SVM). A diferença em cada modelo de um único *ensemble* foi apenas a semente aleatória de inicialização dos modelos. Todos modelos foram treinados usando o mesmo conjunto de treinamento. Para a combinação das saídas dos modelos, empregaram-se as seguintes regras: média, votação, soma, produto, bayesiana, baiyesiana considerando falsos alarmes (Bayesiana FA).

Nós criamos 10 ensembles para cada um dos métodos (CNN e SVM) e para cada valor de $nr\_modelos$. A Tabela 2, a seguir, mostra os resultados obtidos. Como pode ser percebido, *ensembles* com 4 ou mais modelos de **CNN superam os de SVM**. As melhores regras para a CNN foram a bayesiana e a bayesiana com falsos alarmes; e, para o SVM, foi a votação.

Tabela 2 – Ensemble de SVM x Ensemble de CNN - com diferentes regras de combinação e número de modelos em cada ensemble.

		SVM						CNN					
Nr Modelos		**Média**	**Votação**	**Soma**	**Produto**	**Bayesiana**	**Bayesiana FA**	**Média**	**Votação**	**Soma**	**Produto**	**Bayesiana**	**Bayesiana FA**
3	Média	98.66	98.74	98.74	98.74	98.74	98.74	98.37	98.56	98.61	98.63	98.63	98.61
	Std	0.25	0.25	0.24	0.24	0.24	0.24	0.23	0.34	0.29	0.31	0.31	0.29
	Mínimo	98.21	98.35	98.35	98.35	98.35	98.35	98.03	97.80	98.08	97.94	97.94	98.08
4	Média	98.67	98.70	98.67	98.67	98.67	98.67	98.37	98.53	98.78	98.79	98.80	98.79
	Std	0.23	0.26	0.28	0.28	0.28	0.28	0.21	0.26	0.28	0.32	0.29	0.27
	Mínimo	98.21	98.35	98.21	98.21	98.21	98.21	98.08	98.08	98.35	98.21	98.35	98.35
5	Média	98.66	98.79	98.74	98.74	98.74	98.74	98.28	98.71	98.79	98.79	98.80	98.79
	Std	0.22	0.27	0.24	0.24	0.24	0.24	0.22	0.28	0.38	0.40	0.38	0.38
	Mínimo	98.27	98.35	98.35	98.35	98.35	98.35	97.80	98.21	98.08	98.08	98.08	98.08
6	Média	98.67	98.78	98.78	98.78	98.78	98.78	98.27	98.67	98.79	98.79	98.79	98.82
	Std	0.21	0.26	0.22	0.22	0.22	0.22	0.21	0.31	0.42	0.42	0.44	0.42
	Mínimo	98.24	98.35	98.35	98.35	98.35	98.35	97.78	98.08	97.80	97.80	97.80	97.80
7	Média	98.66	98.75	98.72	98.72	98.72	98.72	98.26	98.80	98.87	98.83	98.83	98.86
	Std	0.22	0.24	0.25	0.25	0.25	0.25	0.19	0.35	0.36	0.35	0.35	0.36
	Mínimo	98.23	98.35	98.21	98.21	98.21	98.21	97.84	98.08	98.08	98.08	98.08	98.08
8	Média	98.65	98.75	98.72	98.72	98.74	98.74	98.22	98.70	98.83	98.83	98.82	98.82
	Std	0.20	0.23	0.22	0.21	0.19	0.19	0.17	0.35	0.35	0.35	0.35	0.35
	Mínimo	98.25	98.35	98.21	98.35	98.35	98.35	97.91	97.94	98.21	98.21	98.21	98.21
9	Média	98.61	98.72	98.71	98.68	98.71	98.71	98.24	98.79	98.82	98.89	98.89	98.86
	Std	0.21	0.24	0.20	0.21	0.20	0.20	0.16	0.37	0.40	0.36	0.34	0.35
	Mínimo	98.17	98.35	98.35	98.21	98.35	98.35	97.97	98.08	98.08	98.21	98.21	98.21
10	Média	98.62	98.78	98.76	98.74	98.75	98.76	98.21	98.71	98.87	98.89	98.90	98.91
	Std	0.20	0.23	0.20	0.19	0.19	0.20	0.16	0.31	0.34	0.32	0.31	0.31
	Mínimo	98.19	98.49	98.35	98.35	98.35	98.35	97.95	98.21	98.21	98.35	98.35	98.35
Média Total:		98.65	98.75	98.73	98.72	98.73	98.73	98.28	98.68	98.8	98.8	98.81	98.81

Esses resultados estão de acordo com a nossa suposição de que, pelo menos, uma das dificuldades da CNNs em tratar dados de expressão gênica estão na sua alta expressividade, fazendo com que as mesmas se especializem no conjunto de treinamento e percam capacidade de generalização.

7.3 Próximos Experimentos

Nos próximos experimentos e comparações finais dos diferentes métodos de Aprendizagem de Máquina, empregaram-se 20 **sementes aleatórias** (20 execuções de cada classificador, sendo a semente usada na inicialização dos modelos e na divisão dos datasets).

A maioria dos experimentos foi realizada empregando os 11 **problemas** definidos na Seção 4.2.1. Para o BorutaShap e uma pré-seleção de classificadores de Aprendizagem Rasa foi usado apenas o dataset piloto. Para o SVM, alguns experimentos foram com 185 problemas e, para o XGBoost, alguns experimentos foram executados com 31 problemas. Esses experimentos serão abordados nas próximas Seções.

Para os métodos SVM e XGBoost foram executados experimentos com e sem **ponderação por classes** (*class_weight*) - para que classificações errôneas de amostras de classes com menor número de amostras recebessem maior peso. Para os demais métodos de Aprendizagem de Máquina, não foi utilizado ponderação por classes.

7.3.1 Formas de Entradas

Para os métodos XGBoost e SVM, realizaram-se experimentos com cinco formas distantas de entrada (explicadas a seguir): probes; genes; tfs; genes-probes; e tfs-probes. Empregaram-se essas cinco formas de entrada com ou sem a realização de filtragem com o teste de **Kruskal-Wallis** com $pvalue = 0.05$, a implementação do filtro usada foi a disponível na *scipy. stats*.

Ou seja, para XGBoost e SVM, usamos ao todo 10 formas de entrada (cinco com filtro e cinco sem). Para os demais métodos de Aprendizagem de Máquina, escolhemos apenas a entrada no formato de **genes** com a realização de filtro **kruskal**. A seguir, explicamos os procedimentos para a obtenção das cinco formas de entradas.

7.3.1.1 Sondas (probes)

Os valores de expressão foram lidos dos arquivos *series_matrix* obtidos no GEO (GEO, 2019). Com exceção do dataset GSE42743, cujos valores de expressão foram lidos a partir do csv disponibilizado pelo CuMiDa (GRISCI et al., 2019) (uma vez que, no GEO, para esse GSE, estavam disponíveis apenas os dados brutos).

Optou-se por empregar os dados dos arquivos obtidos no GEO, ao invés dos disponibilizados no CuMiDa, para poder fazer experimentos de diferentes formas de definição dos rótulos a serem utilizados na classificação. Após a leitura dos dados:

- removemos as sondas com valores faltantes;
- e realizamos a transformação logarítmica (log2) nos datasets que ainda não estavam com a transformação aplicada (a verificação se deveria ser aplicada a

transformação foi feita analisando os percentis de cada *feature*, seguindo o especificado no *script* do GEO2R do GEO database (GEO, 2019)).

7.3.1.2 Genes

Os valores de expressão da versão da entrada em sondas foram utilizados (ou seja, após a remoção de features com valores faltantes e da aplicação da transformação log2, caso fosse necessário). Então:

- removemos as sondas ambíguas (que possuiam associações a mais do que um gene);
- obtemos uma sonda por gene - no caso de existirem mais do que uma sonda mapeando para um único gene, escolhemos aquela de máxima variância considerando todas as amostras do dataset;
- e removemos as sondas que não eram associadas a qualquer gene.

Obtemos, assim, uma matriz de valores de expressão com um valor de expressão por amostra para cada gene. Essa versão de entrada permite reduzir o número de *features*, removendo redundâncias e favorecendo a tarefa de interpretabilidade (escolha de genes importantes).

7.3.1.3 TFs

Realizamos o mesmo procedimento executado para a obtenção da versão de entrada em genes, mas dessa vez empregando apenas as sondas associadas a genes definidos como Fatores de Transcrição (TFs) e obtemos, portanto, uma matriz com um valor por amostra para cada TF.

Essa versão de entrada permite reduzir consideravelmente o número de *features* e possibilita que sejam escolhidos, como *features* relevantes, genes com maior quantidade de informação disponível na literatura.

7.3.1.4 Sondas Associadas a Genes (genes-probes)

Esta versão de entrada é semelhante a de genes: o mesmo procedimento é realizado, excetuando a escolha de uma única sonda para representar cada gene. Na versão de entrada anterior, escolhemos uma sonda para cada gene (a de maior variância).

Enquanto que, nesta versão, mantemos todas sondas que podem ser mapeadas a um único gene (ou seja, sondas ambíguas e aquelas que não são mapeadas a qualquer gene continuam sendo removidas). Dessa forma - sem descartar qualquer sonda associada a algum gene, evitamos que sondas ruidosas possam ser escolhidas

para representar genes em detrimento de outras sondas descartadas que poderiam conter informações mais úteis (melhores representativas dos genes).

7.3.1.5 Sondas Associadas a TFs (tfs-probes)

Esta versão de entrada é semelhante a das sondas associadas a genes. Mas, nesta versão, mantemos todas as sondas que possam ser associadas a um único gene definido como TF (ao invés de qualquer gene).

7.4 SVM e XGBoost - Diferentes Formas de Entrada

7.4.1 Hiperparâmetros

O **SVM** foi executado para os **kernels**: Linear L2; Linear L1; RBF e Poly. Empregou-se máximo número de iterações como 10.000. A implementação usada foi a disponível no *sklearn*, módulo *LinearSVC* para os kernels lineares e *SVC* para os demais.

A implementação do **XGBoost** empregada foi a do *XGBClassifier* disponível no módulo *xgboost*. O método foi executado com os hiperparâmetros *default* do modelo, com exceção dos seguintes hiperparâmetros:

- *max_depth* (número máximo de profundidade de cada ramo das árvores) foi variado entre 1, 2 e 3, pois experimentos iniciais indicaram que esses valores forneciam resultados melhores do que o valor *default* de *max_depth=6*;
- *colsample_bynode* (proporção de features amostrados para serem considerados na definição de cada nodo das árvores de decisão) igual a 0.8;
- *colsample_bytree* (proporção de features amostrados para a definição de cada árvore de decisão) igual a 0.8;
- *subsample* (proporção de amostras consideradas em cada árvores de decisão) igual a 0.9.

Essas amostragens (*colsample_bynode, colsample_bytree* e *subsample*) favorecem a generalização dos modelos, pois, em cada momento, uma parcela do conjunto de treinamento é considerada.

Posteriormente, o XGBoost também foi executado com todos os parâmetros *default* usando o pacote PyCaret (ver Seção 7.11).

7.4.2 Demais Especificações

Tanto para o SVM quanto para o XGBoost, foram realizadas 20 **execuções** para cada problema tratado.

O **SVM** foi aplicado em 80 GSEs (datasets) diferentes com formas distintas de dividir as classes do datasets, totalizando **185 problemas** (cada problema é uma forma distinta de dividir as classes de um dataset) (ver Seção 4.2.1). Enquanto que, para o **XGBoost**, foram realizadas 20 execuções apenas para **31 problemas**.

Os métodos **SVM e XGBoost** foram executados com e sem **ponderação por classes** (*class_weight*). Foram consideradas as **diferentes formas de entrada**: sondas; genes; TFs (fatores de transcrição); somente sondas com associação a algum gene; somente sondas com associação a algum TF. Cada tipo de entrada foi aplicada sem filtragem dos *features* e utilizando o filtro ***kruskal***.

7.4.3 Resultados - Kernel - SVM

A Figura 19 compara os tipos de kernel do SVM, considerando a média de 20 execuções e 185 problemas. Como pode ser percebido, o SVM kernel linear L2 apresenta maior média de acurácias.

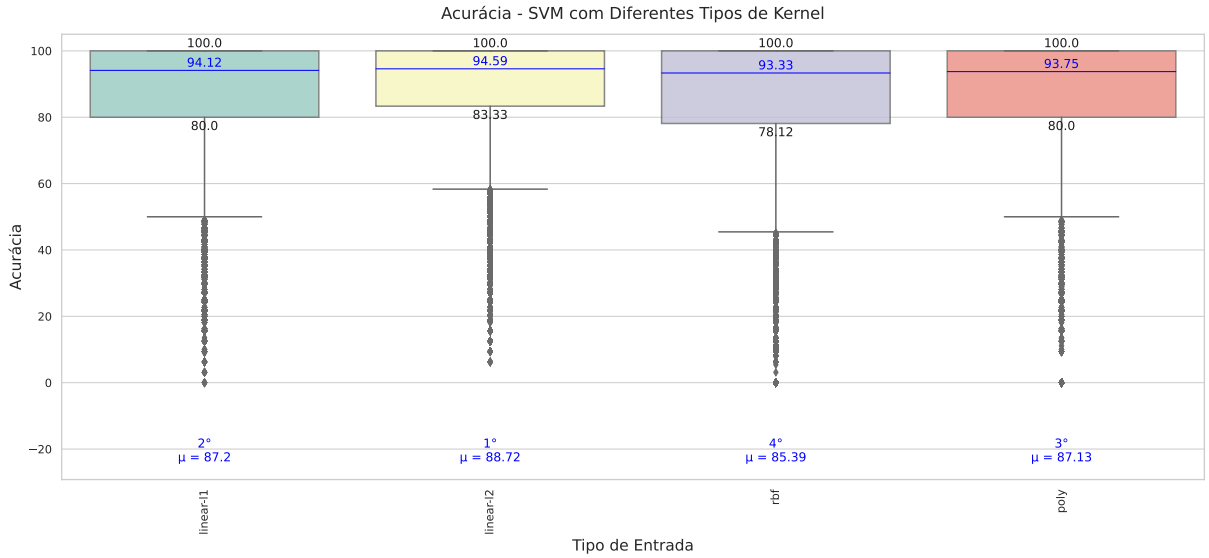

Figura 19 – Comparação dos Kenels do SVM (20 execuções e 185 problemas).

Entretanto, o melhor kernel varia de problema para a problema. A Figura 20 demonstra a porcentagem de problemas na qual cada kernel teve a melhor média de acurácia, na ordem de preferência para os kernels que a média geral de todos problemas deram maior.

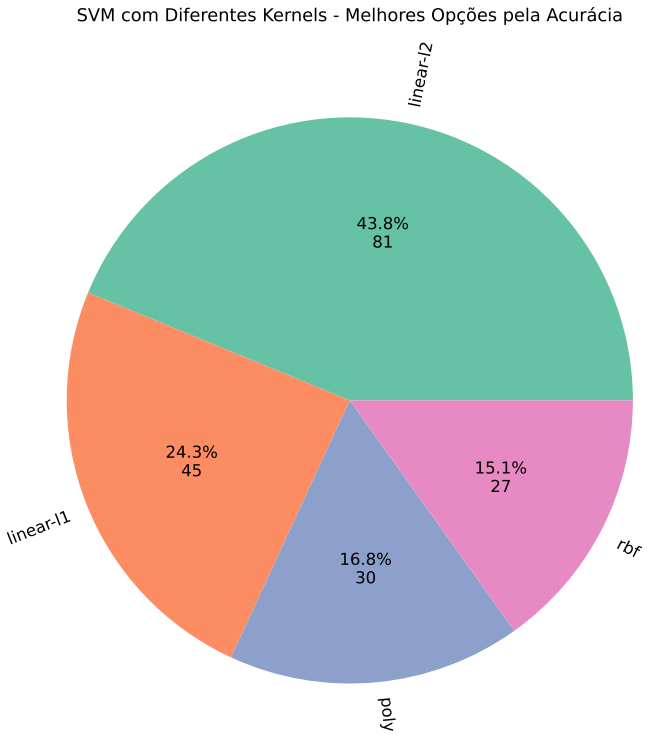

Figura 20 – Comparação dos Kenels do SVM - proporção de problemas que tem a maior média de acurácia em cada tipo de kernel (20 execuções e 185 problemas).

Fazendo o teste de hipóteses *bootstrap* pareado (KABACOFF, 2015; BRUCE; BRUCE, 2019) para comparação das médias de acurácia entre o SVM linear L2 e o SVM Linear L1 - os dois kernels de maior média de acurácias, concluímos, com 95% de confiança, que a média de acurácias do kernel **linear L2 é superior ao L1** em pelo menos 1.2% (ver Figura 21).

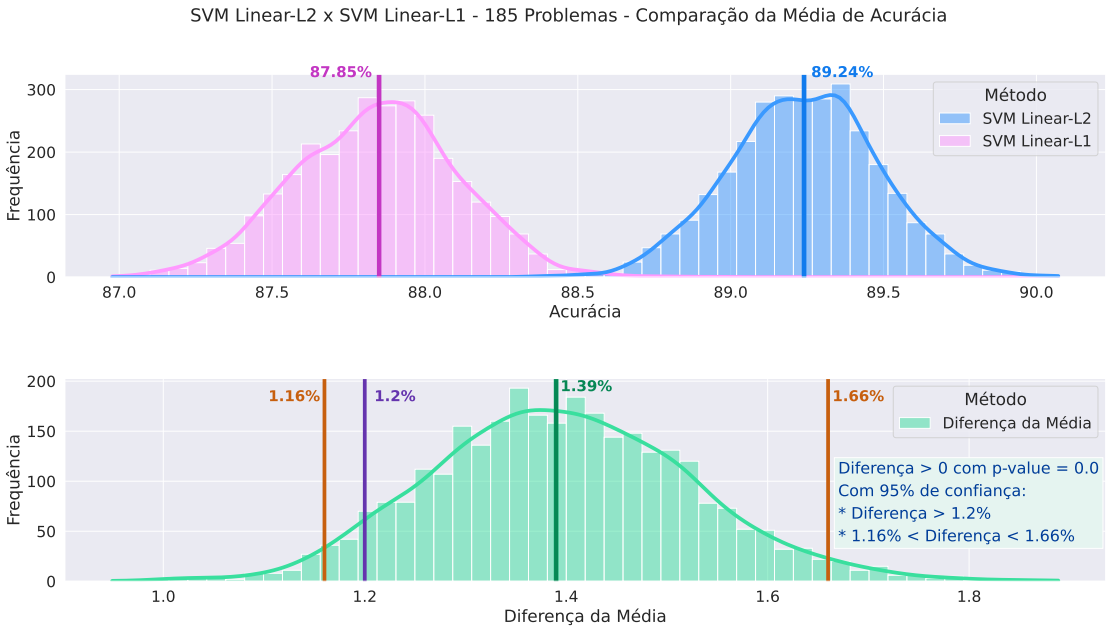

Figura 21 – Kernel Linear L2 x Linear L1 - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

7.4.4 Resultados - max_depth - XGBoost

A Figura 22 compara diferentes valores para o hiperparâmetro max_depth no XGBoost, considerando a média de 20 execuções e 31 problemas. Como pode ser percebido, existe pouca diferença nos valores de média de acurácia, sendo $max\_depth = 3$ o que apresenta maior média.

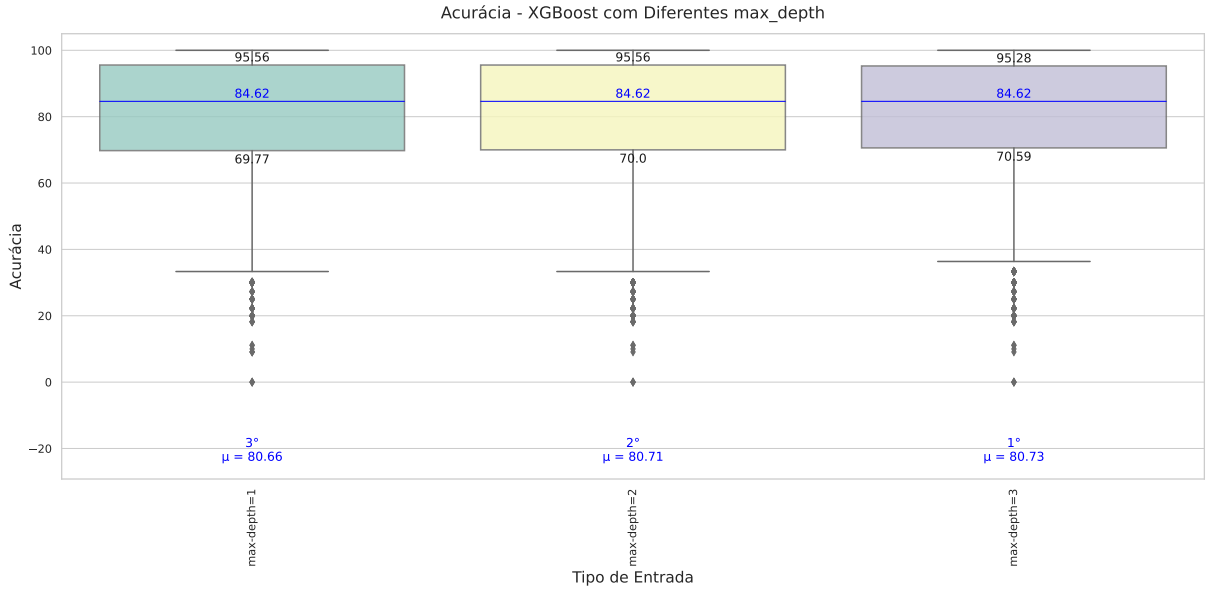

Figura 22 – Comparação de diferentes valores de max_depth no XGBoost (20 execuções e 31 problemas).

7.4.5 Resultados - Ponderação de Classes - SVM e XGBoost

A ponderação de classes (*class_weigth*) costuma ser usada quando existe um desbalanceamento no número de amostras entre cada classe (classes com muitas amostras e outras com poucas) e funciona penalizando mais as classificações erradas de classes com menor número de amostras.

Foi verificado se a ponderação de classes poderia aumentar a revocação média (média entre todas as classes de todos os problemas) ou, até mesmo, a média de acurácias. Os valores de média de acurácias foram muito similares: para o SVM, obtemos 87.13% sem *class_weight* e 87.1% com *class_weight*; para o XGBoost, 80.75% sem e 80.65% com *class_weight*. Os valores revocação média foram melhores com *class_weight*: para o SVM, obtemos 82.76% sem e 82.98% com *class_weight*; para XGBoost, 71.84% sem e 74.3% com *class_weight*.

Concluímos que, se a métrica que se está priorizando for a **revocação média, a ponderação de classes pode ser útil**, melhorando os resultados. Por outro lado, se a métrica priorizada for a **acurácia média, a não utilização de *class_weight* é melhor**. Nos próximos experimentos, como definimos a métrica para comparação dos métodos de Aprendizagem de Máquina como a acurácia, decidimos não empregá-la.

7.4.6 Resultados - Diferentes Formas de Entrada - SVM e XGBoost

Como explicado na Seção 7.3.1, foram realizados experimentos com cinco formas distintas de entrada (probes, genes, tfs, genes-probes e tfs-probes) com e sem filtragem com o teste de Kruskal-Wallis, totalizando 10 formas de entrada.

Para o SVM, considerando os 185 problemas e 20 execuções por problema e analisando todos os quatro kernels usados em conjunto ou analisando individualmente o kernel linear L2 (o que obteve melhor média de acurácias), a entrada em genes com filtro de kruskal (**genes-kruskal**) foi a que obteve maior média de acurácias, seguido de sondas associadas à genes com filtro de kruskal (genes-probes-kruskal) e sondas com filtro de kruskal (probes-kruskal). A Figura 23 apresenta os resultados das diferentes entradas para o SVM com kernel linear L2.

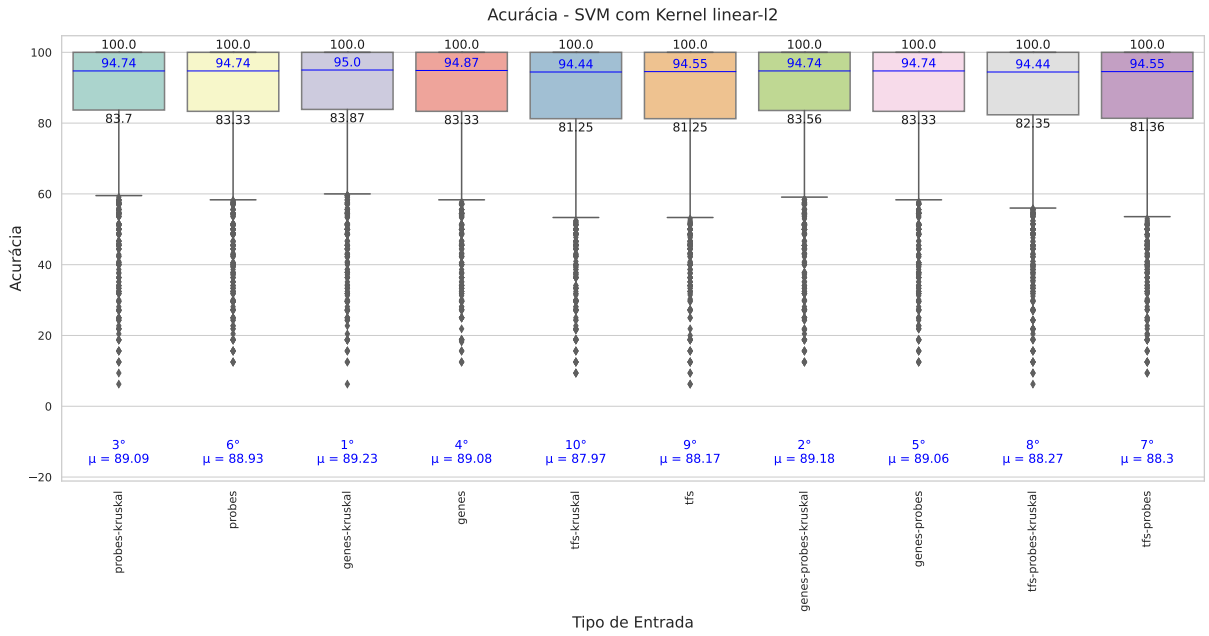

Figura 23 – Comparação de diferentes formas de entrada para o SVM Linear L2 (20 execuções e 185 problemas).

Para o XGBoost, considerando os 31 problemas e 20 execuções por problema e analisando todos os valores de *max_depth* em conjunto, a entrada em genes foi a que obteve maior média de acurácias, seguido de genes com filtro de kruskal (genes-kruskal) e, após, sondas associadas à genes com filtro de kruskal (genes-probes-kruskal).

Analisando individualmente os classificadores com $max\_depth = 3$ (o que obteve melhor média de acurácias), **genes-kruskal** aparece em primeiro lugar, seguido de genes e genes-probes-kruskal. A Figura 24 apresenta os resultados das diferentes entradas para o XGBoost com $max\_depth = 3$.

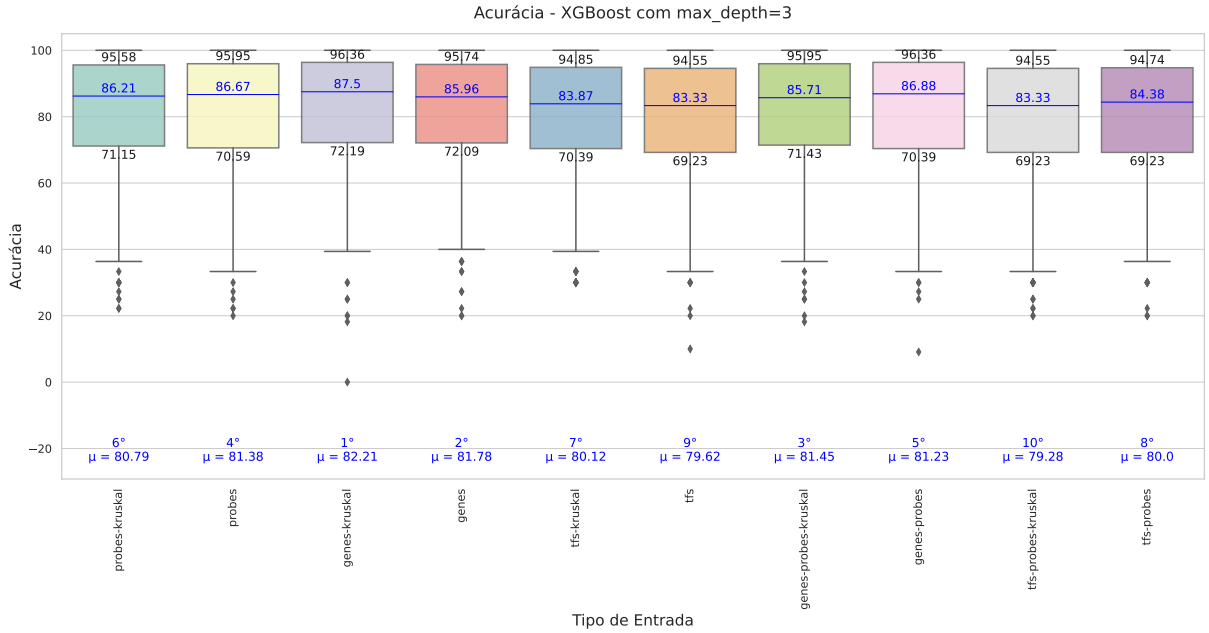

Figura 24 – Comparação de diferentes formas de entrada para o XGBoost com $max\_depth = 3$ (20 execuções e 31 problemas).

Entretanto, o melhor tipo de entrada varia de problema para problema. A Figura 25 demonstra a porcentagem de problemas na qual cada forma de entrada teve a melhor média de acurácia, usando o SVM com os quatro kernels. A ordem de preferência para decidir a melhor forma de entrada foi a média geral de todos problemas.

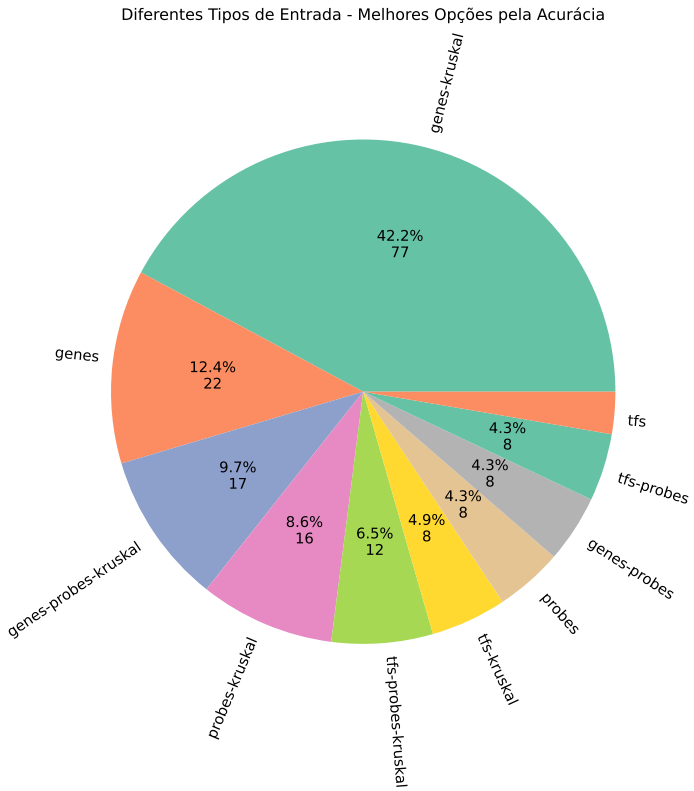

Figura 25 – Comparação de Melhores Formas de Entrada no SVM - proporção de problemas que tem a maior média de acurácia em cada tipo de entrada (20 execuções e 185 problemas).

Observando esses resultados, decidiu-se por adotar apenas a entrada na forma de **genes-kruskal nos próximos experimentos**.

7.4.7 Resultados - SVM x XGBoost

Comparando os métodos XGBoost com $max\_depth = 3$ e SVM com kernel linear L2, entrada na forma de genes-kruskal e sem ponderação *class_weight*, para 31 problemas e 20 execuções, o **SVM supera o XGBoost**, sendo as médias de acurácia de 85.48% para o SVM e 82.21% para o XGBoost.

Entretanto, o melhor método varia de problema para problema, a Figura 26 apresenta as proporções de problemas na qual cada método teve a melhor média de acurácia.

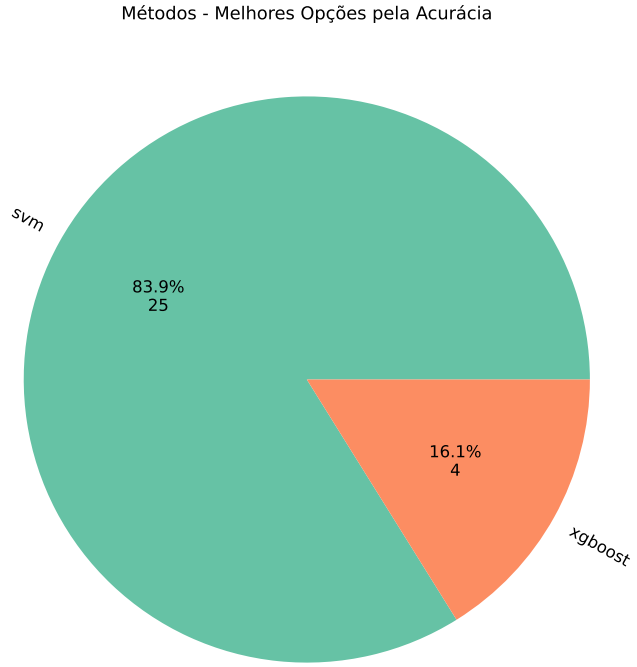

Figura 26 – Comparação SVM x XGBoost - proporção de problemas que tem a maior média de acurácia em cada tipo de método (20 execuções e 31 problemas).

A Figura 27 apresenta as proporções de problemas na qual cada opção experimentada obteve melhor média de acurácia (XGBoost variando max_depth, SVM variando o kernel, 10 formas de entrada, com e sem *class_weigth* (cw)) (ordem de preferência dada pela média de acurácia geral).

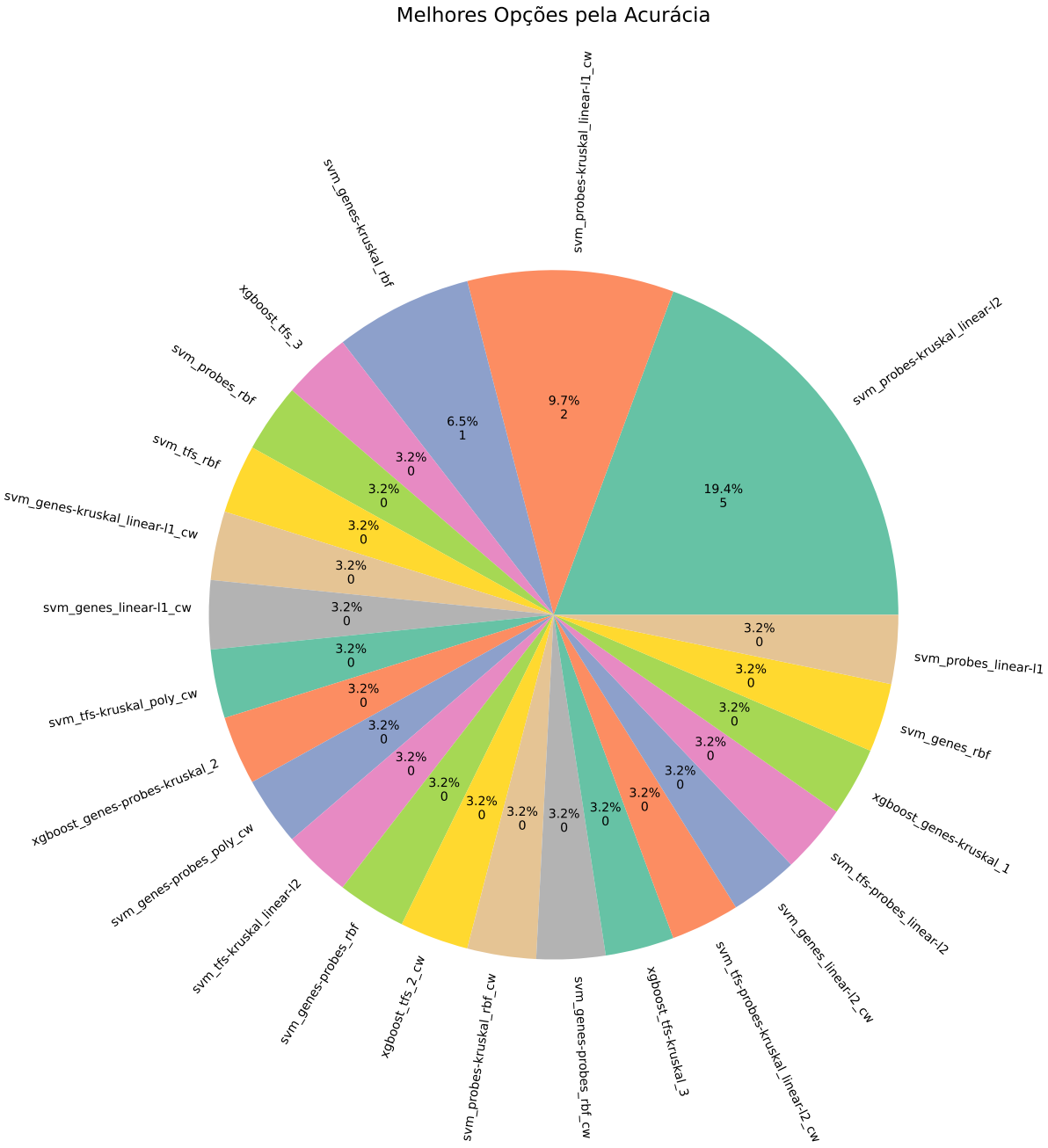

Figura 27 – Comparação entre Opções Experimentadas - proporção de problemas que tem a maior média de acurácia em cada opção experimentada - XGBoost variando max_depth, SVM variando o kernel, 10 formas de entrada, com e sem *class_weigth*(cw) (20 execuções e 31 problemas).

Concluímos que não existe um método e conjunto de opções que fornecem a melhor acurácia para qualquer *dataset* de expressão gênica. As melhores opções vão depender dos dados que se está usando, mas podemos definir um conjunto de métodos que a média geral entre diferentes problemas é a maior.

7.5 ResNet e Transcriptograma

A arquitetura de CNN **ResNet** com **18 e 34** camadas foram investigadas, utilizando como entrada dados em formato **1D** (convoluções sendo executadas em 1D) e **2D** (tradicionalmente usado nas CNNs que tratam imagens).

No formato 1D, a entrada é uma lista de valores de expressão gênica por amostra. Enquanto que, para a utilização do formato 2D, os dados de expressão foram transformados em uma imagem por amostra: a lista de valores de expressão gênica foi distribuída em zig zag numa matriz 2D, resultando, portanto, numa imagem em escala de cinza por amostra.

Uma vez que, para as CNNs, a ordem dos features na entrada da rede neural importa, experimentamos duas formas para estabelecer essa ordem: i) ordem **aleatória** ii) ordem dada pelo **Transcriptograma**.

Foram experimentados diferentes conjuntos de hiperparâmetros, a seguir listamos os valores testados.

- A probabilidade de dropout: 0, 0.1, 0.2, 0.5 e 0.7.
- Número de canais nas camadas internas: 8, 16, 32 e 64.
- Função de ativação: *ReLU*, *Leaky ReLU* com declive (*slope*) de 0.1, 0.2 e 0.3 e *RReLU (Randomized Leaky ReLU)* com valor de declive variando uniformemente entre 0.125 e 0.333... (valores *default* no PyTorch).
- Tamanho do kernel da primeira camada convolucional: 3, 5 e 7.
- Uso ou não da inicialização com zero na última camada de *Batch Normalization* de cada bloco residual.
- Número de épocas máximo: 100, 300 e 500.

Como condição de parada do treinamento adotamos o número de épocas máximo e valor mínimo de loss (computada no conjunto de treinamento). Observou-se duas formas de definir o conjunto de pesos final da rede neural:

- últimos pesos obtidos quando o treinamento é finalizado;
- pesos salvos com menor loss avaliada no conjunto de treinamento.

7.5.1 Datasets / Problemas

Como a execução de Redes Neurais Profundas em dados de alta dimensionalidade demandam elevado tempo de processamento, decidimos fazer uma seleção de datasets. Escolhemos, assim, apenas onze GSEs (datasets) de oito tecidos diferentes, conforme demonstrado na Seção 4.2.1. Nós definimos apenas uma separação de classes (que nomeamos no texto de 'problema') por GSE, ou seja, temos um problema para cada dataset.

7.5.2 Definição dos hiperparâmetros

Realizar *fine-tuning* dos hiperparâmetros em cada dataset demandaria alto custo de processamento (tempo) e, também dificultaria na comparação de métodos, pois teríamos um melhor conjunto de hiperparâmetros por dataset.

Dessa forma, para a escolha do melhor conjunto de hiperparâmetros realizamos o procedimento descrito na Seção 4.4.1, ou seja, foi feito um treinamento de modelo (apenas uma semente aleatória) por problema e conjunto de hiperparâmetros definido.

Após, selecionaram-se os melhores conjuntos de hiperparâmetros: o com maior média de acurácia entre todos 11 problemas e aqueles selecionados como melhores em cada problema individual. Então, foi executado o treinamento de modelos com outras sementes aleatórias, completando 20 execuções por problema e alcançando $11 * 20 = 220$ execuções por conjunto de hiperparâmetros.

O melhor conjunto de hiperparâmetros obtido foi:

- 0 de probabilidade de dropout (sem *dropout*);
- 64 canais nas camadas internas;
- função de ativação *Leaky ReLU* com declive (slope) 0.3;
- tamanho do kernel da primeira camada convolucional igual a 7;
- uso da inicialização com zero na última camada de *Batch Normalization* de cada bloco residual;
- número de épocas máximo igual a 500.

7.5.3 Transcriptograma

No Transcriptograma (método descrito no Capítulo 2.8), adotou-se: $\alpha = 1$; número de passos de Monte Carlo em cada temperatura no *Simulated Annealing* igual a 20; temperatura inicial igual a $6 * 10^5$; número de vezes que a temperatura é diminuída antes do encerramento do algoritmo igual a 200.

Na Figura 28 é apresentada a matriz de adjacências do grafo de interação entre genes antes da execução do Transcriptograma, na qual as posições dos nodos (genes) foram definidas aleatoriamente. Um ponto preto na coordenada (x, y) representa a existência de uma aresta entre os nodos da posição x (coluna) da matriz com o da posição y (linha). Uma aresta entre dois genes indica que se possui informação biológica sobre a existência de uma interação entre esses dois genes.

A Figura 29 mostra a matriz de adjacência ao final da execução do Transcriptograma. Como pode ser percebido, as arestas ficam agrupadas na diagonal da matriz, mostrando que nodos (genes) com funções similares (com mais arestas entre si) são posicionados em uma posição próxima uns dos outros (em outras palavras, o gene na

posição x apresenta maior número de arestas com o nodo na posição $x + 1$ do que com o nodo na posição $x + 1000$).

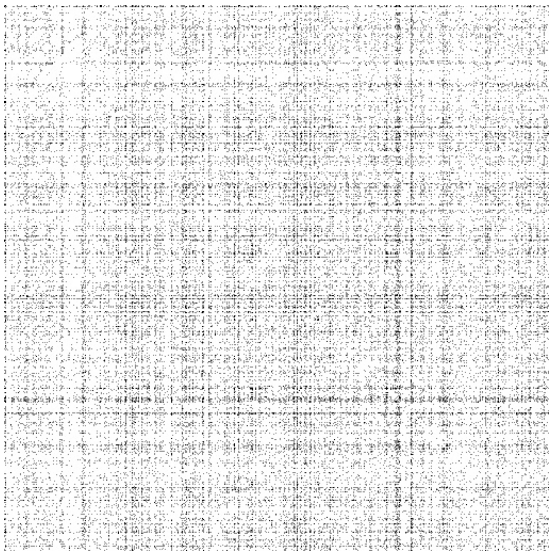

Figura 28 – Matriz de adjacências com as posições dos nodos (genes) inicializadas aleatoriamente. Pontos pretos representam a existência de uma aresta entre os nodos na posição x (coluna) com o na posição y (linha).

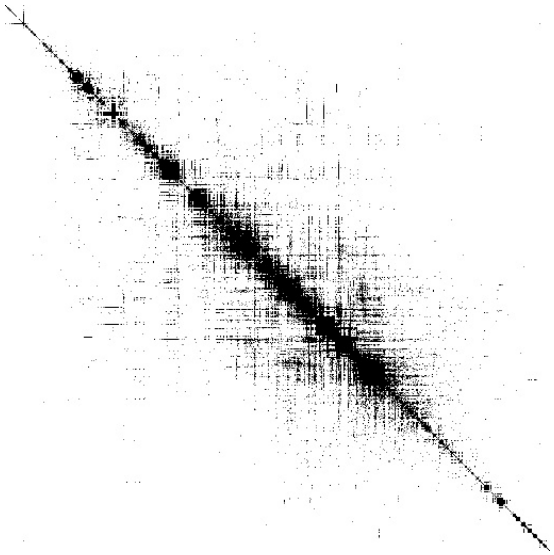

Figura 29 – Matriz de adjacência obtida pelo Transcriptograma - os nodos (genes) com mais arestas entre si (funções relacionadas) estão posicionados de forma próxima, conforme pode ser percebido pela diagonal preenchida.

O Transcriptograma fornece, portanto, uma ordem: posição para cada um dos genes em uma lista ordenada. Essa ordem é usada para montar as entradas da CNN - na ResNet 1 D a expressão gênica é disposta em linha e na 2D, em zigzag condensada numa imagem.

A rede de interação gênica (grafo) usada como entrada do Transcriptograma foi obtida no dataset BioGRID. Esse dataset não possui informação sobre todos os genes disponíveis nos datasets de *microarray* usados, pois não se tem informação biológica de interação gênica (com relativa confiança) sobre todos os genes existentes. Para esses genes (sem informação), o Transcriptograma não pode estabelecer uma posição. Então, nós escolhemos posicionar a expressão dos mesmos no final da lista, após os demais com informação biológica.

7.5.4 Ordem Aleatória x Ordem Fornecida pelo Transcriptograma

A Figura 30 e as Tabelas 3 e 4, a seguir, comparam os resultados obtidos com a ResNet 18 2D com os features de entrada disponibilizados em ordem aleatória e com os features organizados pela ordem estabelecida pelo transcriptograma (esses experimentos foram feitos com os melhores hiperparâmetros definidos na Seção 7.5.2, mas ainda sem a inicialização residual com zero das últimas batch normalization; foi usado 20 sementes aleatórias e 11 problemas).

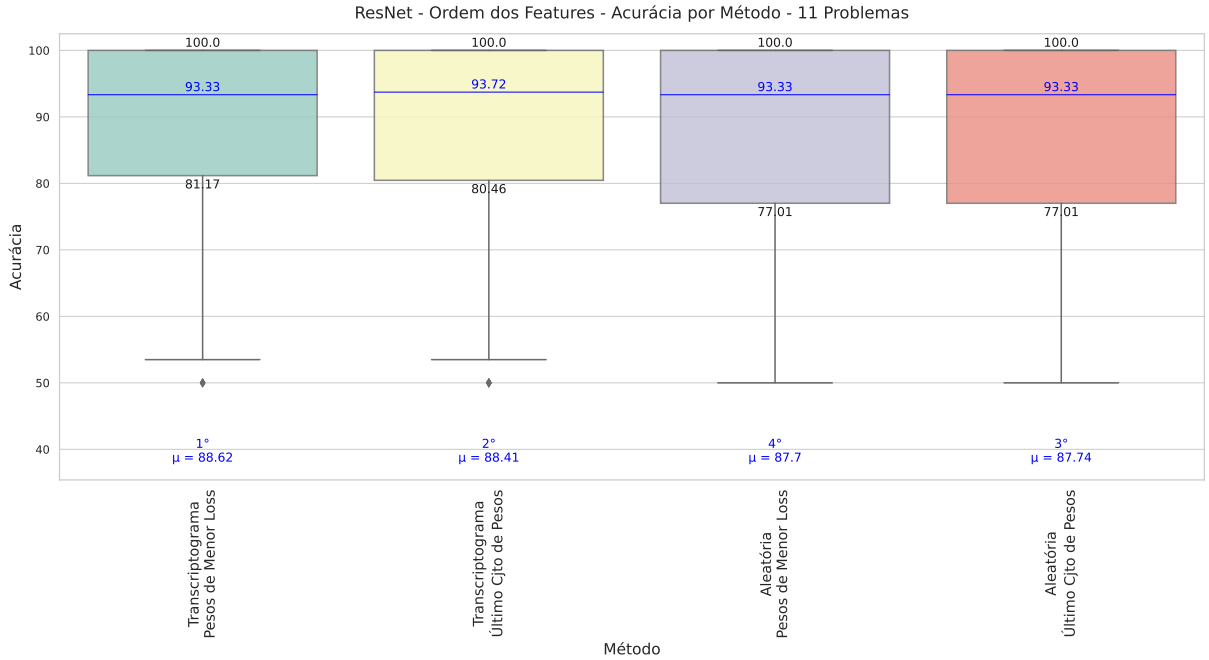

Figura 30 – ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - 11 problemas.

Tabela 3 – ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
Transcriptograma Pesos de Menor Loss	88.623	6.137	81.165	93.330	100.000	86.765	89.934	92.774	87.304	0.867
Transcriptograma Último Cjto de Pesos	88.408	6.084	80.460	93.725	100.000	84.891	88.830	92.774	87.149	0.865
Aleatória Pesos de Menor Loss	87.701	6.361	77.010	93.330	100.000	83.592	89.482	93.183	86.389	0.855
Aleatória Último Cjto de Pesos	87.741	6.373	77.010	93.330	100.000	83.566	88.240	93.209	86.529	0.856

Tabela 4 – ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - média de acurácia por método - cada problema mostrado separadamente.

	Transcriptograma Pesos de Menor Loss	Transcriptograma Último Cjto de Pesos	Aleatória Pesos de Menor Loss	Aleatória Último Cjto de Pesos
(1, 4)	99.455	99.455	99.455	99.455
(8, 2)	97.926	97.882	98.013	97.948
(13, 2)	82.500	77.500	81.250	80.000
(18, 1)	88.596	88.596	88.146	87.865
(42, 1)	77.701	77.816	75.230	75.345
(51, 4)	67.674	67.907	68.488	68.488
(59, 4)	79.231	79.231	75.000	75.000
(62, 2)	95.882	95.882	94.412	95.000
(71, 1)	95.000	96.667	94.167	94.167
(77, 5)	92.000	92.667	91.667	93.000
(82, 4)	98.889	98.889	98.889	98.889
Total	88.623	88.408	87.701	87.741

Após, realizamos um teste de hipóteses bootstrap pareado (ver Figura 31) e concluímos, com 95% de confiança, que a ordem dada pelo Transcriptograma tem média

de acurácias superior à ordem aleatória em pelo menos 0.06%.

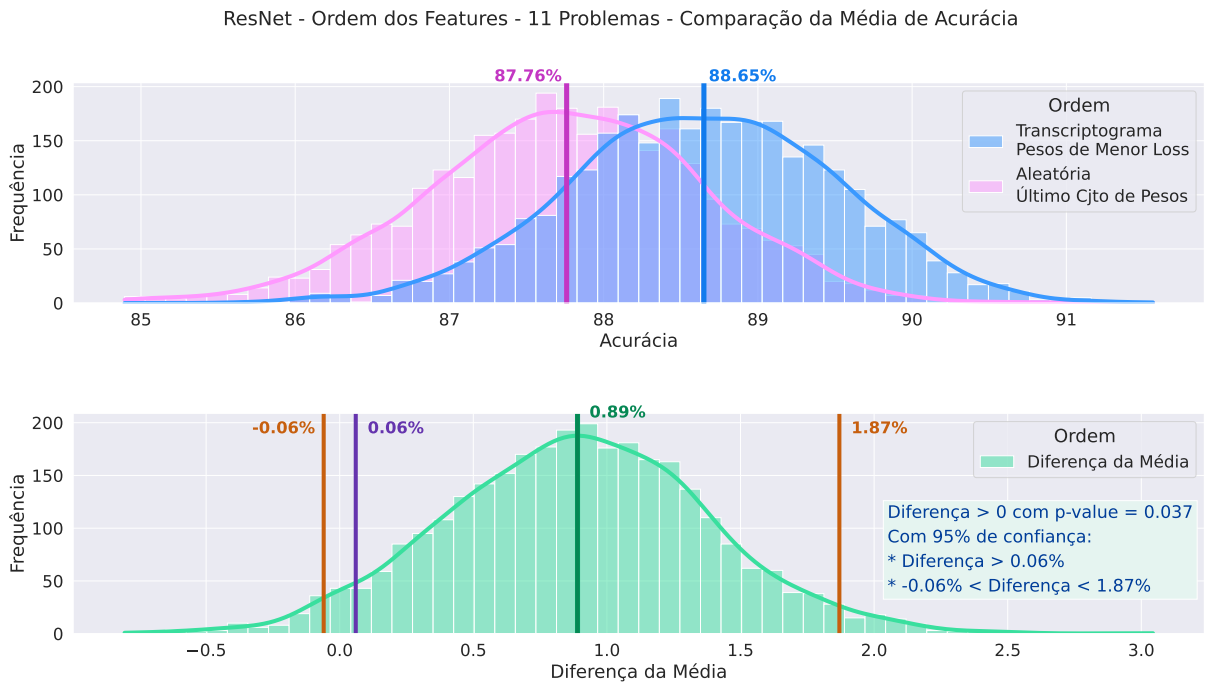

Figura 31 – ResNet 18 2D com Ordem do Transcriptograma x Ordem Aleatória - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

7.5.5 ResNet 1D x ResNet 2D

A Figura e Tabelas a seguir mostram a comparação entre a ResNet 18 recebendo entradas no formato 2D versus a mesma arquitetura recebendo entradas no formato 1D. Como pode ser percebido a ResNet com entrada 2D fornece melhores resultados (esses experimentos foram feitos com os melhores hiperparâmetros definidos na Seção 7.5.2, mas ainda sem a inicialização residual com zero das últimas batch normalization; foi usado 5 sementes aleatórias e 11 problemas).

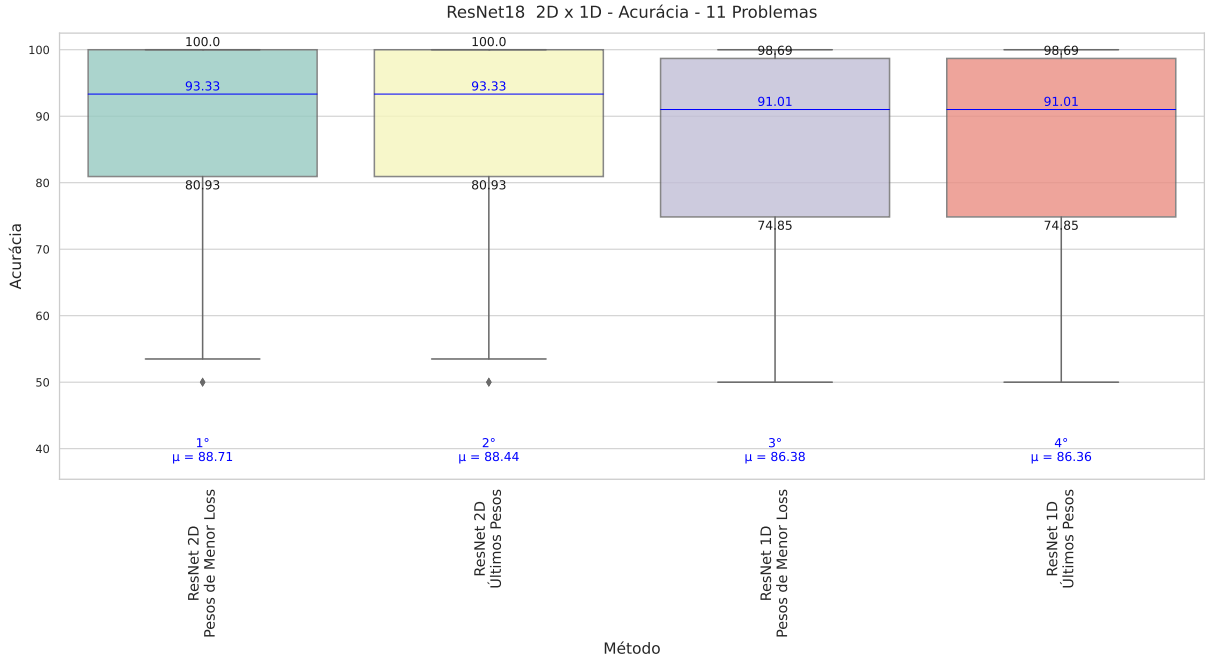

Figura 32 – ResNet 18 2D x 1D - 11 problemas.

Tabela 5 – ResNet 18 2D x 1D - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
ResNet 2D Pesos de Menor Loss	88.710	5.859	80.930	93.330	100.000	87.803	89.383	93.589	87.388	0.868
ResNet 2D Últimos Pesos	88.444	5.776	80.930	93.330	100.000	85.428	89.343	93.589	87.131	0.866
ResNet 1D Pesos de Menor Loss	86.384	5.863	74.855	91.010	98.690	83.500	86.491	90.132	84.352	0.836
ResNet 1D Últimos Pesos	86.364	5.842	74.855	91.010	98.690	83.500	86.491	90.132	84.331	0.836

Tabela 6 – ResNet 18 2D x 1D - média de acurácia por método - cada problema mostrado separadamente.

	ResNet 2D Pesos de Menor Loss	ResNet 2D Últimos Pesos	ResNet 1D Pesos de Menor Loss	ResNet 1D Últimos Pesos
(1, 4)	99.273	99.273	98.909	98.909
(8, 2)	97.729	97.642	97.817	97.817
(13, 2)	80.000	75.000	75.000	75.000
(18, 1)	88.989	88.989	89.438	89.213
(42, 1)	81.609	81.609	73.333	73.333
(51, 4)	69.767	69.767	69.302	69.302
(59, 4)	80.000	80.000	72.308	72.308
(62, 2)	100.000	98.824	94.118	94.118
(71, 1)	90.000	93.333	93.333	93.333
(77, 5)	89.333	89.333	88.000	88.000
(82, 4)	99.111	99.111	98.667	98.667
Total	88.710	88.444	86.384	86.364

Após, realizamos um teste de hipóteses bootstrap pareado (ver Figura 33) e con-

cluímos, com 95% de confiança, que a **ResNet 18 2D tem média de acurácias superior a ResNet 18 1D** em pelo menos 0.96%.

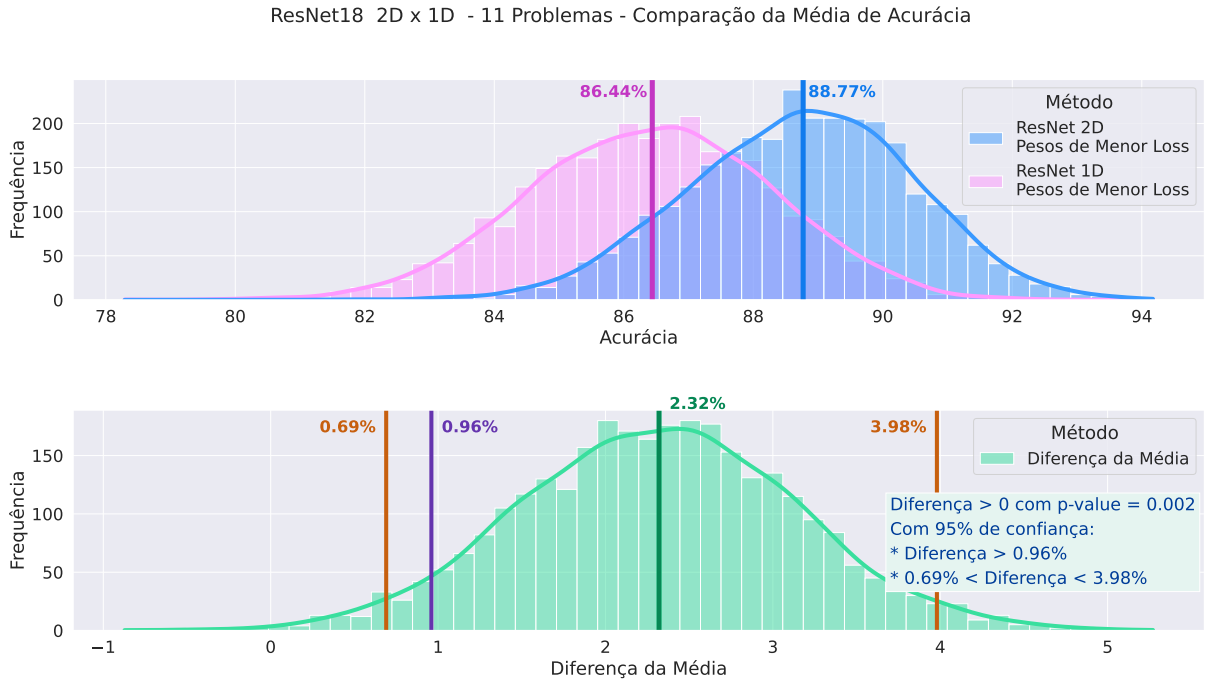

Figura 33 – ResNet 18 2D x 1D - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

7.5.6 ResNet 18 x ResNet 34

Após termos definido o melhor conjunto de hiperparâmetros para a ResNet 18 2D, utilizamos esses mesmos hiperparâmetros, mas aumentamos o número de camadas da rede de 18 para 34 - treinamos 20 modelos para cada um dos 11 problemas da ResNet 34 e comparamos com os resultados com os obtidos pela ResNet 18. A Figura 34 e as Tabelas 7 e 8 resumem os resultados. Como podemos perceber a **ResNet 18 obteve melhores resultados do que a ResNet 34**, então, nos próximos experimentos adotamos essa arquitetura.

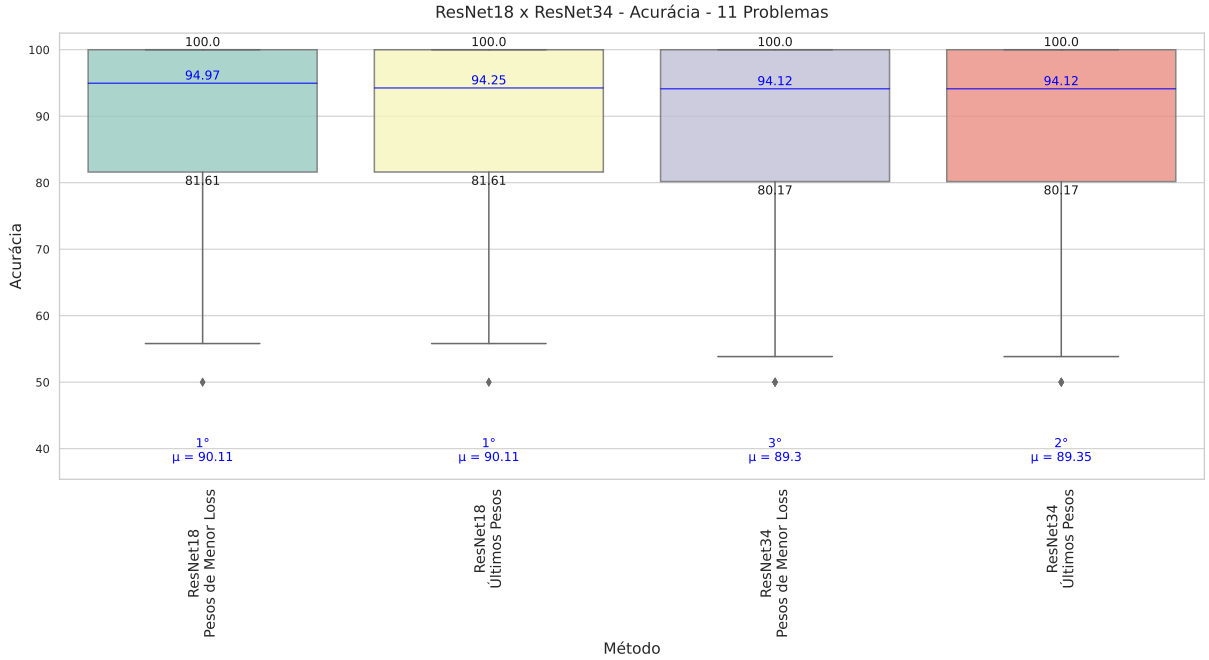

Figura 34 – ResNet 18 x ResNet 34 - 11 problemas.

Tabela 7 – ResNet 18 x ResNet 34 - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
ResNet18 Pesos de Menor Loss	90.105	4.805	81.610	94.970	100.000	86.941	90.253	93.802	89.020	0.885
ResNet18 Últimos Pesos	90.106	4.806	81.610	94.250	100.000	87.019	90.379	93.812	89.029	0.885
ResNet34 Pesos de Menor Loss	89.299	4.868	80.172	94.120	100.000	86.467	89.319	93.800	88.199	0.875
ResNet34 Últimos Pesos	89.355	4.857	80.172	94.120	100.000	86.467	89.623	93.800	88.302	0.876

Tabela 8 – ResNet18 e ResNet34 - média de acurácia por método - cada problema mostrado separadamente.

	ResNet18 Pesos de Menor Loss	ResNet18 Últimos Pesos	ResNet34 Pesos de Menor Loss	ResNet34 Últimos Pesos
(1, 4)	99.455	99.455	99.455	99.455
(8, 2)	98.362	98.384	98.188	98.253
(13, 2)	83.750	83.750	76.250	76.250
(18, 1)	88.708	88.764	89.045	89.045
(42, 1)	79.713	79.828	79.138	79.253
(51, 4)	72.209	72.326	71.279	71.047
(59, 4)	78.462	78.462	77.692	77.692
(62, 2)	97.941	97.647	97.353	97.353
(71, 1)	98.333	98.333	100.000	100.000
(77, 5)	94.667	94.667	94.333	95.000
(82, 4)	99.556	99.556	99.556	99.556
Total	90.105	90.106	89.299	89.355

7.6 MLP

Nossa implementação de rede neural MLP emprega *batch normalization* e *dropout* entre as camadas escondidas. Para o estabelecimento dos melhores conjuntos de hiperparâmetros, primeiramente, fixou-se o número de camadas escondidas em 3 e variaram-se os seguintes hiperparâmetros:

- número de neurônios das camadas escondidas: 8, 16, 32 e 64;
- probabilidade de *dropout*: 0.05, 0.1, 0.5, 0.7 e 0.9;
- função de ativação: *Leaky ReLU* com declive (*slope*) de 0.1, 0.2 e 0.3 e RReLU (Randomized Leaky ReLU) com valor de declive variando uniformemente entre 0.125 e 0.333... (valores *default* no PyTorch).

O número máximo de épocas foi fixado em 500. Foram realizadas, para cada um dos conjuntos de hiperparâmetros, 10 execuções (com 10 sementes aleatórias) para cada um dos 11 problemas.

Os melhores conjuntos de hiperparâmetros foram, então, selecionados (aquele com maior média de acurácia e o melhor para cada um dos 11 problemas). Então, utilizou-se os hiperparâmetros selecionados, mas aumentando o número de camadas escondidas para 4 e realizaram-se mais 10 novas execuções para cada um dos conjuntos de hiperparâmetros com 4 camadas escondidas.

Os resultados do melhor conjunto de hiperparâmetros com 3 camadas escondidas e com 4 camadas são demonstrados na Figura 35 e Tabelas 9 e 10 a seguir (para 10 sementes aleatórias).

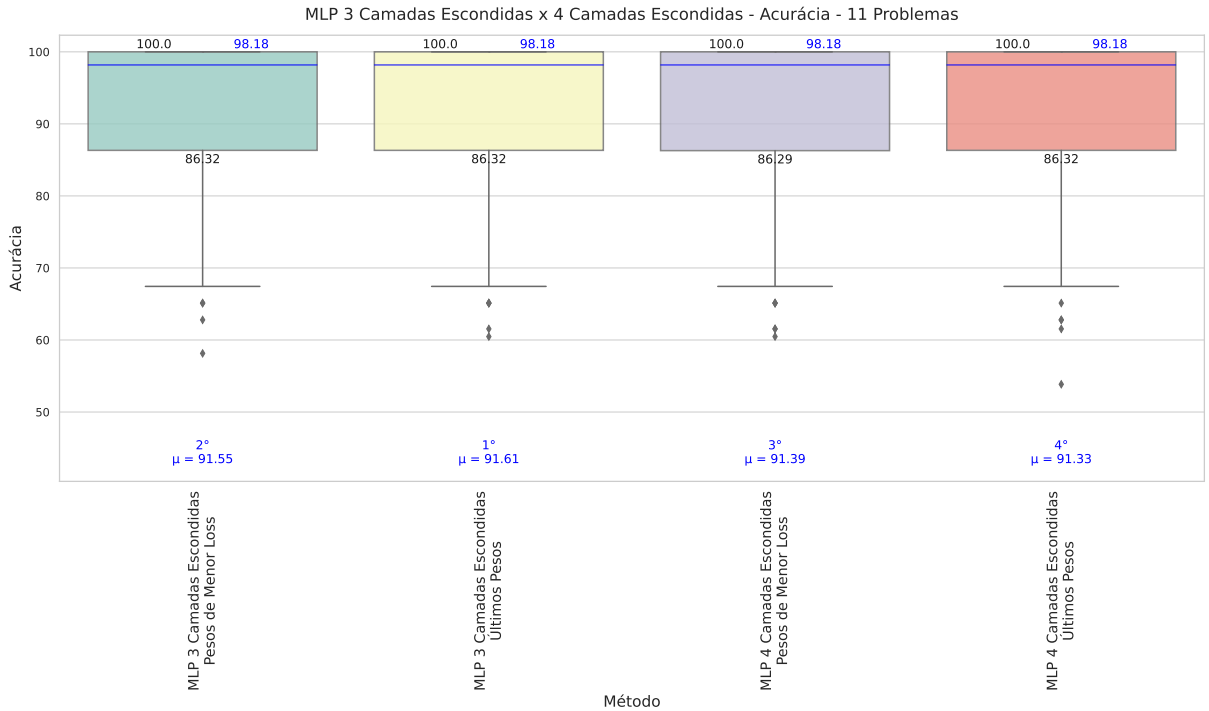

Figura 35 – MLP com 3 camadas escondidas x 4 camadas escondidas - 11 problemas.

Tabela 9 – MLP com 3 camadas escondidas x 4 camadas escondidas - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
MLP 3 Camadas Escondidas Pesos de Menor Loss	91.550	3.723	86.325	98.180	100.000	88.916	92.574	93.875	90.819	0.903
MLP 3 Camadas Escondidas Últimos Pesos	91.606	3.970	86.325	98.180	100.000	88.837	93.132	93.956	91.032	0.905
MLP 4 Camadas Escondidas Pesos de Menor Loss	91.393	3.949	86.287	98.180	100.000	88.709	92.522	94.320	90.742	0.903
MLP 4 Camadas Escondidas Últimos Pesos	91.329	4.055	86.325	98.180	100.000	88.762	92.522	94.320	90.784	0.903

Tabela 10 – MLP com 3 camadas escondidas x 4 camadas escondidas - média de acurácia - cada problema mostrado separadamente.

	MLP 3 Camadas Escondidas Pesos de Menor Loss	MLP 3 Camadas Escondidas Últimos Pesos	MLP 4 Camadas Escondidas Pesos de Menor Loss	MLP 4 Camadas Escondidas Últimos Pesos
(1, 4)	99.091	99.091	99.091	99.091
(8, 2)	98.472	98.472	98.428	98.384
(13, 2)	92.500	92.500	92.500	92.500
(18, 1)	89.663	90.112	90.112	90.225
(42, 1)	85.632	85.172	85.402	85.402
(51, 4)	68.140	69.535	69.535	69.535
(59, 4)	81.538	80.769	78.462	77.692
(62, 2)	98.235	98.235	98.235	98.235
(71, 1)	100.000	100.000	100.000	100.000
(77, 5)	94.000	94.000	94.000	94.000
(82, 4)	99.778	99.778	99.556	99.556
Total	91.550	91.606	91.393	91.329

A média de acurácias para a MLP de 3 camadas escondidas foi levemente maior do que a com 4 camadas; entretanto, analisando o teste de hipóteses (Figura abaixo 36), conclui-se que não podemos afirmar que exista diferença de média de acurácias entre 3 e 4 camadas com $pvalue < 0.05$.

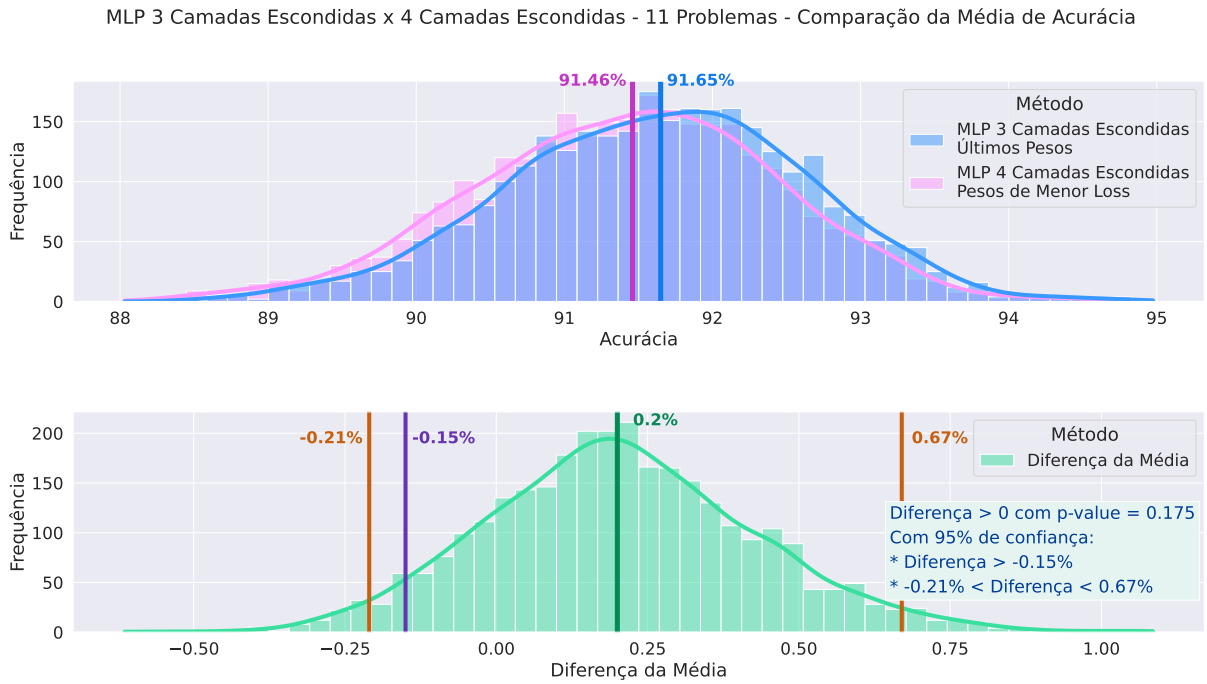

Figura 36 – MLP com 3 camadas escondidas x 4 camadas escondidas - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

Os melhores conjuntos de hiperparâmetros, considerando ambos números de camadas escondidas, foram selecionados e executou-se para mais sementes aleatórias até completar 20 sementes por conjunto de hiperparâmetros.

Então, definiu-se o melhor conjunto de hiperparâmetros para a comparação da MLP com outras abordagens:

- 3 camadas escondidas;
- 64 neurônios nas camadas escondidas;
- função de ativação *Leaky ReLU* com declive de 0.2;
- probabilidade de *dropout* igual a 0.7.

A média de acurácias considerando todos 11 problemas e 20 sementes aleatórias, para o melhor conjunto de hiperparâmetros foi **91.543%** para o conjunto de pesos salvos pela menor *loss* (avaliada no conjunto treinamento) e **91.571%** para os últimos pesos obtidos ao final do treinamento. A Figura 37 e Tabelas 11 e 12 a seguir comparam os resultados obtidos com os da ResNet 18.

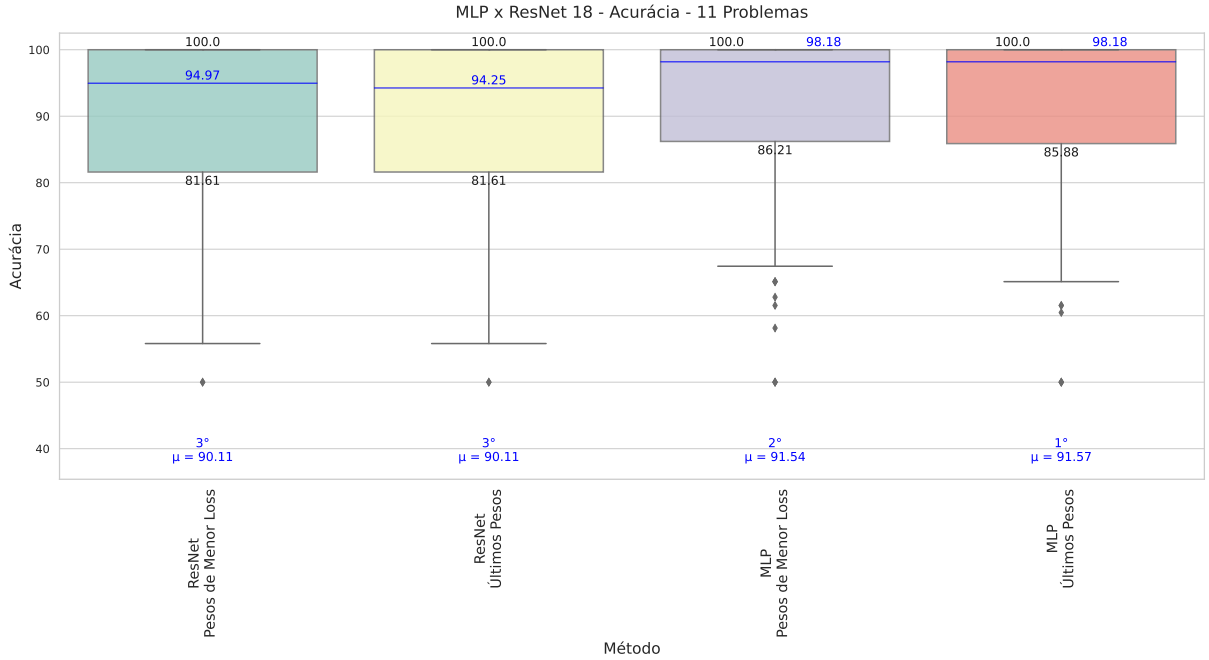

Figura 37 – MLP x ResNet 18 - 11 problemas.

Tabela 11 – MLP x ResNet 18 - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
ResNet Pesos de Menor Loss	90.105	4.805	81.610	94.970	100.000	86.941	90.253	93.802	89.020	0.885
ResNet Últimos Pesos	90.106	4.806	81.610	94.250	100.000	87.019	90.379	93.812	89.029	0.885
MLP Pesos de Menor Loss	91.543	4.624	86.210	98.180	100.000	88.747	93.521	94.629	90.880	0.904
MLP Últimos Pesos	91.571	4.653	85.885	98.180	100.000	88.849	93.519	94.655	90.994	0.904

Tabela 12 – MLP x ResNet 18 - média de acurácia - cada problema mostrado separadamente.

	ResNet Pesos de Menor Loss	ResNet Últimos Pesos	MLP Pesos de Menor Loss	MLP Últimos Pesos
(1, 4)	99.455	99.455	99.455	99.455
(8, 2)	98.362	98.384	98.231	98.231
(13, 2)	83.750	83.750	86.250	86.250
(18, 1)	88.708	88.764	90.000	90.337
(42, 1)	79.713	79.828	84.885	84.655
(51, 4)	72.209	72.326	71.860	72.442
(59, 4)	78.462	78.462	81.538	81.154
(62, 2)	97.941	97.647	98.529	98.529
(71, 1)	98.333	98.333	100.000	100.000
(77, 5)	94.667	94.667	96.333	96.333
(82, 4)	99.556	99.556	99.889	99.889
Total	90.105	90.106	91.543	91.571

Aplicando o teste de hipóteses bootstrap pareado para diferença das médias entre MLP e a ResNet 18 (resultados demonstrados na Figura 38), podemos afirmar, com

95% de confiança, que a média de acurácias da **MLP é superior** a da ResNet 18 em pelo menos 0.75%.

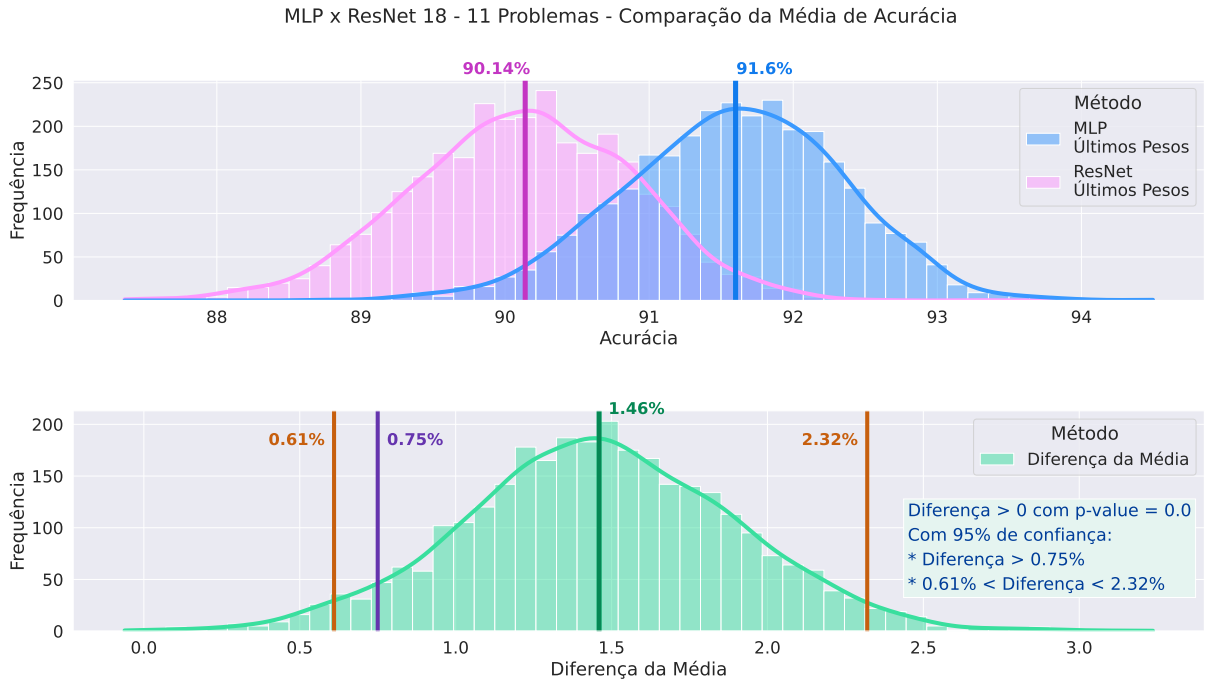

Figura 38 – MLP x ResNet 18 - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

7.7 ResNet 18 Pré-treinada

Os experimentos com ResNet 18 Pré-Treinada nos permitiu investigar *Transfer Learning* a partir dos modelos pré-treinados na ImageNet. Esse experimento consistiu em uma tentativa de, usando os pesos obtidos com a ImageNet (dataset com muitas imagens) pudesse evitar o sobreajuste (*overfitting*) dos modelos, uma vez que se percebeu que os modelos de ResNet 18 tradicional estavam alcançando acurácia de 100% no treino e valores menores de acurácia no teste do que o método baseline SVM.

Para os hiperparâmetros, adotou-se o mesmo conjunto dos modelos disponíveis online treinados com a ImegeNet (LAB, 2018a). Como o dataset de expressão gênica é muito diferente da ImageNet, as camadas convolucionais não foram congeladas. A última camada *fully connected* do modelo pré-treinado foi removida e adicionou-se um conjunto de nr_novas_layers novas camadas *fully connected* no topo das camadas convolucionais da ResNet pré-treinada.

Como hiperparâmetros dessas novas camadas, adotaram-se os melhores hiperparâmetros da MLP (apresentada na Seção anterior 7.6), ou seja, adotou-se:

- função de ativação *Leaky ReLU* com declive (*slope*) de 0.2;

- probabilidade de *dropout* igual a 0.7;
- 64 neurônios.

O número máximo de épocas foi fixado em 500.

Para definição do nr_novas_layers adotou-se o procedimento descrito na Seção 4.4.1. Ou seja, realizou-se uma execução (uma semente aleatória) do modelo por cada um dos 11 problemas para cada um dos conjuntos de hiperparâmetros. Experimentaram-se os seguintes valores para nr_novas_layers: 1, 3, 4 e 6.

A média de acurácias considerando todos 11 problemas e 20 sementes aleatórias foi **87.172%** para o conjunto de pesos salvos pela menor *loss* (avaliada no conjunto treinamento) e **87.434%** para os últimos pesos obtidos ao final do treinamento.

Analisando o teste de hipóteses bootstrap pareado para diferença das médias entre a ResNet 18 (resultados demonstrados na Figura 39), conclui-se, com 95% de confiança, que a **ResNet 18 tradicional** apresenta média de acurácias **superior** em pelo menos 1.73%.

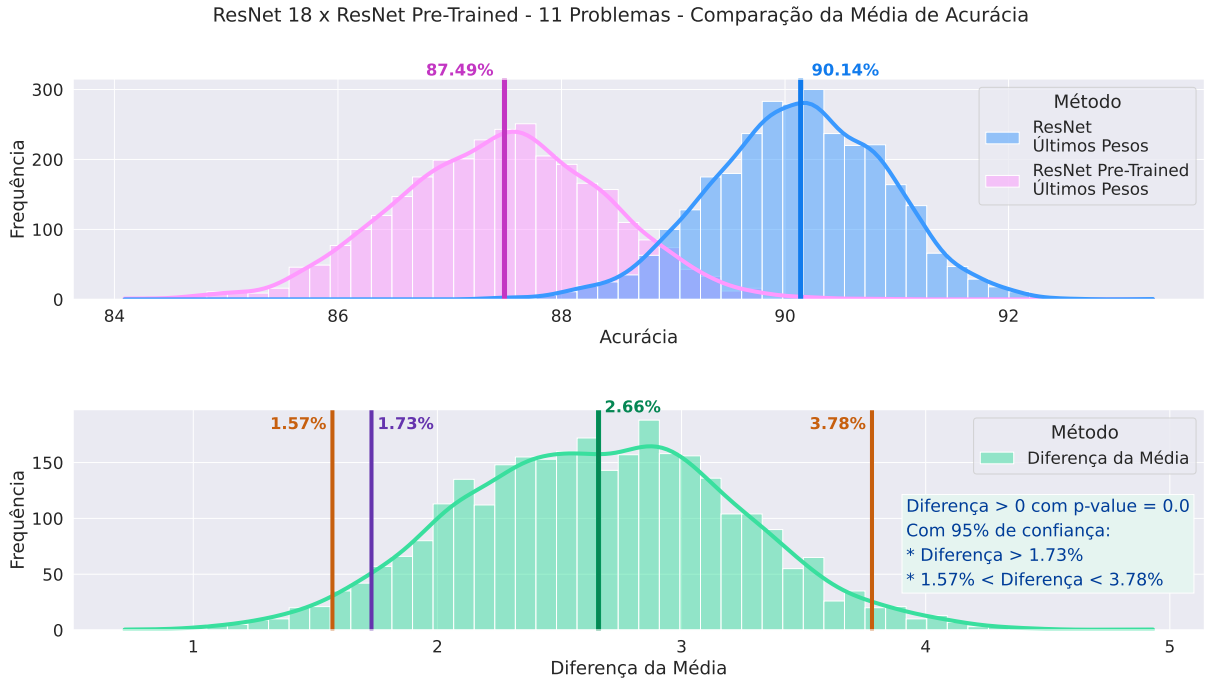

Figura 39 – ResNet 18 x ResNet 18 Pré-treinada - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

7.8 Redes Neurais Profundas com Mecanismo de Atenção

Três abordagens de Redes Neurais Profundas com Mecanismo de Atenção foram analisadas através experimentos no conjunto de **11 problemas** (um problema de classificação por cada um dos datasets selecionados). As abordagens estudadas são listadas a seguir.

- **ResNet 18 Pre-Activated Simple Self-Attention** (ZHANG et al., 2019) - ResNet de 18 camadas com entrada no formato 2D e ordem dos features estabelecida pelo Transcriptograma. Essa rede neural possui pré-ativação nos blocos residuais e um mecanismo de atenção ao final de cada bloco residual.

- **Vision Transformer** (DOSOVITSKIY et al., 2020) - rede neural com camadas *fully connected* (sem camadas convolucionais), composta de blocos residuais com mecanismos de atenção.

- **Hybrid Vision Transformer** (DOSOVITSKIY et al., 2020) - rede neural *Vision Transformer* que recebe, como entrada, a representação embutida (*embedded representation*) obtida por uma ResNet 18 pré-treinada (com entrada no formato 2D e ordem dos features estabelecida pelo Transcriptograma). Ou seja, a última camada (a camada *fully connected*) da ResNet 18 é excluída e a representação obtida nos diferentes canais da última camada convolucional é empregada como features para um *Vision Transformer*.

Os resultados obtidos por cada uma dessas abordagens são resumidos na Figura 40 e Tabelas 13 e 14, sendo comparados com os resultados alcançados pela ResNet 18 tradicional. A Figura 40 e Tabela 13 mostram o resultado considerando todos os 11 problemas em conjunto e a Tabela 14 apresenta os resultados obtidos em cada um dos 11 problemas separadamente.

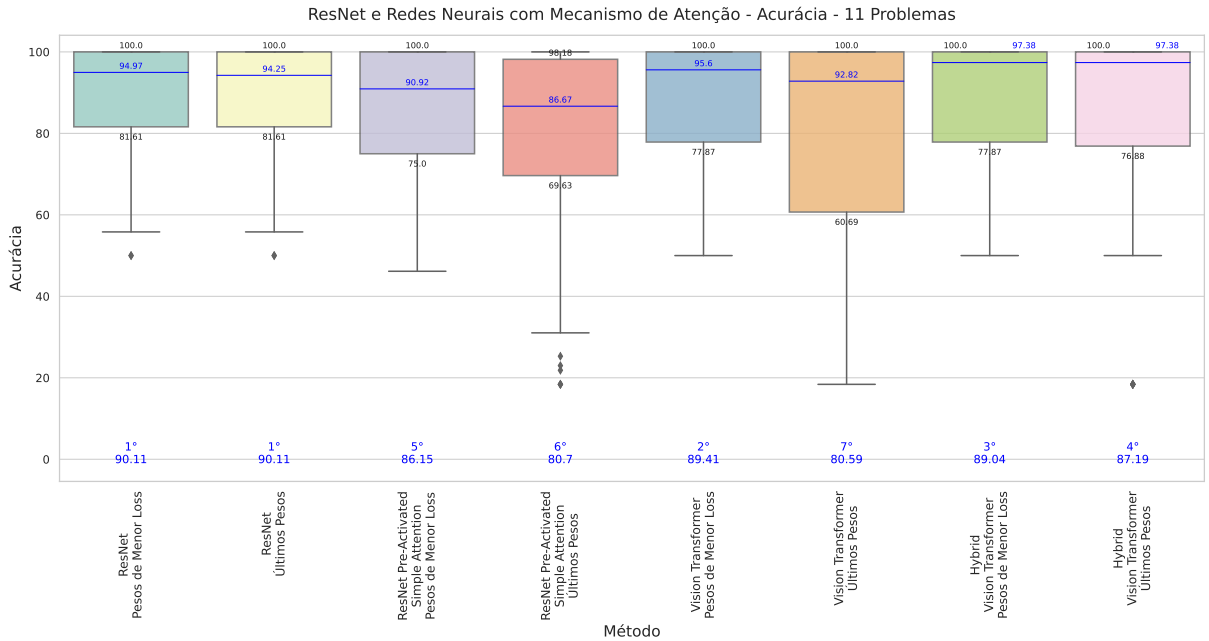

Figura 40 – ResNet 18 e redes neurais com Mecanismo de Atenção - 11 problemas.

Tabela 13 – ResNet e Redes Neurais com Mecanismo de Atenção - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
ResNet Pesos de Menor Loss	90.105	4.805	81.610	94.970	100.000	86.941	90.253	93.802	89.020	0.885
ResNet Últimos Pesos	90.106	4.806	81.610	94.250	100.000	87.019	90.379	93.812	89.029	0.885
ResNet Pre-Activated Simple Attention Pesos de Menor Loss	86.153	6.997	75.000	90.920	100.000	81.888	88.515	90.914	83.738	0.832
ResNet Pre-Activated Simple Attention Últimos Pesos	80.698	11.672	69.635	86.670	98.180	71.961	82.212	90.448	76.465	0.752
Vision Transformer Pesos de Menor Loss	89.406	5.438	77.873	95.595	100.000	86.406	91.040	93.160	88.095	0.873
Vision Transformer Últimos Pesos	80.592	12.045	60.690	92.820	100.000	72.222	80.761	91.125	77.955	0.745
Hybrid Vision Transformer Pesos de Menor Loss	89.039	5.501	77.873	97.380	100.000	86.014	90.503	92.978	87.974	0.871
Hybrid Vision Transformer Últimos Pesos	87.195	7.103	76.875	97.380	100.000	81.234	89.824	92.351	86.057	0.849

Tabela 14 – ResNet e Redes Neurais com Mecanismo de Atenção - média de acurácia por método - cada problema mostrado separadamente.

	ResNet Pesos de Menor Loss	ResNet Últimos Pesos	ResNet Pre-Activated Simple Attention Pesos de Menor Loss	ResNet Pre-Activated Simple Attention Últimos Pesos	Vision Transformer Pesos de Menor Loss	Vision Transformer Últimos Pesos	Hybrid Vision Transformer Pesos de Menor Loss	Hybrid Vision Transformer Últimos Pesos
(1, 4)	99.455	99.455	99.364	98.818	99.545	99.545	99.455	99.455
(8, 2)	98.362	98.384	96.725	63.166	97.293	61.965	97.904	97.904
(13, 2)	83.750	83.750	83.750	83.750	86.250	86.250	81.250	81.250
(18, 1)	88.708	88.764	87.472	86.854	88.876	67.416	87.472	87.472
(42, 1)	79.713	79.828	72.126	54.943	74.425	50.460	74.023	53.736
(51, 4)	72.209	72.326	69.186	67.326	69.651	57.209	71.163	71.163
(59, 4)	78.462	78.462	68.846	65.385	78.077	74.615	76.538	76.538
(62, 2)	97.941	97.647	90.882	90.882	98.235	97.941	98.235	98.235
(71, 1)	98.333	98.333	91.667	91.667	98.333	98.333	99.167	99.167
(77, 5)	94.667	94.667	89.333	88.333	93.000	93.000	95.333	95.333
(82, 4)	99.556	99.556	98.333	96.556	99.778	99.778	98.889	98.889
Total	90.105	90.106	86.153	80.698	89.406	80.592	89.039	87.195

As subseções a seguir explicarão, em maiores detalhes, os resultados de cada uma das abordagens com Mecanismo de Atenção.

7.8.1 ResNet 18 Pre-Activated Simple Self-Attention

Os hiperparâmetros usados nessa abordagem de rede neural foram os mesmos que obtiveram o melhor valor de acurácia para o modelo de ResNet 18 tradicional (sem ser pré-ativada e sem possuir mecanismo de atenção). Ou seja, adotou-se:

- formato de entrada 2D (imagem) com ordem dos *features* dada pelo Transcriptograma;

- função de ativação *Leaky ReLU* com declive (*slope*) de 0.3;
- probabilidade de *dropout* nula (sem *dropout*);
- 64 canais nas camadas convolucionais;
- *kernel* da primeira camada convolucional de tamanho 7 e nas demais de tamanho 3;
- inicialização com zero na última camada de *Batch Normalization* de cada bloco residual;
- número máximo de épocas igual a 500.

A média de acurácia dos 11 problemas e 20 execuções por problema (cada execução com uma semente aleatória diferente, usada na inicialização dos pesos do modelo e na divisão dos datasets em treino e teste), ou seja, de 220 valores de acurácia foi de **86.153%** para os pesos (parâmetros) do modelo selecionados pela menor *loss* avaliada no conjunto de treinamento e **80.698%** para o conjunto de pesos obtidos na última época do treinamento.

Esses valores de acurácia média são menores dos obtidos com uma ResNet 18 com o mesmo conjunto de hiperparâmetros e treinada com as mesmas divisões dos datasets em treino e teste, como pode ser percebido através da Figura 40 e da Tabela 13 (apresentadas anteriormente).

Analisando a Tabela 14, exposta anteriormente, é possível perceber que a ResNet 18 Pre-Activated Simple Self-Attention apresenta resultados piores que a ResNet 18 tradicional em todos os problemas, com exceção do problema $(13, 2)$, no qual alcançou o mesmo resultado do que a ResNet 18.

A Figura 41 a seguir demonstra os resultados do teste de hipóteses bootstrap pareado para diferença de médias entre a Resnet 18 tradicional e a Pre-Activated Simple Self-Attention. O teste mostra que com 95% de confiança que a média de acurácias da **ResNet 18 tradicional é superior** em pelo menos 2.93%. Portanto, para dados de expressão gênica, a ResNet 18 Pre-Activated Simple Self-Attention não parece ser uma boa opção.

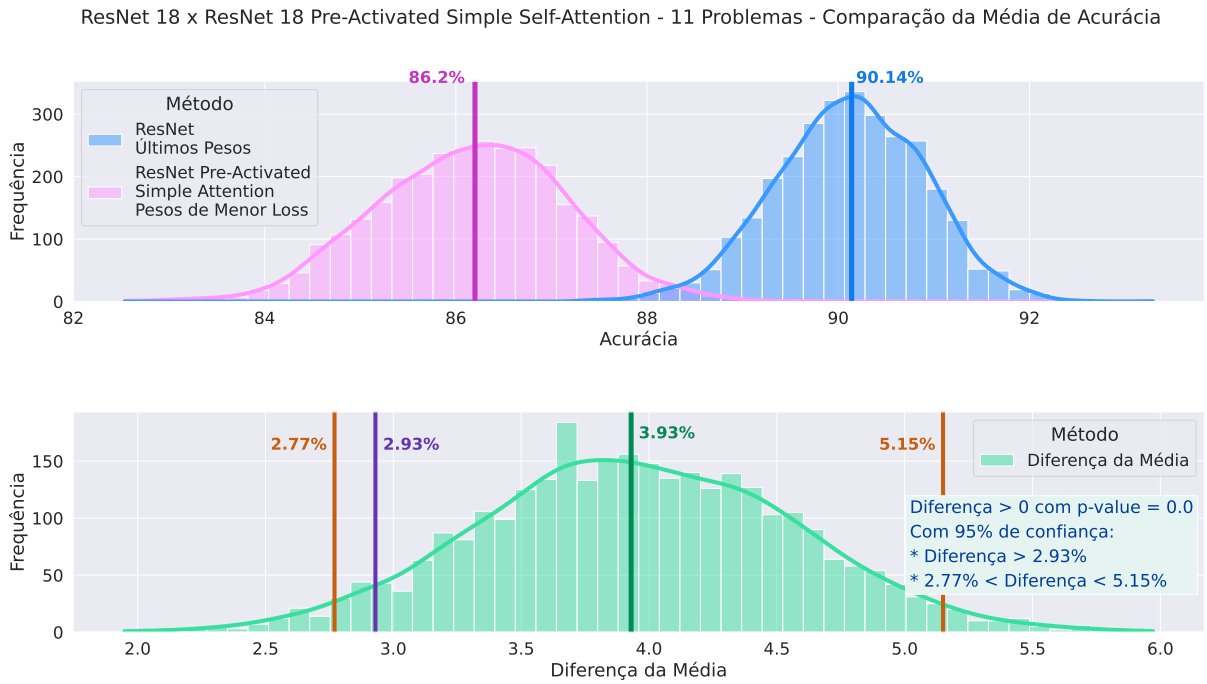

Figura 41 – ResNet 18 x ResNet 18 Pre-Activated Simple Self-Attention - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

Talvez, fazendo *fine-tuning* de hiperparâmetros seja possível melhorar um pouco as acurácias obtidas, mas ainda assim, os resultados que obtivemos indicam que esse modelo de Aprendizagem de Máquina não é uma das melhores soluções a serem adotadas.

7.8.2 Vision Transformer

Para o estabelecimento dos melhores conjuntos de hiperparâmetros para a rede neural Vision Transformer adotou se o procedimento descrito na Seção 4.4.1. Ou seja, realizou-se uma execução (uma semente aleatória) do modelo por cada um dos 11 problemas para cada um dos conjuntos de hiperparâmetros.

Após, para os melhores conjuntos de hiperparâmetros, realizaram-se mais execuções, completando 20 execuções por problema ($20 * 11 = 220$ valores de acurácia) e, então, selecionou-se o melhor conjunto de hiperparâmetros para a comparação desse modelo com os demais.

Os hiperparâmetros analisados foram:

- número de cabeças: 8 e 16;
- número de blocos de *transformer*: 4, 6 e 12;
- número de neurônios das camadas escondidas (tamanho da representação embutida *embedded representation*) 32 e 64:
- tamanho de *patch_size* (quadrado da porção da imagem tomada por vez): 10 e 20.

O número máximo de épocas adotado foi fixado em 500. A entrada usada foi no formato 2D com ordem dos *features* dada pelo Transcriptograma. E adotou-se probabilidade de *dropout* de 0.1. A função de ativação empregada foi a identidade (ou seja, nenhuma operação), seguindo o adotado no trabalho de (DOSOVITSKIY et al., 2020).

O melhor conjunto de hiperparâmetros obtido foi:

- 8 cabeças;
- 6 blocos de *transformer*;
- 64 neurônios nas camadas escondidas;
- tamanho de *patch_size* igual a 20.

A média de acurácias considerando todos 11 problemas e 20 sementes aleatórias foi **89.406%** para o conjunto de pesos salvos pela menor *loss* (avaliada no conjunto treinamento) e **80.592%** para os últimos pesos obtidos ao final do treinamento. Portanto, nesta abordagem, o conjunto de pesos salvos pela menor loss apresentam considerável valor de média maior do que os obtidos pelo último conjunto de pesos.

Embora o valor de acurácia média do conjunto de pesos salvos pela menor *loss* (89.406%) seja um pouco menor do que obtido pela ResNet 18 tradicional, o Vision Transformer obteve mediana de acurácias maior, conforme pode ser verificado na Figura 13 (acima) e obteve valor de média superior em 5 dos 11 problemas e igual em 1 dos problemas.

Aplicando o teste de hipóteses bootstrap pareado para diferença das médias entre a ResNet 18 e o *Vision Transformer* (resultados demonstrados na Figura 42), não podemos afirmar que existe diferença significativa com $pvalue < 0.05$.

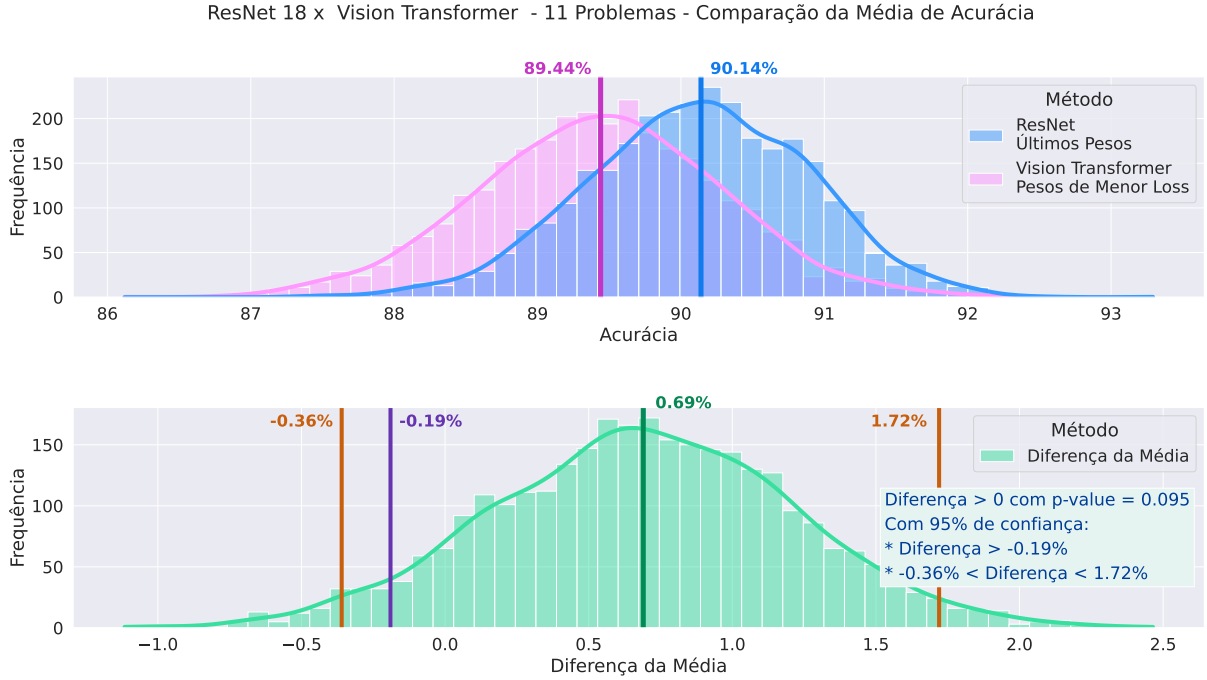

Figura 42 – ResNet 18 x Vision Transformer - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

Portanto, tanto a **ResNet 18** tradicional, quanto o **Vision Transformer** fornecem **resultados comparáveis** nos datasets analisados.

7.8.3 Hybrid Vision Transformer - Features Obtidos com a ResNet

Os hiperparâmetros usados nessa abordagem de rede neural foram os mesmos que obtiveram o melhor valor de acurácia para o modelo de Vision Transformer (descritos na Seção anterior 7.8.2).

Nessa abordagem, os *features* de entrada da rede neural foram obtidos pela representação embutida da última camada convolucional de uma ResNet 18 tradicional treinada previamente. Adotamos os modelos de ResNet 18 tradicional com o melhor conjunto de hiperparâmetros analisados anteriormente (Seção 7.5.2). As camadas da ResNet 18 foram congeladas, ou seja, o treinamento dos modelos Hybrid Vision Transformer alteram apenas as camadas relativas ao Vision Transformer.

A média de acurácias considerando todos 11 problemas e 20 sementes aleatórias foi **89.039%** para o conjunto de pesos salvos pela menor *loss* (avaliada no conjunto treinamento) e **87.195%** para os últimos pesos obtidos ao final do treinamento.

A Figura 43 abaixo apresenta os resultados do teste de hipóteses bootstrap pareado para diferença das médias com a ResNet 18. Segundo o teste, podemos afirmar com 95% de confiança que a **ResNet 18 tradicional** apresenta média de acurácias **superior** em pelo menos 0.49%.

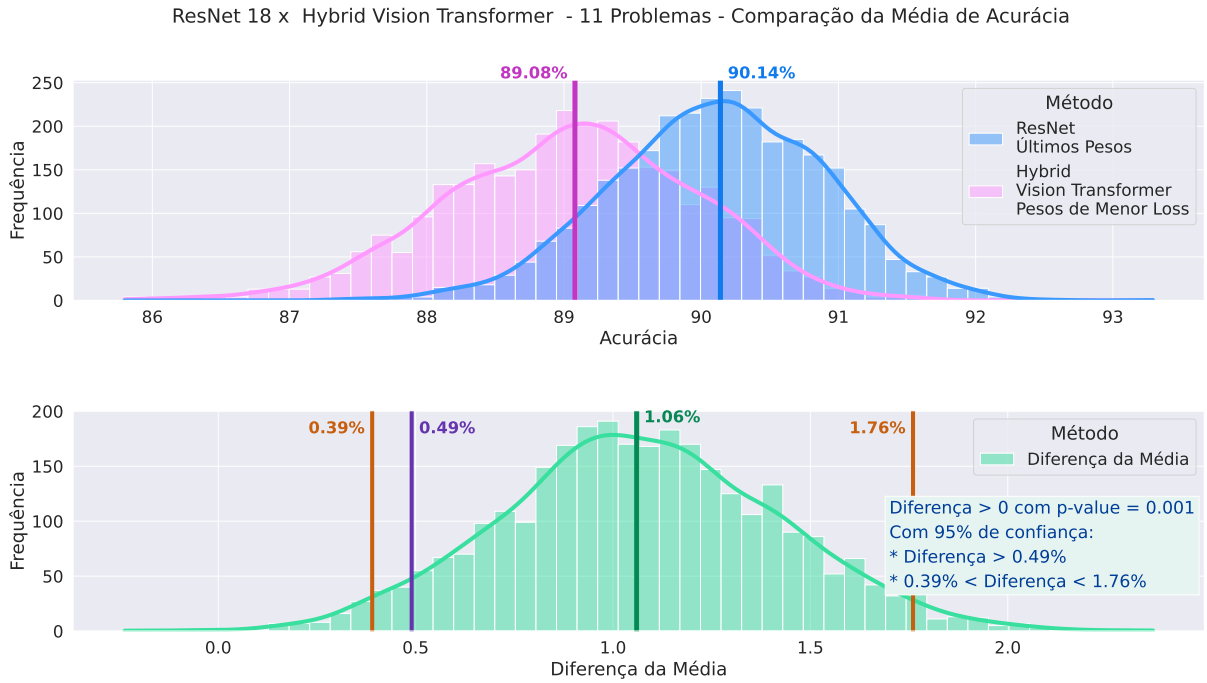

Figura 43 – ResNet 18 x Hybrid Vision Transformer - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

Talvez, fazendo *fine-tuning* dos hiperparâmetros, seja possível melhorar um pouco

as acurácias obtidas, mas os resultados obtidos sugerem que esse método não é uma das melhores soluções a serem adotadas.

7.9 GAN-MLP

O experimento com GAN-MLP foi motivado pelo trabalho de (ZHANG et al., 2018), o qual apresentou uma abordagem de *Generative Adversarial Networks* (GAN), nomeada de MetaGAN, aplicada ao problema de *few-shot learning* (aprendizagem a partir de poucas amostras).

Essa abordagem visava aproveitar o potencial das GAN em gerar amostras *fake* para explorar o espaço amostral, favorecendo a generalização de modelos treinados a partir de datasets com poucas amostras.

Em nossos experimentos, adotamos os modelos Discriminador e Gerador da GAN como redes neurais MLP, a seguir explicamos cada um desses modelos.

- O **Gerador** recebe, como entrada, o código latente Z de tamanho tam_z com valores variando aleatoriamente em uma distribuição uniforme entre -1 e 1. A camada de saída apresenta tantos neurônios quanto o número de *features* (genes) do dataset para o qual o modelo está sendo aplicado. A saída da última camada dessa rede passa por uma função sigmóide para que os valores de saída fiquem entre 0 e 1.

- O **Discriminador** é igual a uma rede MLP tratada na Seção 7.6, com a diferença de ter um neurônio a mais na última camada da rede (camada de saída). Esse neurônio modela a resposta do Discriminador sobre se a amostra, recebida como entrada, é *fake* (produzida pela rede geradora) ou real. Ou seja, a rede Discriminadora funciona como um discriminador do *framework* GAN tradicional e como um classificador.

Como hiperparâmetros, fixamos a função de ativação como *Leaky ReLU* com declive (*slope*) de 0.2 e 64 neurônios nas camadas escondidas (melhores hiperparâmetros obtidos nos experimentos com a MLP), também fixamos o número máximo de épocas em 500.

Os hiperparâmetros que variamos foram: o tamanho do código latente Z (tam_z) da rede neural Geradora como 16 e 64; o número de camadas escondidas (nr_hidden_layers) como 3 e 4; e a probabilidade de *dropout* (pdo) como 0.7 e 0.1.

Primeiramente, testamos, com uma semente aleatória, os conjuntos de hiperparâmetros ($tam\_z = 64$, $pdo = 0.7$, $nr\_hidden\_layers = 3$) e ($tam\_z = 16$, $pdo = 0.7$, $nr\_hidden\_ layers = 3$), ou seja, variando o tamanho do código latente. Percebemos que o código latente maior fornecia melhores resultados.

Então testamos com 4 camadas escondidas ($tam\_z = 64$, $pdo = 0.7$, $nr\_hidden\_layers = 4$) e percebemos um resultado melhor do que com 3 camadas.

Por último, diminuímos a probabilidade de dropout para 0.1 ($tam\_z = 64$, $pdo = 0.1$, $nr\_hidden\_layers = 4$) e obtemos piores resultados.

Então, rodamos até completar 20 sementes aleatórias para os três conjuntos de hiperparâmetros com probabilidade de *dropout* 0.7. Os resultados são resumidos na Figura 44 e Tabelas 15 e 16.

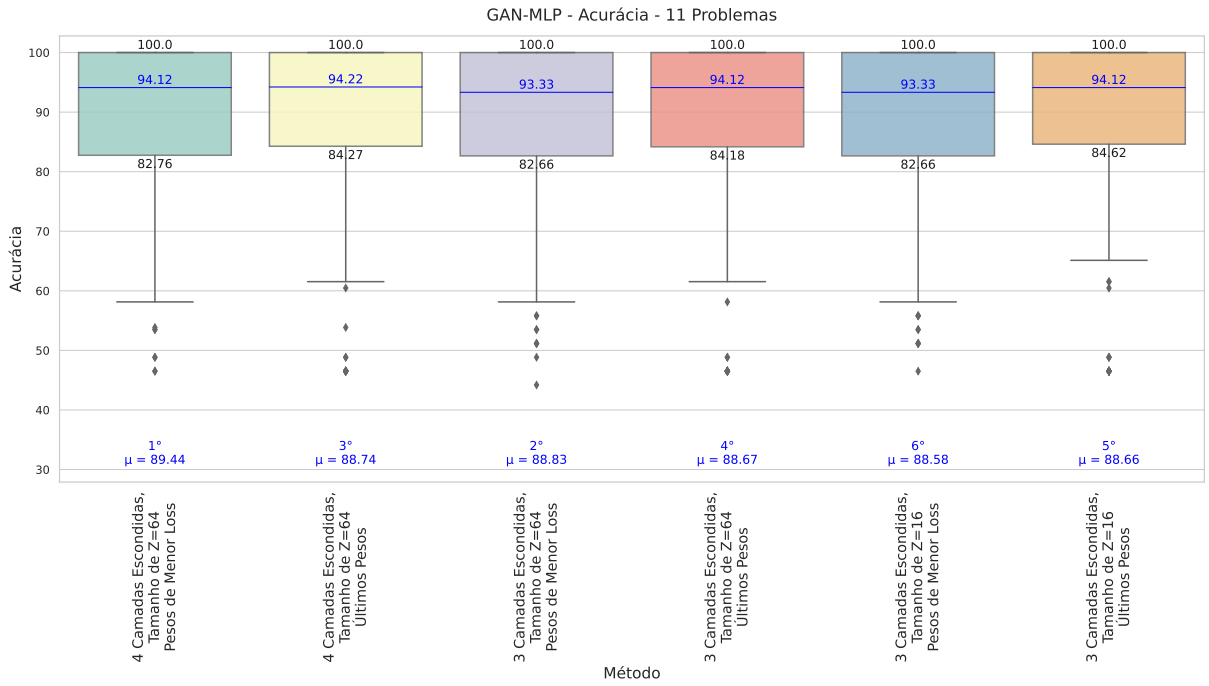

Figura 44 – Hiperparâmetros da GAN-MLP - 11 problemas.

Tabela 15 – Hiperparâmetros da GAN-MLP - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
4 Camadas Escondidas, Tamanho de Z=64 Pesos de Menor Loss	89.441	4.979	82.760	94.120	100.000	84.781	91.123	93.560	89.107	0.878
4 Camadas Escondidas, Tamanho de Z=64 Últimos Pesos	88.739	4.518	84.270	94.220	100.000	84.679	89.612	91.629	88.822	0.869
3 Camadas Escondidas, Tamanho de Z=64 Pesos de Menor Loss	88.829	5.203	82.657	93.330	100.000	84.749	90.522	93.198	88.714	0.872
3 Camadas Escondidas, Tamanho de Z=64 Últimos Pesos	88.666	4.686	84.180	94.120	100.000	85.551	89.252	91.955	88.596	0.869
3 Camadas Escondidas, Tamanho de Z=16 Pesos de Menor Loss	88.577	5.073	82.657	93.330	100.000	84.548	89.986	93.213	88.448	0.869
3 Camadas Escondidas, Tamanho de Z=16 Últimos Pesos	88.662	4.633	84.620	94.120	100.000	85.302	89.323	91.915	88.558	0.868

Tabela 16 – Hiperparâmetros da GAN-MLP - média de acurácia - cada problema mostrado separadamente.

	4 Camadas Escondidas, Tamanho de Z=64 Pesos de Menor Loss	4 Camadas Escondidas, Tamanho de Z=64 Últimos Pesos	3 Camadas Escondidas, Tamanho de Z=64 Pesos de Menor Loss	3 Camadas Escondidas, Tamanho de Z=64 Últimos Pesos	3 Camadas Escondidas, Tamanho de Z=16 Pesos de Menor Loss	3 Camadas Escondidas, Tamanho de Z=16 Últimos Pesos
(1, 4)	98.818	99.364	98.364	99.364	98.545	99.273
(8, 2)	95.459	95.939	95.699	96.048	95.611	96.354
(13, 2)	90.000	90.000	88.750	88.750	88.750	88.750
(18, 1)	88.371	88.034	88.933	88.933	88.708	88.764
(42, 1)	82.069	83.333	82.356	84.195	81.092	84.080
(51, 4)	63.256	51.628	60.698	52.209	60.349	52.093
(59, 4)	76.538	77.692	76.154	77.308	74.615	78.077
(62, 2)	94.118	95.588	92.941	95.294	94.118	95.000
(71, 1)	100.000	100.000	100.000	100.000	100.000	100.000
(77, 5)	95.333	94.667	94.333	93.333	93.333	93.000
(82, 4)	99.889	99.889	98.889	99.889	99.222	99.889
Total	89.441	88.739	88.829	88.666	88.577	88.662

O melhor conjunto de hiperparâmetros foi ($tam\_z = 64$, $pdo = 0.7$, $nr\_hidden\_layers = 4$) com os valores de acurácia média **89.441%** para os pesos salvos pela menor *loss* e **88.739%** para os últimos pesos (do final do treinamento).

A Figura 45 e Tabelas 18 e 17 comparam os resultados da GAN-MLP com o melhor conjunto de hiperparâmetros com os obtidos pela MLP. Podemos percerber que, para todos os problemas, com exceção do $(13, 2)$, a GAN-MLP apresenta piores resultados.

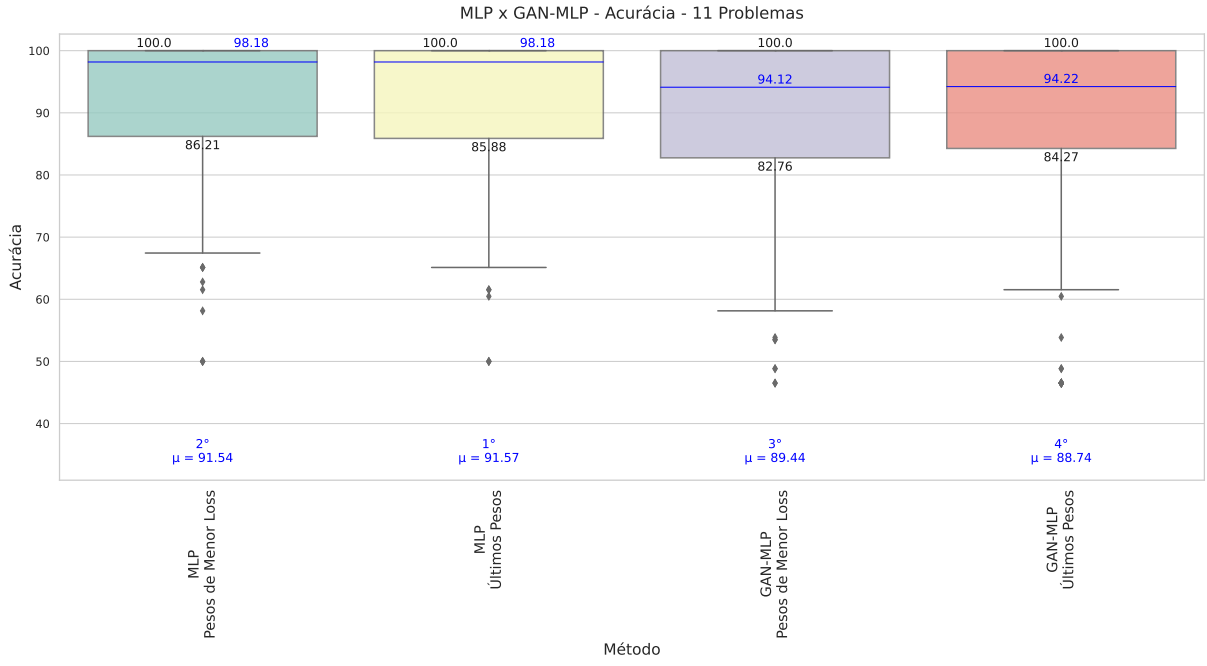

Figura 45 – MLP x GAN-MLP - 11 problemas.

Tabela 17 – MLP x GAN-MLP - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
MLP Pesos de Menor Loss	91.543	4.624	86.210	98.180	100.000	88.747	93.521	94.629	90.880	0.904
MLP Últimos Pesos	91.571	4.653	85.885	98.180	100.000	88.849	93.519	94.655	90.994	0.904
GAN-MLP Pesos de Menor Loss	89.441	4.979	82.760	94.120	100.000	84.781	91.123	93.560	89.107	0.878
GAN-MLP Últimos Pesos	88.739	4.518	84.270	94.220	100.000	84.679	89.612	91.629	88.822	0.869

Tabela 18 – MLP x GAN-MLP - média de acurácia - cada problema mostrado separadamente.

	MLP Pesos de Menor Loss	MLP Últimos Pesos	GAN-MLP Pesos de Menor Loss	GAN-MLP Últimos Pesos
(1, 4)	99.455	99.455	98.818	99.364
(8, 2)	98.231	98.231	95.459	95.939
(13, 2)	86.250	86.250	90.000	90.000
(18, 1)	90.000	90.337	88.371	88.034
(42, 1)	84.885	84.655	82.069	83.333
(51, 4)	71.860	72.442	63.256	51.628
(59, 4)	81.538	81.154	76.538	77.692
(62, 2)	98.529	98.529	94.118	95.588
(71, 1)	100.000	100.000	100.000	100.000
(77, 5)	96.333	96.333	95.333	94.667
(82, 4)	99.889	99.889	99.889	99.889
Total	91.543	91.571	89.441	88.739

Comparando com a MLP, através do teste de hipóteses (Figura 46), concluímos, com 95% de confiança, que a média de acurácias da **MLP é superior** em pelo menos 1.44%.

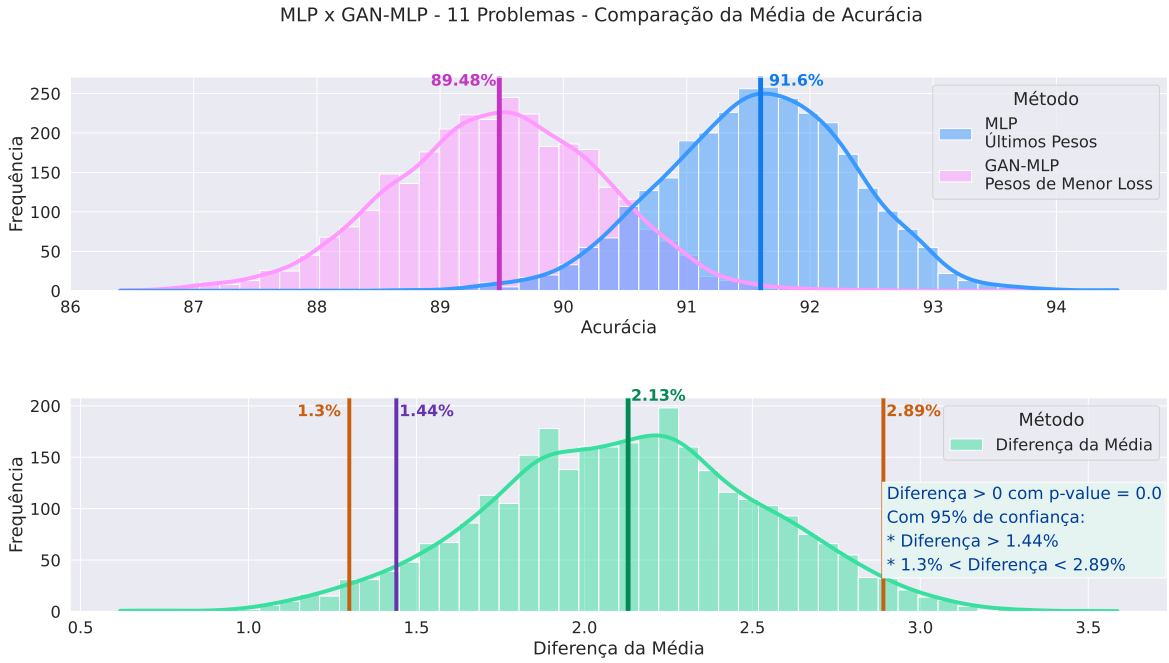

Figura 46 – MLP x GAN-MLP - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

Deve-notar, porém, que como foram testados poucos hiperparâmetros, talvez, fazendo *fine-tuning* dos mesmos, seja possível melhorar um pouco as acurácias obtidas. Mas os resultados que obtivemos sugerem que esse método não é uma das melhores soluções a serem adotadas.

7.10 MLPEns

Como explicado no Capítulo 5, a MLPEns busca utilizar o *dropout* para explorar um modelo de rede neural como um ensemble de modelos, favorecendo na generalização. Nos experimentos dessa abordagem utilizamos uma extensão da arquitetura de rede neural MLP com 3 camadas escondidas, *batch normalization* e *dropout* entre as camadas escondidas - a arquitetura usada foi apresentada na Seção 5.1. Foram testados os seguintes hiperparâmetros, de acordo com o procedimento descrito na Seção 4.4.1.

- probabilidade de *dropout*: 0.05, 0.1, 0.2, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9;
- número de canais nas camadas internas: 8, 16, 32 e 64;
- funções de ativação: *Leaky ReLU* com declive (slope) 0.1, 0.2 e 0.3 e *RReLU (Randomized Leaky ReLU)* com valor de declive variando uniformemente entre 0.125 e 0.333... (valores *default* no PyTorch);
- tipo de *loss*: *individual, combinada* e *individual_combinada*;
- probabilidade de dropout usada na predição: 0.05, 0.1, 0.2, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9;
- regra para combinação das saídas usada na predição: produto e votação.

Para a escolha do melhor conjunto de pesos, fixou-se $nr\_propagacoes\_escolha\_pesos = 20$ (ver Capítulo 5). Na etapa de predição, utilizou-se a combinação de $nr\_propagacoes\_pre- dicao = 300$ saídas do mesmo modelo. Como o *dropout* permanece ativado nessas tarefas, podemos ter diferentes saídas em um único modelo de rede neural para uma mesma entrada.

É importante salientar que, na predição ou na escolha do melhor conjunto de pesos, não é necessário computar gradientes, portanto, o tempo para cada propagação de amostra pela rede neural é consideravelmente menor do que no treinamento. De forma que o custo, em tempo de processamento, para computar as 300 saídas não se torna um empecilho. Além disso, o custo de predição / treino de uma arquitetura com camadas *fully connected* (como as MLP) é razoavelmente menor do que em uma rede neural CNN.

O conjunto de hiperparâmetros selecionado como o melhor, para ser testado com as 20 sementes aleatórias, foi:

- probabilidade de *dropout* ($prob\_dropout$): 0.5;
- número de canais nas camadas internas: 64;
- função de ativação: *Leaky ReLU* com declive (slope) 0.2;
- tipo de *loss*: *individual*;
- probabilidade de dropout usada na predição ($prob\_dropout\_predicao$): 0.9.
- regra para combinação das saídas usada na predição: votação.

A Figura 47 e as Tabelas 19 e 20 a seguir mostram os resultados obtidos com o melhor conjunto de hiperparâmetros comparando com os principais métodos estudados.

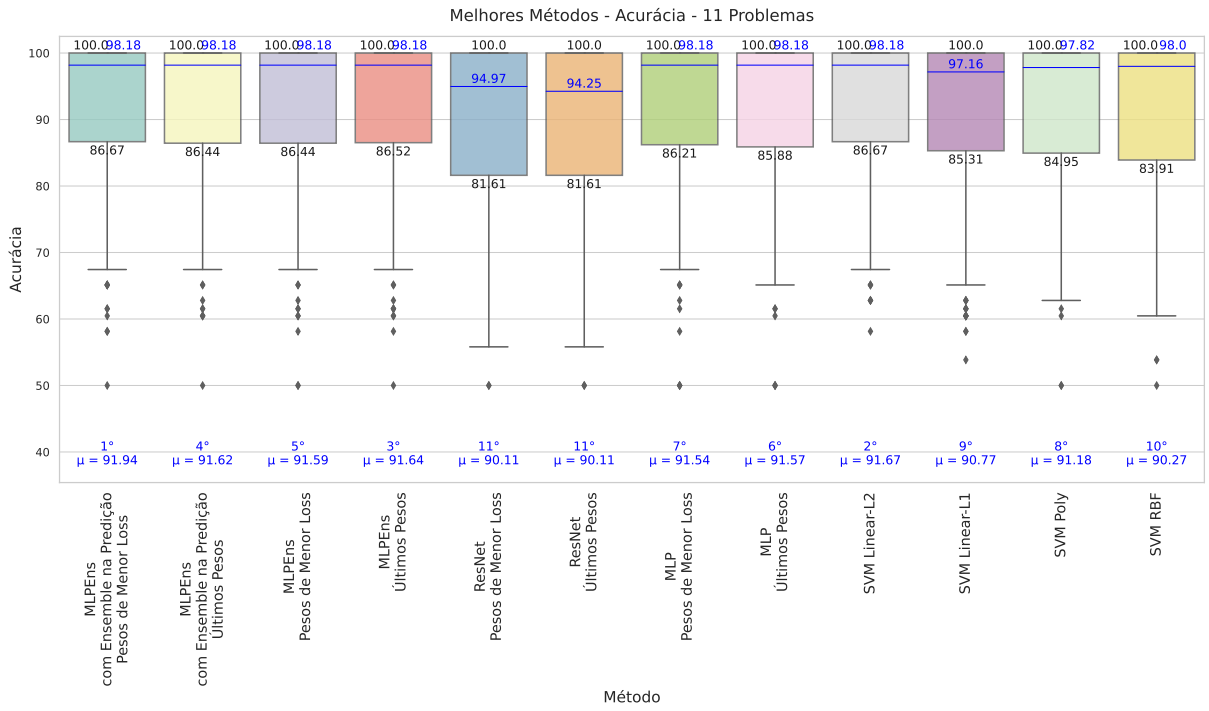

Figura 47 – MLPEns x ResNet x MLP x SVM - 11 problemas.

Tabela 19 – MLPEns x ResNet x MLP x SVM - 11 problemas.

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
MLPEns com Ensemble na Predição Pesos de Menor Loss	91.938	4.394	86.670	98.180	100.000	88.200	93.592	95.424	91.400	0.910
MLPEns com Ensemble na Predição Últimos Pesos	91.620	4.406	86.442	98.180	100.000	88.070	93.210	95.266	91.031	0.906
MLPEns Pesos de Menor Loss	91.587	4.571	86.442	98.180	100.000	88.730	93.190	95.338	91.058	0.906
MLPEns Últimos Pesos	91.635	4.422	86.520	98.180	100.000	88.206	93.211	95.126	91.064	0.907
ResNet Pesos de Menor Loss	90.105	4.805	81.610	94.970	100.000	86.941	90.253	93.802	89.020	0.885
ResNet Últimos Pesos	90.106	4.806	81.610	94.250	100.000	87.019	90.379	93.812	89.029	0.885
MLP Pesos de Menor Loss	91.543	4.624	86.210	98.180	100.000	88.747	93.521	94.629	90.880	0.904
MLP Últimos Pesos	91.571	4.653	85.885	98.180	100.000	88.849	93.519	94.655	90.994	0.904
SVM Linear-L2	91.666	3.903	86.667	98.182	100.000	88.369	92.735	94.232	90.647	0.904
SVM Linear-L1	90.770	3.552	85.308	97.160	100.000	89.331	91.626	92.989	89.758	0.896
SVM Poly	91.179	4.336	84.950	97.820	100.000	88.264	92.133	94.467	89.539	0.893
SVM RBF	90.266	4.341	83.910	98.000	100.000	86.133	91.458	93.127	88.456	0.881

Tabela 20 – MLPEns x ResNet x MLP x SVM - média de acurácia por método - cada problema mostrado separadamente.

	MLPEns com Ensemble na Predição Pesos de Menor Loss	MLPEns com Ensemble na Predição Últimos Pesos	MLPEns Pesos de Menor Loss	MLPEns Últimos Pesos	ResNet Pesos de Menor Loss	ResNet Últimos Pesos	MLP Pesos de Menor Loss	MLP Últimos Pesos	SVM Linear-L2	SVM Linear-L1	SVM Poly	SVM RBF
(1, 4)	99.364	99.455	99.545	99.545	99.455	99.455	99.455	99.455	99.455	99.364	99.455	99.455
(8, 2)	98.231	98.253	98.166	98.275	98.362	98.384	98.231	98.231	98.362	97.795	98.341	98.275
(13, 2)	91.250	88.750	87.500	88.750	83.750	83.750	86.250	86.250	91.250	95.000	87.500	90.000
(18, 1)	90.618	90.562	90.562	90.506	88.708	88.764	90.000	90.337	89.494	90.169	89.551	88.371
(42, 1)	85.517	85.115	85.230	85.460	79.713	79.828	84.885	84.655	86.092	84.425	85.575	82.471
(51, 4)	70.814	71.047	70.930	70.698	72.209	72.326	71.860	72.442	69.651	64.651	71.395	72.442
(59, 4)	80.769	80.000	80.769	80.000	78.462	78.462	81.538	81.154	80.385	73.846	78.846	69.231
(62, 2)	98.529	98.529	98.529	98.529	97.941	97.647	98.529	98.529	98.529	95.882	98.529	98.235
(71, 1)	100.000	100.000	100.000	100.000	98.333	98.333	100.000	100.000	100.000	100.000	100.000	100.000
(77, 5)	96.333	96.333	96.333	96.333	94.667	94.667	96.333	96.333	95.333	97.333	94.000	94.667
(82, 4)	99.889	99.778	99.889	99.889	99.556	99.556	99.889	99.889	99.778	100.000	99.778	99.778
Total	91.938	91.620	91.587	91.635	90.105	90.106	91.543	91.571	91.666	90.770	91.179	90.266

Embora a média de acurácia da MLPEns esteja um pouco acima do SVM, realizando o teste de hipóteses (ver Figura 48), concluímos que **não podemos afirmar que exista diferença de média de acurácia entre a MLPEns e o SVM** com $pvalue < 0.05$.

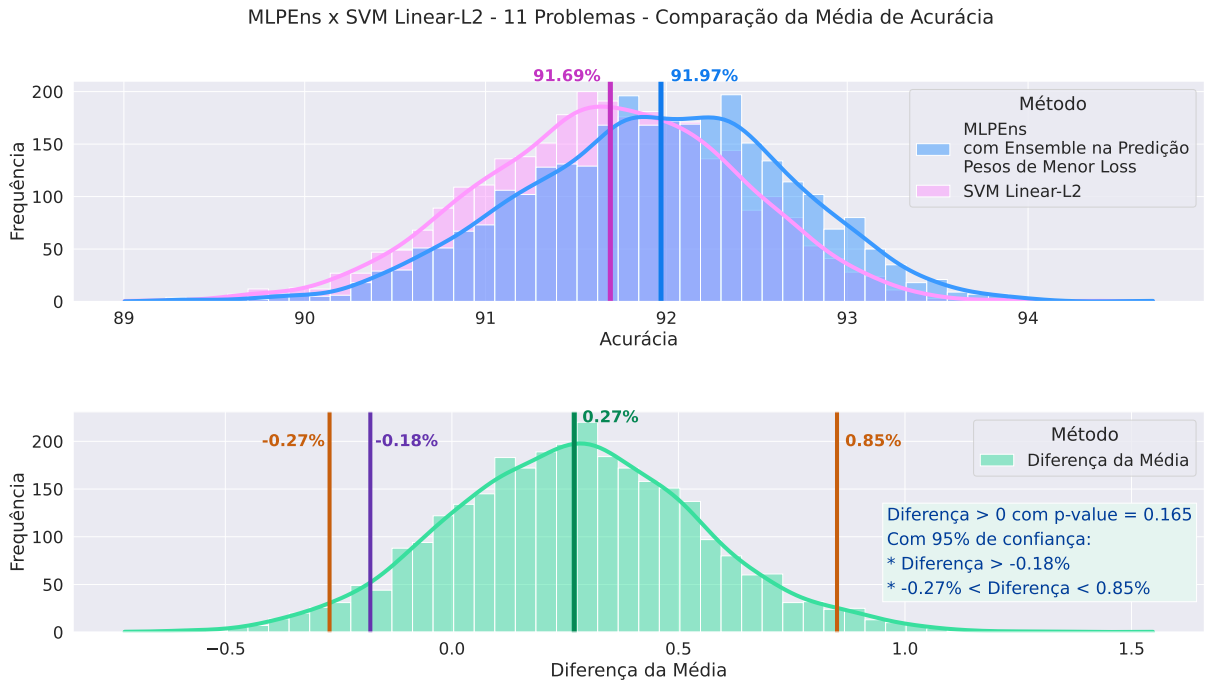

Figura 48 – Comparação da MLPEns com o SVM - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

Fazendo o teste de hipóteses para comparação de médias da MLPEns com a ResNet (ver Figura 49), concluímos que a média de acurácia da **MLPEns é superior à da**

ResNet, em pelo menos, 1.14%.

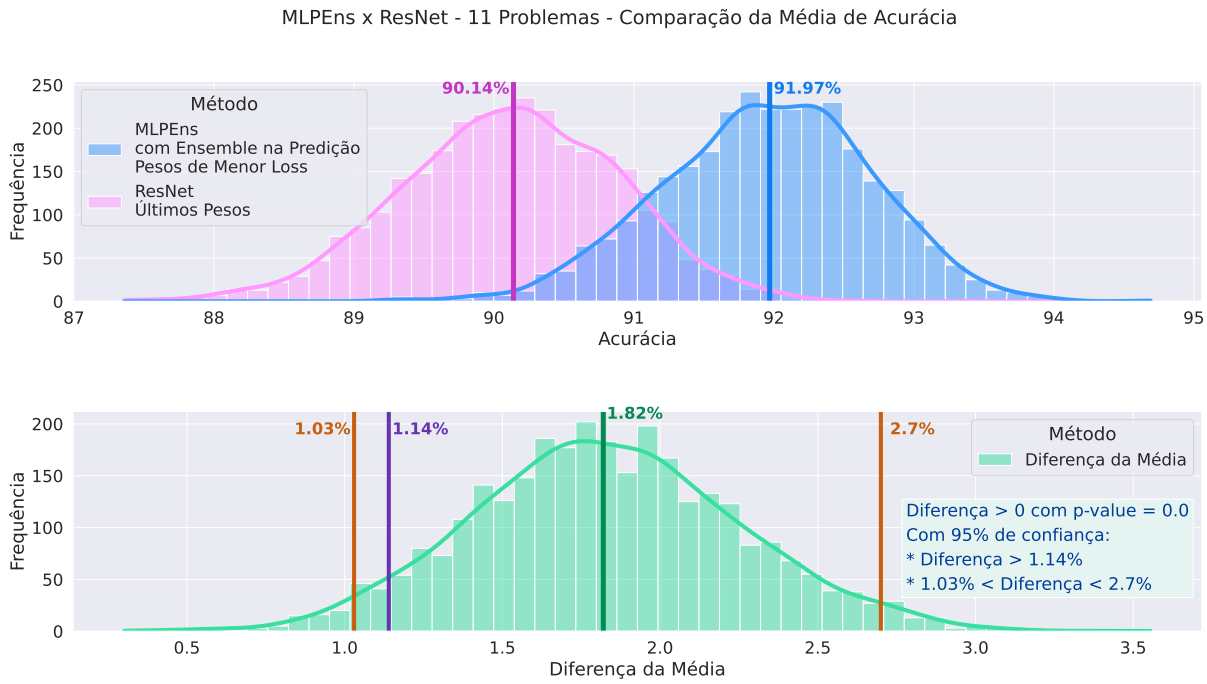

Figura 49 – Comparação da MLPEns com a ResNet - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

A média de acurácia da MLPEns obtida foi um pouco acima da MLP, mas, realizando o teste de hipóteses (ver Figura 50), concluímos que **não podemos afirmar que exista diferença de média de acurácia entre a MLPEns e a MLP** com $pvalue < 0.05$.

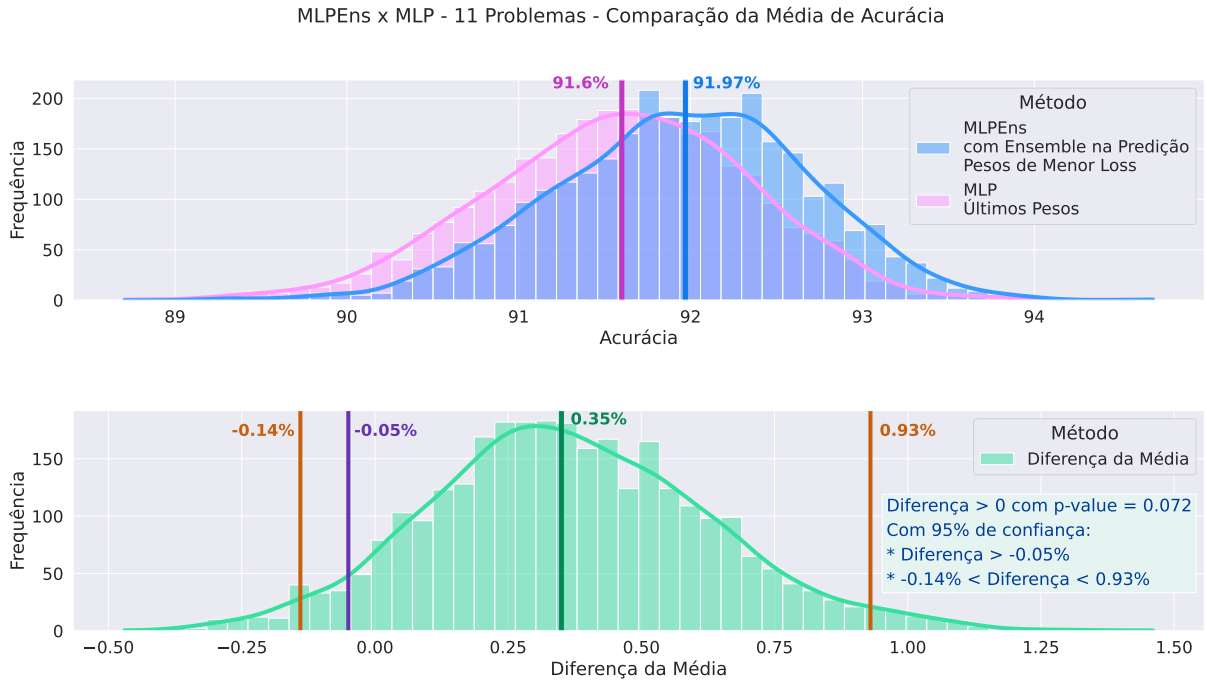

Figura 50 – Comparação da MLPEns com a MLP - Teste de hipóteses bootstrap pareado para diferença das médias de acurácia.

7.11 Métodos de Aprendizagem Rasa

Em uma primeira etapa, realizamos os experimentos no dataset piloto sobre Meduloblastoma GSE85217 com $n = 763$ amostras, para separação (classificação) dos quatro subgrupos moleculares: *Group 3, Group 4, SHH e WNT*.

Treinamos 18 diferentes métodos de Aprendizagem Rasa, com uma execução (uma semente aleatória) por método. A Tabela 21 apresenta os resultados obtidos. Para esses experimentos foi utilizada a biblioteca PyCaret e os hiperparâmetros *default* da biblioteca. Em todos os métodos, empregou-se o conjunto total de features, com exceção do CatBoost que, por restrições de memória, empregou-se uma redução dimensional por PCA e utilização de 90% dos principais componentes.

Tabela 21 – Resultados da classificação com 18 métodos de Aprendizagem Rasa (dataset piloto e 1 semente aleatória).

Id	Método	Acurácia(%)
lr	Logistic Regression	98.63
knn	K Neighbors Classifier	98.63
nb	Naive Bayes	99.09
dt	Decision Tree Classifier	90.87
svm	SVM - Linear Kernel	98.17
rbfsvm	SVM - Radial Kernel	93.61
gpc	Gaussian Process Classifier	9.13
mlp	MLP Classifier	98.17
ridge	Ridge Classifier	98.63
rf	Random Forest Classifier	97.26
qda	Quadratic Discriminant Analysis	47.95
ada	Ada Boost Classifier	79.91
gbc	Gradient Boosting Classifier	97.26
lda	Linear Discriminant Analysis	98.17
et	Extra Trees Classifier	99.09
xgboost	Extreme Gradient Boosting	97.72
lightgbm	Light Gradient Boosting Machine	97.26
catboost	CatBoost Classifier (PCA - 90% dos pcs)	97.26

Os melhores classificadores em termos de acurácia nessa etapa foram executados com 20 sementes aleatórias distintas para cada um dos 11 problemas (datasets) selecionados. Os resultados são apresentados na Figura 51 e nas Tabelas 22 e 23 a seguir.

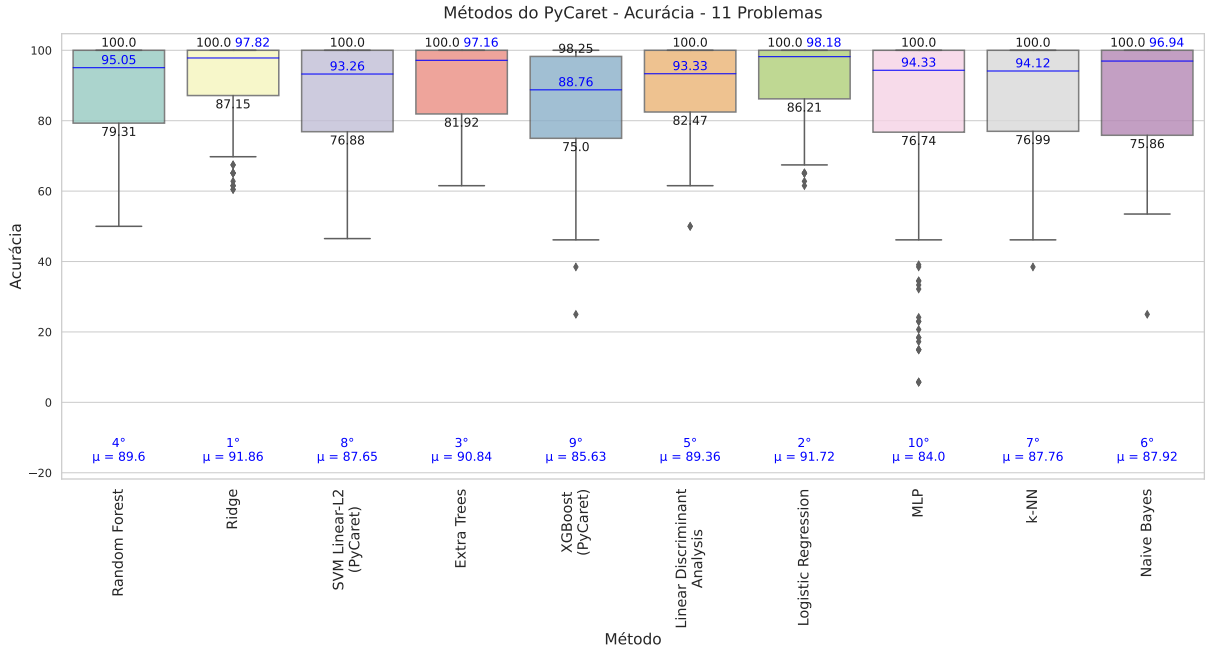

Figura 51 – Métodos de Aprendizagem Rasa do PyCaret (11 problemas e 20 sementes aleatórias).

Tabela 22 – Métodos de Aprendizagem Rasa do PyCaret (11 problemas e 20 sementes aleatórias).

	Média	Desvio Padrão	Q1	Mediana	Q3	Média de Q1	Média da Mediana	Média de Q3	Revocação	F1-score
Random Forest	89.595	3.984	79.310	95.051	100.000	86.747	91.328	92.154	86.160	0.863
Ridge	91.855	3.951	87.146	97.817	100.000	88.643	93.606	94.413	90.695	0.906
SVM Linear-L2 (PyCaret)	87.649	6.414	76.878	93.258	100.000	83.904	88.072	92.892	85.902	0.848
Extra Trees	90.837	3.786	81.919	97.162	100.000	87.441	91.600	93.661	88.546	0.886
XGBoost (PyCaret)	85.625	5.803	75.000	88.764	98.253	82.559	87.132	89.532	83.419	0.833
Linear Discriminant Analysis	89.363	5.207	82.471	93.333	100.000	85.423	90.695	93.703	88.029	0.876
Logistic Regression	91.720	3.912	86.207	98.182	100.000	88.382	93.275	94.332	90.404	0.902
MLP	84.001	6.767	76.744	94.332	100.000	80.248	85.948	89.516	82.249	0.811
k-NN	87.759	4.367	76.989	94.118	100.000	83.967	88.980	90.955	85.616	0.852
Naive Bayes	87.919	4.612	75.862	96.943	100.000	84.909	89.516	91.025	84.904	0.842

Tabela 23 – Métodos de Aprendizagem Rasa do PyCaret - média de acurácia - cada problema mostrado separadamente (20 sementes aleatórias).

	Random Forest	Ridge	SVM Linear-L2 (PyCaret)	Extra Trees	XGBoost (PyCaret)	Linear Discriminant Analysis	Logistic Regression	MLP	k-NN	Naive Bayes
(1, 4)	97.636	99.455	98.727	98.455	98.091	98.000	99.455	98.909	99.455	99.091
(8, 2)	98.057	98.319	97.249	97.882	97.511	98.559	98.559	98.035	97.293	97.664
(13, 2)	88.750	91.250	83.750	92.500	68.750	85.000	90.000	86.250	90.000	88.750
(18, 1)	86.404	89.888	89.551	87.247	88.034	89.551	90.506	84.270	84.101	81.685
(42, 1)	79.770	86.609	71.322	80.690	81.264	82.874	86.092	23.966	76.207	75.747
(51, 4)	71.163	70.116	69.419	72.326	70.814	71.279	72.442	70.814	71.395	67.791
(59, 4)	73.462	80.385	76.923	77.692	61.154	75.385	79.231	73.077	59.615	63.846
(62, 2)	98.529	97.941	93.529	98.529	89.706	98.235	98.529	96.471	97.059	97.647
(71, 1)	100.000	100.000	93.333	100.000	100.000	91.667	100.000	100.000	100.000	100.000
(77, 5)	92.333	96.667	91.667	94.333	88.000	92.667	94.333	93.000	91.000	95.333
(82, 4)	99.444	99.778	98.667	99.556	98.556	99.778	99.778	99.222	99.222	99.556
Total	89.595	91.855	87.649	90.837	85.625	89.363	91.720	84.001	87.759	87.919

O classificador de maior média de acurácia foi o **Ridge Classifier** com **91.855%**, seguido da **Regressão Logística** com acurácia média de **91.720%** e as **Extra Trees** com **90.837%**.

7.12 Comparação entre os Principais Métodos de Classificação

As Tabelas 24, 25 e 26 apresentam uma comparação entre os melhores métodos de classificação. Entre os classificadores rasos o método **Ridge Classifier** apresentou maior acurácia, após a **Regressão Logística, SVM Linear 2** (implementação do *sklean.linearSVC*). Nos métodos de redes neurais os melhores métodos foram a **MLPEns e a MLP**.

Tabela 24 – Melhores Métodos (parte 1/3) - média de acurácia - cada problema mostrado separadamente.

	MLPEns com Ensemble na Predição Pesos de Menor Loss	MLPEns com Ensemble na Predição Últimos Pesos	MLPEns Pesos de Menor Loss	MLPEns Últimos Pesos	ResNet Pesos de Menor Loss	ResNet Últimos Pesos
(1, 4)	99.364	99.455	99.545	99.545	99.455	99.455
(8, 2)	98.231	98.253	98.166	98.275	98.362	98.384
(13, 2)	91.250	88.750	87.500	88.750	83.750	83.750
(18, 1)	90.618	90.562	90.562	90.506	88.708	88.764
(42, 1)	85.517	85.115	85.230	85.460	79.713	79.828
(51, 4)	70.814	71.047	70.930	70.698	72.209	72.326
(59, 4)	80.769	80.000	80.769	80.000	78.462	78.462
(62, 2)	98.529	98.529	98.529	98.529	97.941	97.647
(71, 1)	100.000	100.000	100.000	100.000	98.333	98.333
(77, 5)	96.333	96.333	96.333	96.333	94.667	94.667
(82, 4)	99.889	99.778	99.889	99.889	99.556	99.556
Total	91.938	91.620	91.587	91.635	90.105	90.106

Tabela 25 – Melhores Métodos (parte 2/3) - média de acurácia - cada problema mostrado separadamente.

	MLP Pesos de Menor Loss	MLP Últimos Pesos	SVM Linear-L2	SVM Linear-L1	SVM Poly	SVM RBF	Random Forest
(1, 4)	99.455	99.455	99.455	99.364	99.455	99.455	97.636
(8, 2)	98.231	98.231	98.362	97.795	98.341	98.275	98.057
(13, 2)	86.250	86.250	91.250	95.000	87.500	90.000	88.750
(18, 1)	90.000	90.337	89.494	90.169	89.551	88.371	86.404
(42, 1)	84.885	84.655	86.092	84.425	85.575	82.471	79.770
(51, 4)	71.860	72.442	69.651	64.651	71.395	72.442	71.163
(59, 4)	81.538	81.154	80.385	73.846	78.846	69.231	73.462
(62, 2)	98.529	98.529	98.529	95.882	98.529	98.235	98.529
(71, 1)	100.000	100.000	100.000	100.000	100.000	100.000	100.000
(77, 5)	96.333	96.333	95.333	97.333	94.000	94.667	92.333
(82, 4)	99.889	99.889	99.778	100.000	99.778	99.778	99.444
Total	91.543	91.571	91.666	90.770	91.179	90.266	89.595

Tabela 26 – Melhores Métodos (parte 3/3) - média de acurácia - cada problema mostrado separadamente.

	Ridge	SVM Linear-L2 (PyCaret)	Extra Trees	XGBoost (PyCaret)	Linear Discriminant Analysis	Logistic Regression	MLP (PyCaret)	k-NN	Naive Bayes
(1, 4)	99.455	98.727	98.455	98.091	98.000	99.455	98.909	99.455	99.091
(8, 2)	98.319	97.249	97.882	97.511	98.559	98.559	98.035	97.293	97.664
(13, 2)	91.250	83.750	92.500	68.750	85.000	90.000	86.250	90.000	88.750
(18, 1)	89.888	89.551	87.247	88.034	89.551	90.506	84.270	84.101	81.685
(42, 1)	86.609	71.322	80.690	81.264	82.874	86.092	23.966	76.207	75.747
(51, 4)	70.116	69.419	72.326	70.814	71.279	72.442	70.814	71.395	67.791
(59, 4)	80.385	76.923	77.692	61.154	75.385	79.231	73.077	59.615	63.846
(62, 2)	97.941	93.529	98.529	89.706	98.235	98.529	96.471	97.059	97.647
(71, 1)	100.000	93.333	100.000	100.000	91.667	100.000	100.000	100.000	100.000
(77, 5)	96.667	91.667	94.333	88.000	92.667	94.333	93.000	91.000	95.333
(82, 4)	99.778	98.667	99.556	98.556	99.778	99.778	99.222	99.222	99.556
Total	91.855	87.649	90.837	85.625	89.363	91.720	84.001	87.759	87.919

A Tabela 27 apresenta o tempo de execução (treino e avaliação nos conjuntos de dados de treinamento e de teste, respectivamente) dos principais métodos, com os conjuntos de hiperparâmetros selecionados como melhores, e features de entrada como a expressão dos genes com filtragem prévia pelo filtro de kruskal, para uma semente aleatória e dataset GSE85217 de 763 amostras com o problema de separação dos subtipos moleculares de meduloblastoma - problema (8,2). Ambos tempos foram tomados com a execução em uma conta pro do Google Colab com GPU P100.

Tabela 27 – Tempo de Execução dos Principais Métodos - uma semente aleatória e um conjunto de hiperparâmetros - problema de classificação de subtipos moleculares de meduloblastoma (8,2) - GSE85217 (763 amostras) - executados em uma conta pro do Google Colab com GPU P100.

Método	**Tempo (minutos : segundos)**
ResNet 18 - 64 canais	11:03
MLPEns - 64 neurônios	05:00
GAN-MLP - 64 neurônios, 64 códigos latentes, 4 camadas escondidas	02:17
MLP - 64 neurônios	01:34
SVM - kernel linear L2	00:45

Como podemos visualizar, o método ResNet apresenta o maior tempo de execução (**11:03** - 11 minutos e 3 segundos) devido à realização das convoluções que demandam maior tempo para propagação e retropropagação do que em camadas *fully connected*. Seguindo, temos a MLPEns com tempo de **5:00** minutos - que demanda menos do que a metade de tempo que a ResNet e cerca de três vezes mais tempo que a MLP devido à necessidade de executar a combinação de saídas dos neurônios (*ensemble*). Após, temos a GAN-MLP (tempo de **2:17**) e MLP (tempo de **1:34**) que apresentam apenas camadas *fully connected* e, por último, o SVM (tempo de **0:45**) que demanda cerca da metade do tempo que a MLP.

8 RESULTADOS E DISCUSSÕES - SELEÇÃO DE GENES

Este capítulo apresenta os resultados e discussões relacionadas às abordagens de seleção de features importantes. Primeiramente, é abordada a seleção de genes com SVM e a MLPEns aplicada em meduloblastoma e sarcoma de Ewing. Após, descrevem-se os resultados com o BorutaShap aplicado em meduloblastoma. Por fim, apresentam-se algumas discussões comparativas.

8.1 Seleção de Genes Relevantes

SVM com kernel Linear-L2, MLPEns e BorutaShap foram usados para seleção de genes relevantes, o procedimento foi descrito no Capítulo 6. A seguir, na Seção 8.2, apresentamos os resultados da seleção de genes com o SVM e a MLPEns, aplicadas nos datasets de Meduloblastoma (GSE85217) - problema (8,2) e de Sarcoma de Ewing (GSE2553) - problema (1,4). Após, na Seção 8.3, apresentamos os resultados do BorutaShap aplicado no dataset de Meduloblastoma (GSE85217) - problema (8,2).

8.2 SVM Linear L2 e MLPEns

Para os métodos **SVM** e **MLPEns** temos duas formas (critérios) para selecionar os genes:

- através dos **coeficientes** propriamente ditos (ou através da média de coeficientes para a combinação dos modelos);
- estabelecendo um rank de *features* (genes) mais importantes dos modelos e, após, realizando uma média das **posições** nos ranks dos diferentes modelos a serem combinados.

As Tabelas 28 e 29 mostram o número de genes na interseção dos conjuntos de genes selecionados pelos dois tipos de critérios obtidos pelos métodos SVM (Tabela 28) e MLPEns (Tabela 29), como explicado a seguir.

As colunas indicam o número de genes que se está escolhendo como relevantes; as linhas, o problema de classificação tratado. Então, por exemplo, no problema (1,4) (linha), se tomarmos 30 genes (coluna) como importantes pelo critério da posição e, após, pelo critério de coeficiente, teremos 26 genes selecionados igualmente por ambos critérios no SVM (Tabela 28); e 29 na MLPEns (Tabela 29). Como podemos visualizar, os genes selecionados variam com o critério especificado.

Tabela 28 – SVM - Interseção dos genes selecionados i) pelo critério de posição; e ii) pelos coeficientes.

	1	**2**	**3**	**5**	**10**	**20**	**30**	**100**	**200**	**300**	**500**	**1000**
(1, 4)	1	1	3	4	9	19	26	87	178	266	452	862
(8, 2)	0	2	3	5	9	17	25	92	193	281	462	932
(13, 2)	0	1	2	4	7	17	28	87	180	266	426	823
(18, 1)	1	1	2	3	6	13	21	79	147	224	396	831
(42, 1)	0	1	3	4	8	19	25	91	188	278	469	937
(51, 4)	0	0	0	0	5	9	13	41	85	124	242	712
(59, 4)	0	1	2	3	6	19	25	72	137	204	366	785
(62, 2)	1	1	3	4	7	16	28	82	174	255	428	860
(71, 1)	1	2	3	5	10	19	29	97	190	287	480	922
(77, 5)	0	1	1	2	5	12	20	76	160	250	411	861
(82, 4)	1	2	3	5	9	17	26	89	179	269	454	932

Tabela 29 – MLPEns - Interseção dos genes selecionados i) pelo critério de posição; e ii) pelos coeficientes.

	1	**2**	**3**	**5**	**10**	**20**	**30**	**100**	**200**	**300**	**500**	**1000**
(1, 4)	1	2	3	4	9	16	29	91	187	285	468	957
(8, 2)	1	1	2	4	8	12	22	67	146	226	386	835
(13, 2)	0	1	2	3	5	12	20	72	153	250	464	972
(18, 1)	1	1	2	2	9	14	21	70	153	241	418	889
(42, 1)	0	2	2	4	7	16	22	85	170	260	462	952
(51, 4)	0	1	2	3	6	18	24	91	185	286	489	983
(59, 4)	0	1	3	4	7	16	22	82	164	256	440	923
(62, 2)	1	1	2	3	7	16	25	85	179	270	451	935
(71, 1)	0	1	3	4	7	11	21	82	165	253	448	939
(77, 5)	0	2	2	3	6	15	21	85	170	265	450	916
(82, 4)	0	2	2	4	8	16	24	84	172	257	445	933

Estudando ambos critérios, consideramos o de **posição** como mais consistente e de maior **compreensão intuitiva**, pois o coeficiente (valor de importância dado a um gene) pode variar consideravelmente de modelo para modelo.

As Tabelas 30 e 31 apresentam a interseção dos genes selecionados por ambos métodos (SVM e MLPEns) pelo critério de posição (Tabela 30) e pelos coeficientes (Tabela 31). Como pode ser observado, existe **variabilidade do conjunto de genes selecionados** pelos diferentes métodos. Essa variabilidade (no conjunto de genes selecionados) também foi encontrada no método BorutaShap como observaremos na Seção 8.3.

Tabela 30 – SVM e MLPEns - Posição - Interseção dos conjuntos de genes selecionados pelo critério de posição pelo i) SVM e ii) MLPEns.

	1	2	3	5	10	20	30	100	200	300	500	1000
(1, 4)	0	1	2	5	5	10	19	58	133	194	336	779
(8, 2)	0	0	0	1	1	2	3	10	31	64	139	344
(13, 2)	0	0	0	0	0	2	4	25	76	159	420	987
(18, 1)	0	1	2	3	7	12	14	71	132	203	359	724
(42, 1)	0	0	0	0	1	2	2	27	75	126	229	564
(51, 4)	0	0	1	2	5	9	16	60	124	238	471	979
(59, 4)	0	0	0	0	1	6	10	52	129	206	361	799
(62, 2)	0	0	0	1	2	6	12	49	103	170	287	645
(71, 1)	0	0	0	0	0	0	1	20	62	118	235	698
(77, 5)	0	0	0	2	4	6	7	44	104	171	301	615
(82, 4)	0	0	0	0	3	5	9	28	67	130	237	537

Tabela 31 – SVM e MLPEns - Coeficiente - Interseção dos conjuntos de genes selecionados pelos coeficientes pelo i) SVM e ii) MLPEns

	1	2	3	5	10	20	30	100	200	300	500	1000
(1, 4)	0	1	2	3	5	13	21	62	132	188	314	679
(8, 2)	0	0	0	0	1	2	3	9	21	39	104	301
(13, 2)	0	0	0	0	1	3	5	25	72	145	359	828
(18, 1)	0	0	1	2	6	10	19	61	145	204	359	729
(42, 1)	0	0	0	0	1	2	4	24	65	120	220	545
(51, 4)	0	1	1	1	4	7	11	31	64	100	224	699
(59, 4)	0	1	1	1	3	5	9	51	103	170	309	703
(62, 2)	0	0	0	1	2	9	13	45	114	161	277	586
(71, 1)	0	0	0	0	0	1	1	22	58	116	238	664
(77, 5)	0	0	0	1	4	6	13	47	115	182	302	595
(82, 4)	0	0	0	0	2	5	9	28	69	119	234	550

Destacamos, como principal motivo por obter essa variabilidade, a **alta redundância** de informação contida nos *features* de dados de análise de expressão gênica - as expressões dos diferentes genes são altamente correlacionadas, um gene interage com muitos. De forma que, para os métodos de classificação, mais do que um mesmo *feature* (gene) pode ser escolhido de forma equivalente. Essa observação nos faz concluir a necessidade de, ao estabelecer assinaturas gênicas, **estudar mais do que um modelo**.

Nas duas subseções, a seguir, mostramos os resultados dos genes selecionados como importantes para o Meduloblastoma - problema (8,2) - GSE85217 e para o Sarcoma de Ewing - (1,4) - GSE2553.

8.2.1 Meduloblastoma - (8,2) - GSE85217

Nessa seção, investigamos a seleção de genes importantes para separar os quatro subgrupos moleculares de meduloblastoma: *Group 3*, *Group 4*, *SHH* e *WNT*. A Figura 52 apresenta o tSNE desse dataset com todos os *features* disponíveis (21.641 genes).

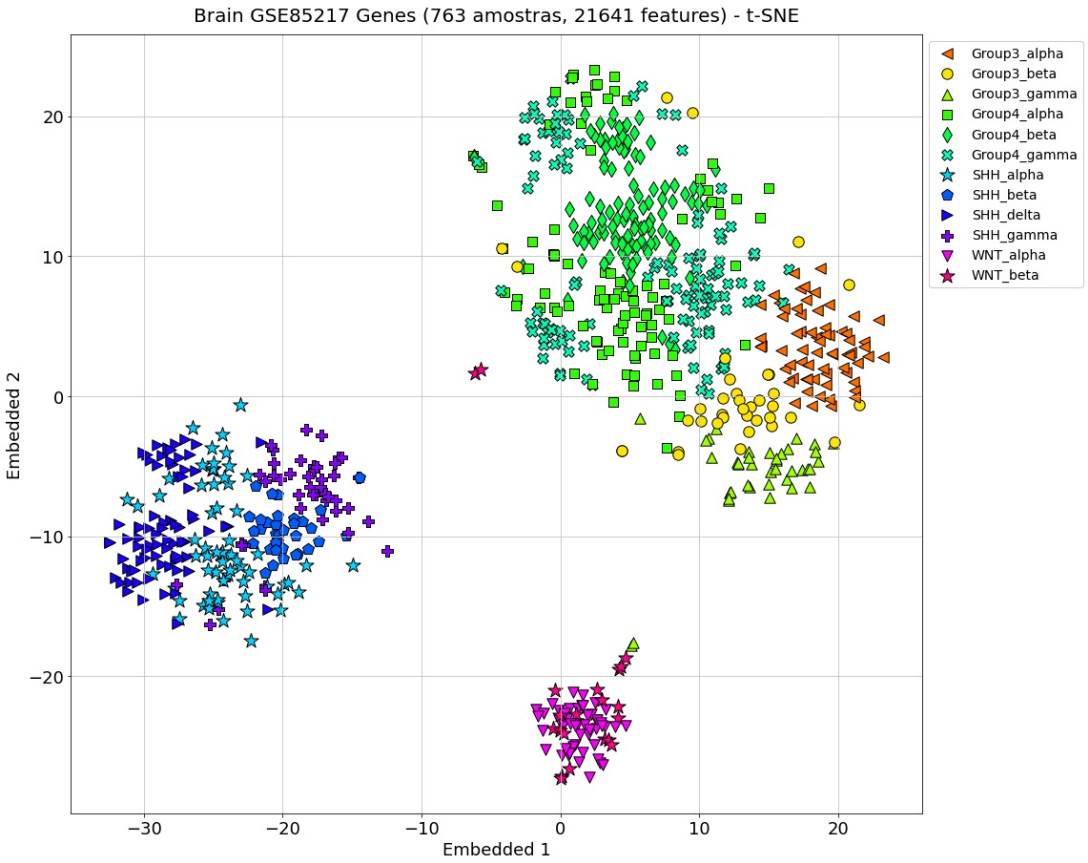

Figura 52 – tSNE - Meduloblastoma - todos features (21.641 genes).

8.2.1.1 SVM

As Tabelas 32 e 33 mostram os 30 genes selecionados pelo **SVM**, expostos de forma decrescente em importância. Na Tabela 32, empregou-se o critério de **posição**; enquanto que, na 33, empregou-se o critério de **coeficientes**. As Figuras 53 e 54 apresentam os *heatmap* (Agrupamento Hierárquico) usando os 30 genes selecionados pela posição e coeficiente, respectivamente.

As tabelas foram montadas combinando a saída de 20 execuções, ou seja, os genes estão ordenados pela média de importância das 20 execuções. A média de posição (*Média Pos*) e de coeficiente (*Média Coef*) estão em negrito (o valor dos coeficientes foi multiplicado por 1.000, por questões de legibilidade). Quanto maior for o valor nas colunas *Média Pos* e *Média Coef*, maior importância foi dada ao gene em questão.

As colunas *Std Pos* e *Std Coef* (desvio padrão de posição e de coeficiente) fornecem uma informação valiosa: **a posição de importância dos genes varia consideravelmente**. Os genes mais distantes nessa ordem de importância (os últimos na tabela) têm maiores valores de desvio padrão. Quanto maior for o desvio padrão de um gene, houve maior variação na forma como os classificadores o viam. Isso pode significar que:

- existem muitos genes com expressão relacionada a esse que também podem estar sendo escolhidos em detrimento desse; ou
- pode indicar que o mesmo pode estar sendo escolhido como um ruído; ou ainda
- pode ser um gene pouco representativo da classe.

Portanto, se a aplicação buscada for um conjunto mínimo de genes biomarcadores para **diagnóstico**, podemos desconsiderar genes com alto desvio padrão. Se a aplicação buscada for **entender o mecanismo de uma doença**, essa tarefa se torna mais complexa e desconsiderá-lo pode fornecer uma percepção incompleta da informação disponível nos dados. Por isso, percebemos a importância de se trabalhar (em trabalhos futuros) com construção de redes gênicas de doenças para obter uma maior interpretabilidade da seleção de genes feita pelos classificadores.

O desvio padrão também pode ser uma métrica analisada para indicar qual é o tamanho mínimo de genes que devem ser selecionados como importantes.

Tabela 32 – SVM pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
LMX1A	**21638.2**	1.9	21641	21633	21637.0	21638.0	21640.0	**0.246**	0.021	0.299	0.214	0.231	0.248	0.262
GPR15	**21637.8**	6.8	21641	21611	21637.8	21640.5	21641.0	**0.260**	0.038	0.343	0.189	0.233	0.263	0.283
TFAP2D	**21635.4**	7.4	21641	21614	21633.0	21639.0	21640.0	**0.244**	0.033	0.299	0.183	0.224	0.244	0.270
PDE10A	**21634.6**	10.2	21641	21596	21635.8	21638.0	21639.0	**0.23**5	0.025	0.280	0.181	0.220	0.233	0.254
NPR3	**21634.2**	7.9	21641	21604	21633.0	21636.0	21639.0	**0.233**	0.025	0.292	0.184	0.217	0.226	0.246
TRPM3	**21628.8**	9.9	21640	21606	21627.0	21632.0	21636.0	**0.212**	0.018	0.246	0.180	0.199	0.215	0.227
RBM24	**21627.7**	12.3	21641	21601	21626.0	21632.5	21635.0	**0.213**	0.020	0.246	0.180	0.203	0.213	0.227
ATOH1	**21625.0**	8.0	21636	21608	21620.8	21627.0	21631.2	**0.203**	0.017	0.244	0.180	0.192	0.197	0.215
DSC2	**21617.4**	19.0	21639	21577	21611.0	21626.5	21631.0	**0.198**	0.021	0.253	0.163	0.186	0.201	0.208
SIX6	**21615.5**	12.1	21637	21576	21610.0	21615.5	21622.0	**0.191**	0.013	0.237	0.168	0.183	0.189	0.196
ROBO1	**21613.5**	34.4	21640	21482	21609.2	21624.0	21631.0	**0.197**	0.020	0.229	0.144	0.187	0.202	0.211
DSCAM	**21613.0**	23.0	21638	21564	21603.0	21622.0	21629.2	**0.195**	0.021	0.247	0.160	0.177	0.194	0.208
DSP	**21609.8**	49.7	21639	21407	21616.5	21622.5	21632.8	**0.201**	0.027	0.253	0.134	0.185	0.201	0.217
NEUROG1	**21609.6**	17.8	21636	21573	21595.8	21613.5	21625.0	**0.190**	0.019	0.229	0.167	0.172	0.184	0.203
SIX1	**21608.5**	33.8	21637	21489	21591.2	21623.5	21630.2	**0.194**	0.024	0.230	0.146	0.172	0.199	0.212
KHDRBS2	**21601.2**	25.8	21630	21538	21595.0	21607.0	21621.8	**0.183**	0.015	0.210	0.160	0.172	0.182	0.196
NDP	**21599.6**	20.0	21629	21538	21592.5	21601.0	21611.8	**0.180**	0.011	0.207	0.156	0.173	0.180	0.186
PBX1	**21599.0**	44.1	21633	21425	21596.5	21605.0	21624.2	**0.185**	0.018	0.219	0.143	0.177	0.183	0.197
SLC24A2	**21597.2**	37.5	21636	21459	21588.2	21605.5	21620.2	**0.184**	0.020	0.229	0.140	0.171	0.184	0.193
ADAMTS6	**21596.8**	28.4	21637	21519	21585.0	21604.5	21616.0	**0.182**	0.019	0.236	0.156	0.168	0.181	0.187
SLC16A4	**21595.8**	46.1	21639	21457	21590.8	21610.0	21624.2	**0.186**	0.021	0.234	0.146	0.170	0.181	0.197
PEX5L	**21594.4**	35.7	21626	21455	21587.8	21606.0	21612.5	**0.179**	0.012	0.204	0.147	0.172	0.180	0.185
HUNK	**21592.5**	38.5	21637	21515	21553.0	21608.0	21626.2	**0.184**	0.025	0.247	0.150	0.163	0.180	0.197
CACNA1G	**21591.2**	40.2	21634	21476	21575.0	21605.5	21618.0	**0.180**	0.019	0.219	0.143	0.169	0.177	0.193
CSMD1	**21590.0**	30.5	21624	21511	21579.8	21597.0	21613.2	**0.176**	0.014	0.203	0.147	0.170	0.175	0.187
ALDH1A3	**21585.0**	61.7	21632	21350	21571.0	21603.0	21622.8	**0.180**	0.021	0.217	0.135	0.164	0.180	0.195
TBR1	**21584.5**	39.3	21634	21473	21575.5	21590.5	21607.0	**0.176**	0.018	0.223	0.144	0.166	0.175	0.184
NID2	**21583.1**	48.4	21637	21417	21568.8	21593.0	21617.0	**0.178**	0.020	0.228	0.142	0.164	0.173	0.185
TENM4	**21580.6**	45.5	21636	21483	21556.8	21589.5	21618.0	**0.177**	0.020	0.214	0.143	0.163	0.174	0.190
GRM8	**21580.2**	35.0	21633	21500	21562.8	21585.5	21601.8	**0.173**	0.015	0.202	0.148	0.165	0.171	0.180

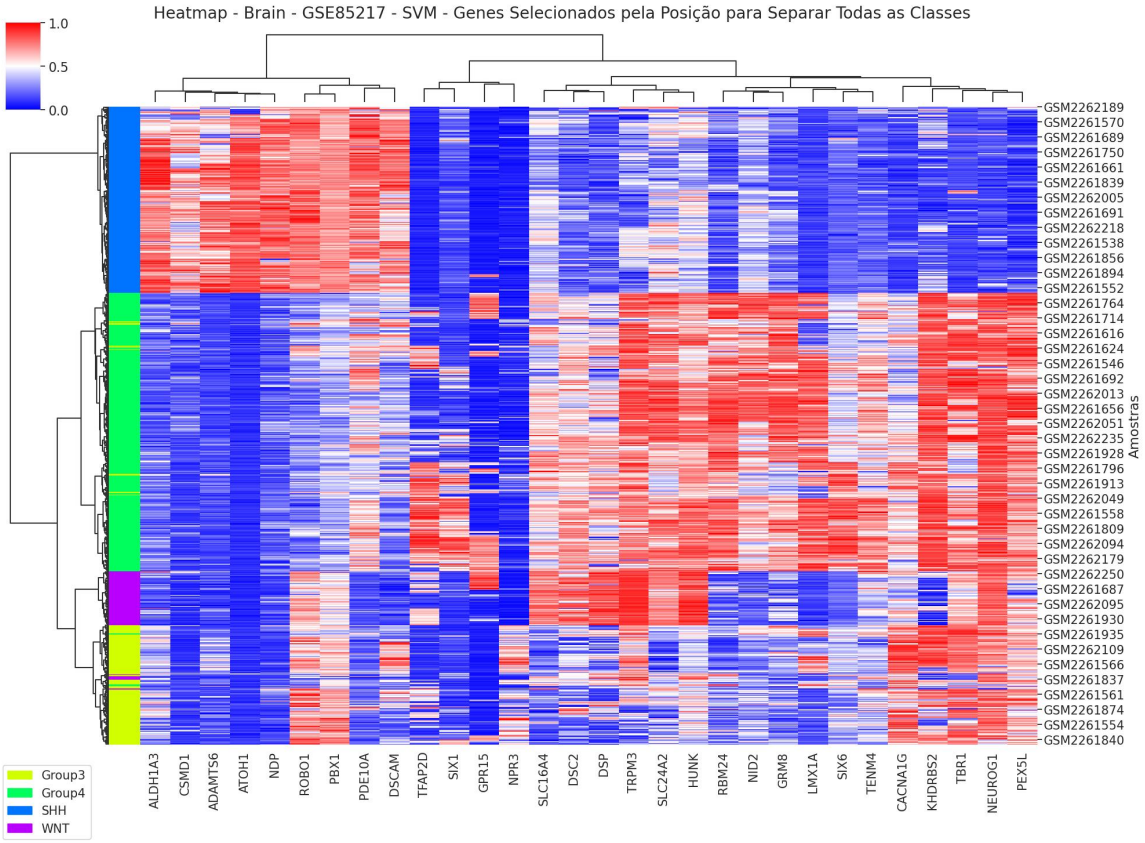

Figura 53 – SVM pela Posição - Melhores Genes- Heatmap.

Tabela 33 – SVM pelo Coeficiente - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
GPR15	**21637.8**	6.8	21641	21611	21637.8	21640.5	21641.0	**0.260**	0.038	0.343	0.189	0.233	0.263	0.283
LMX1A	**21638.2**	1.9	21641	21633	21637.0	21638.0	21640.0	**0.246**	0.021	0.299	0.214	0.231	0.248	0.262
TFAP2D	**21635.4**	7.4	21641	21614	21633.0	21639.0	21640.0	**0.244**	0.033	0.299	0.183	0.224	0.244	0.270
PDE10A	**21634.6**	10.2	21641	21596	21635.8	21638.0	21639.0	**0.235**	0.025	0.280	0.181	0.220	0.233	0.254
NPR3	**21634.2**	7.9	21641	21604	21633.0	21636.0	21639.0	**0.233**	0.025	0.292	0.184	0.217	0.226	0.246
RBM24	**21627.7**	12.3	21641	21601	21626.0	21632.5	21635.0	**0.213**	0.020	0.246	0.180	0.203	0.213	0.227
TRPM3	**21628.8**	9.9	21640	21606	21627.0	21632.0	21636.0	**0.212**	0.018	0.246	0.180	0.199	0.215	0.227
ATOH1	**21625.0**	8.0	21636	21608	21620.8	21627.0	21631.2	**0.203**	0.017	0.244	0.180	0.192	0.197	0.215
DSP	**21609.8**	49.7	21639	21407	21616.5	21622.5	21632.8	**0.201**	0.027	0.253	0.134	0.185	0.201	0.217
DSC2	**21617.4**	19.0	21639	21577	21611.0	21626.5	21631.0	**0.198**	0.021	0.253	0.163	0.186	0.201	0.208
ROBO1	**21613.5**	34.4	21640	21482	21609.2	21624.0	21631.0	**0.197**	0.020	0.229	0.144	0.187	0.202	0.211
DSCAM	**21613.0**	23.0	21638	21564	21603.0	21622.0	21629.2	**0.195**	0.021	0.247	0.160	0.177	0.194	0.208
SIX1	**21608.5**	33.8	21637	21489	21591.2	21623.5	21630.2	**0.194**	0.024	0.230	0.146	0.172	0.199	0.212
LINC02893	**21445.2**	519.0	21640	19364	21566.5	21619.5	21635.8	**0.194**	0.052	0.284	0.082	0.162	0.195	0.242
SIX6	**21615.5**	12.1	21637	21576	21610.0	21615.5	21622.0	**0.191**	0.013	0.237	0.168	0.183	0.189	0.196
NEUROG1	**21609.6**	17.8	21636	21573	21595.8	21613.5	21625.0	**0.190**	0.019	0.229	0.167	0.172	0.184	0.203
SLC16A4	**21595.8**	46.1	21639	21457	21590.8	21610.0	21624.2	**0.186**	0.021	0.234	0.146	0.170	0.181	0.197
PBX1	**21599.0**	44.1	21633	21425	21596.5	21605.0	21624.2	**0.185**	0.018	0.219	0.143	0.177	0.183	0.197
SLC24A2	**21597.2**	37.5	21636	21459	21588.2	21605.5	21620.2	**0.184**	0.020	0.229	0.140	0.171	0.184	0.193
POTEA	**21570.7**	100.6	21637	21203	21563.8	21603.0	21629.2	**0.184**	0.030	0.244	0.121	0.166	0.180	0.212
HUNK	**21592.5**	38.5	21637	21515	21553.0	21608.0	21626.2	**0.184**	0.025	0.247	0.150	0.163	0.180	0.197
KHDRBS2	**21601.2**	25.8	21630	21538	21595.0	21607.0	21621.8	**0.183**	0.015	0.210	0.160	0.172	0.182	0.196
ADAMTS6	**21596.8**	28.4	21637	21519	21585.0	21604.5	21616.0	**0.182**	0.019	0.236	0.156	0.168	0.181	0.187
ZIC2	**21574.9**	83.3	21634	21291	21559.8	21614.5	21623.8	**0.180**	0.024	0.218	0.125	0.161	0.184	0.202
CACNA1G	**21591.2**	40.2	21634	21476	21575.0	21605.5	21618.0	**0.180**	0.019	0.219	0.143	0.169	0.177	0.193
NDP	**21599.6**	20.0	21629	21538	21592.5	21601.0	21611.8	**0.180**	0.011	0.207	0.156	0.173	0.180	0.186
ALDH1A3	**21585.0**	61.7	21632	21350	21571.0	21603.0	21622.8	**0.180**	0.021	0.217	0.135	0.164	0.180	0.195
GABRA5	**21569.4**	105.3	21636	21192	21569.0	21607.0	21626.0	**0.179**	0.024	0.214	0.121	0.167	0.184	0.196
PEX5L	**21594.4**	35.7	21626	21455	21587.8	21606.0	21612.5	**0.179**	0.012	0.204	0.147	0.172	0.180	0.185
KLF8	**21550.4**	132.7	21638	21105	21545.8	21601.5	21619.5	**0.178**	0.031	0.241	0.115	0.159	0.177	0.198

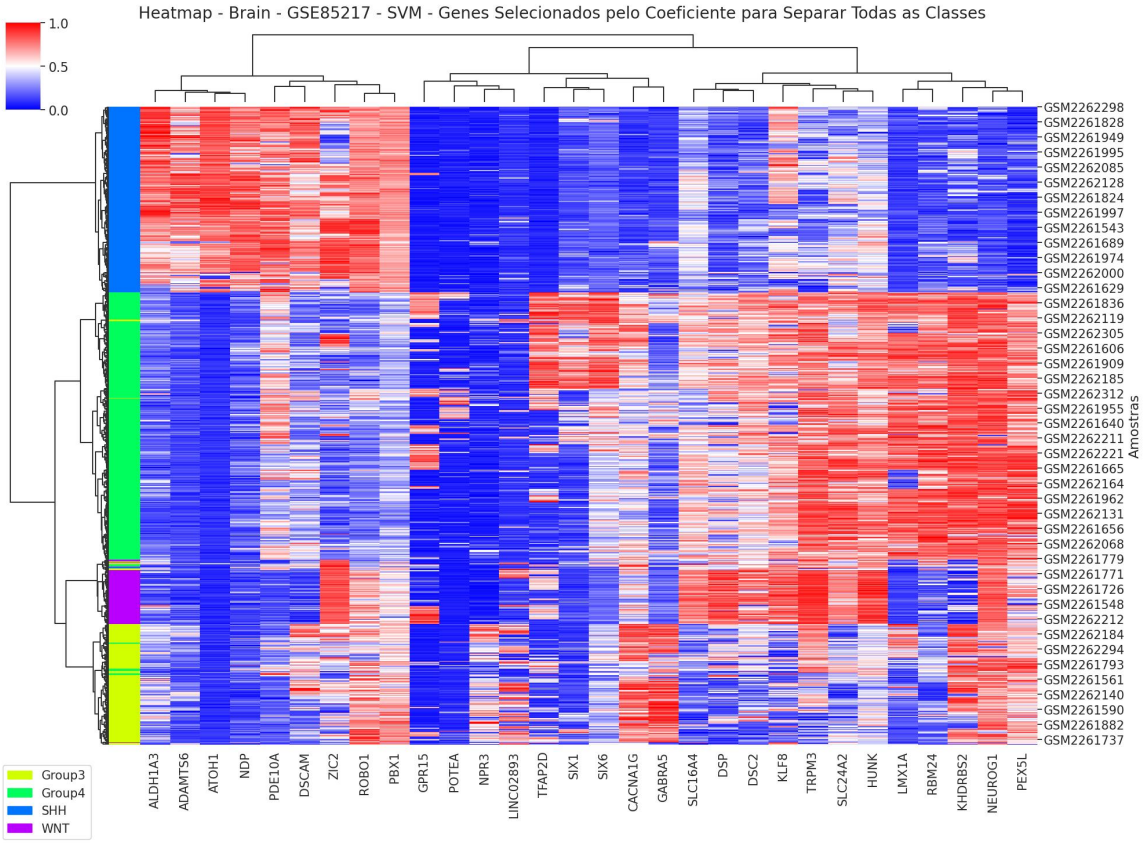

Figura 54 – SVM pelo Coeficiente - Melhores Genes - Heatmap.

8.2.1.2 MLPEns

As Tabelas 34 e 35 mostram os 30 genes selecionados pela **MLPEns**, expostos de forma decrescente em importância. Na Tabela 34, empregou-se o critério de **posição**; enquanto que, na 35, empregou-se o critério de **coeficientes**. As Figuras 55 e 56 apresentam os *heatmaps* (Agrupamentos Hierárquicos), usando os 30 genes selecionados pela posição e coeficiente, respectivamente.

Tabela 34 – MLPEns pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
POTEA	**21634.0**	13.5	21641	21582	21637.5	21639.0	21640.0	**0.168**	0.042	0.258	0.116	0.141	0.152	0.199
RNY3P10	**21614.0**	56.6	21640	21375	21619.2	21629.5	21638.2	**0.139**	0.030	0.196	0.093	0.118	0.128	0.161
ENSG00000201329	**21561.0**	181.0	21640	20863	21590.8	21631.0	21639.0	**0.138**	0.037	0.215	0.067	0.108	0.131	0.162
NPR3	**21538.0**	147.0	21637	21052	21551.8	21589.5	21616.2	**0.114**	0.023	0.166	0.081	0.097	0.110	0.121
NPFFR2	**21523.2**	345.5	21641	20042	21585.0	21633.5	21637.0	**0.132**	0.029	0.187	0.071	0.108	0.131	0.152
RNU6-297P	**21493.2**	212.6	21639	20869	21448.2	21605.0	21634.2	**0.119**	0.033	0.192	0.077	0.094	0.113	0.144
VSX1	**21490.4**	274.0	21640	20520	21536.0	21608.5	21632.0	**0.124**	0.038	0.201	0.070	0.096	0.115	0.144
LMO1	**21464.2**	204.5	21634	20835	21462.8	21538.0	21595.2	**0.103**	0.018	0.143	0.079	0.089	0.099	0.115
RN7SKP104	**21439.4**	584.0	21641	19124	21612.0	21632.0	21638.0	**0.147**	0.061	0.347	0.066	0.115	0.143	0.158
AKAP5	**21433.4**	261.8	21622	20526	21373.0	21549.0	21594.5	**0.102**	0.017	0.137	0.076	0.086	0.103	0.112
RNU6-329P	**21424.4**	244.0	21629	20718	21400.0	21508.0	21604.8	**0.102**	0.020	0.141	0.079	0.084	0.096	0.118
SLC9A2	**21423.5**	324.2	21629	20243	21395.2	21568.0	21602.2	**0.107**	0.023	0.147	0.063	0.086	0.108	0.127
LYPD1	**21418.4**	389.1	21639	19834	21471.8	21513.0	21596.8	**0.102**	0.015	0.137	0.069	0.094	0.102	0.108
TAFA1	**21411.5**	359.0	21638	20151	21473.2	21548.0	21605.8	**0.105**	0.022	0.153	0.072	0.086	0.103	0.118
ROBO3	**21404.8**	272.5	21635	20619	21373.5	21523.5	21567.8	**0.100**	0.017	0.125	0.071	0.091	0.101	0.110
MLIP	**21375.0**	432.9	21634	19873	21371.8	21539.5	21603.2	**0.106**	0.027	0.163	0.070	0.084	0.098	0.127
CCNA1	**21346.4**	286.3	21604	20334	21349.2	21414.5	21527.2	**0.093**	0.012	0.112	0.071	0.085	0.093	0.102
LIPH	**21319.2**	400.8	21635	19824	21361.0	21406.5	21509.2	**0.096**	0.018	0.141	0.071	0.086	0.091	0.098
CLEC9A	**21308.2**	832.5	21629	17725	21383.8	21556.5	21597.5	**0.103**	0.022	0.146	0.057	0.087	0.108	0.116
GPR15	**21275.1**	612.6	21627	19071	21373.5	21518.0	21599.0	**0.100**	0.019	0.126	0.061	0.088	0.103	0.114
ANOS2P	**21243.4**	595.8	21626	19568	21308.2	21466.5	21565.0	**0.097**	0.019	0.130	0.062	0.085	0.095	0.108
SPZ1	**21232.1**	989.6	21637	17145	21493.2	21592.5	21605.2	**0.107**	0.025	0.159	0.058	0.095	0.112	0.119
GLUD2	**21230.4**	530.8	21640	19743	21155.5	21484.5	21598.8	**0.099**	0.023	0.155	0.064	0.080	0.098	0.113
ASIC4	**21210.2**	361.2	21627	20361	20993.0	21340.0	21477.2	**0.089**	0.012	0.117	0.069	0.082	0.088	0.093
JCAD	**21178.5**	587.4	21623	19403	21139.2	21365.5	21573.5	**0.095**	0.023	0.136	0.062	0.075	0.090	0.112
SIX1	**21167.6**	597.7	21615	19182	21086.0	21360.5	21562.2	**0.094**	0.021	0.135	0.064	0.080	0.092	0.104
CARMIL3	**21163.8**	363.0	21573	20085	20940.2	21254.0	21456.5	**0.087**	0.013	0.119	0.066	0.078	0.084	0.091
RNU6-1001P	**21160.0**	1212.7	21641	16298	21439.8	21577.0	21624.0	**0.114**	0.036	0.198	0.055	0.091	0.105	0.135
BRF1	**21139.8**	814.5	21608	18487	21107.5	21521.5	21560.2	**0.093**	0.016	0.117	0.060	0.081	0.094	0.107
SOX14	**21083.6**	856.1	21640	17919	20854.8	21444.0	21583.2	**0.096**	0.025	0.155	0.060	0.077	0.091	0.120

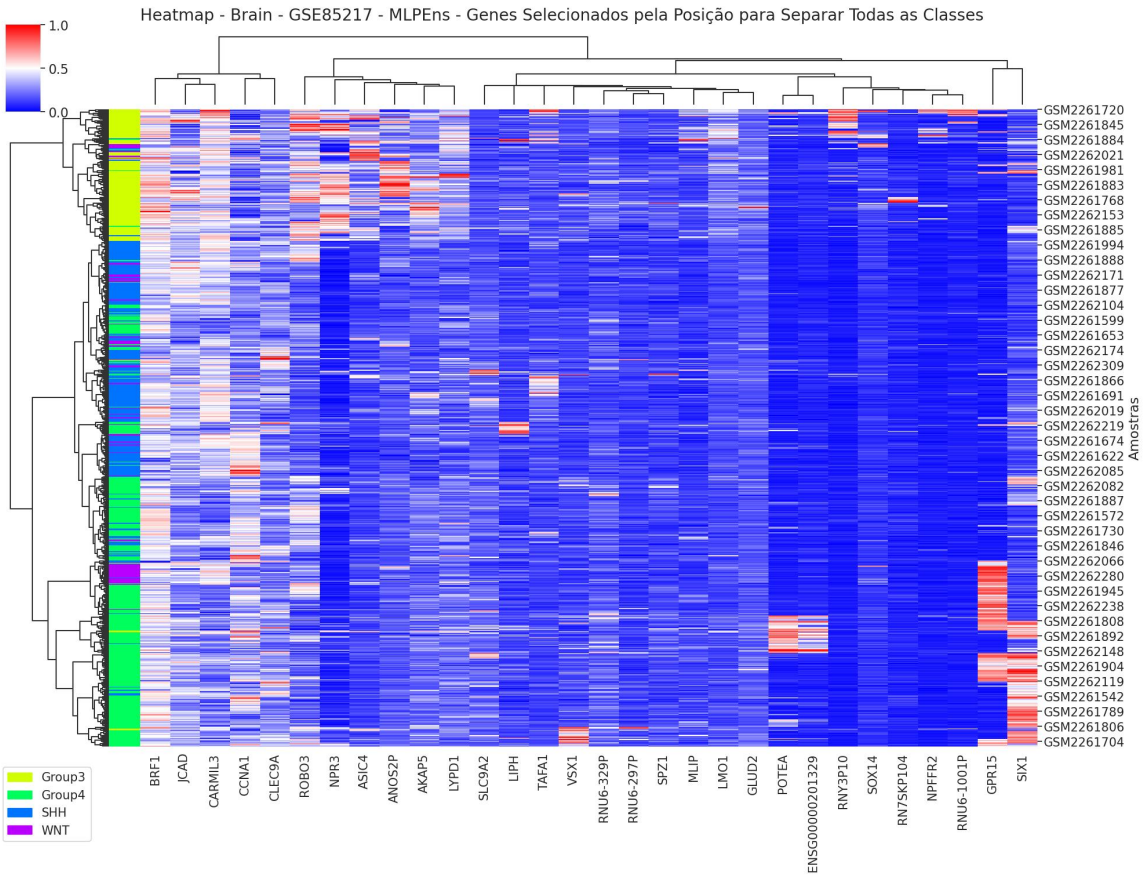

Figura 55 – MLPEns pela Posição - Heatmap.

Tabela 35 – MLPEns pelo Coeficiente - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
POTEA	**21634.0**	13.5	21641	21582	21637.5	21639.0	21640.0	**0.168**	0.042	0.258	0.116	0.141	0.152	0.199
RN7SKP104	**21439.4**	584.0	21641	19124	21612.0	21632.0	21638.0	**0.147**	0.061	0.347	0.066	0.115	0.143	0.158
RNY3P10	**21614.0**	56.6	21640	21375	21619.2	21629.5	21638.2	**0.139**	0.030	0.196	0.093	0.118	0.128	0.161
ENSG00000201329	**21561.0**	181.0	21640	20863	21590.8	21631.0	21639.0	**0.138**	0.037	0.215	0.067	0.108	0.131	0.162
NPFFR2	**21523.2**	345.5	21641	20042	21585.0	21633.5	21637.0	**0.132**	0.029	0.187	0.071	0.108	0.131	0.152
VSX1	**21490.4**	274.0	21640	20520	21536.0	21608.5	21632.0	**0.124**	0.038	0.201	0.070	0.096	0.115	0.144
RNU6-297P	**21493.2**	212.6	21639	20869	21448.2	21605.0	21634.2	**0.119**	0.033	0.192	0.077	0.094	0.113	0.144
CDH19	**21017.5**	970.3	21641	18219	20587.0	21564.0	21633.8	**0.116**	0.048	0.241	0.057	0.069	0.110	0.160
RNU6-1001P	**21160.0**	1212.7	21641	16298	21439.8	21577.0	21624.0	**0.114**	0.036	0.198	0.055	0.091	0.105	0.135
NPR3	**21538.0**	147.0	21637	21052	21551.8	21589.5	21616.2	**0.114**	0.023	0.166	0.081	0.097	0.110	0.121
CBX3P6	**20904.7**	2108.9	21634	12064	21342.8	21580.5	21622.2	**0.107**	0.030	0.157	0.048	0.091	0.103	0.130
SPZ1	**21232.1**	989.6	21637	17145	21493.2	21592.5	21605.2	**0.107**	0.025	0.159	0.058	0.095	0.112	0.119
SLC9A2	**21423.5**	324.2	21629	20243	21395.2	21568.0	21602.2	**0.107**	0.023	0.147	0.063	0.086	0.108	0.127
MLIP	**21375.0**	432.9	21634	19873	21371.8	21539.5	21603.2	**0.106**	0.027	0.163	0.070	0.084	0.098	0.127
RN7SL761P	**20526.4**	2604.2	21638	10312	21254.0	21590.0	21610.8	**0.106**	0.034	0.180	0.047	0.084	0.109	0.124
TSPAN8	**20740.0**	1438.5	21640	16076	20140.5	21564.0	21625.0	**0.106**	0.042	0.210	0.055	0.073	0.093	0.128
TAFA1	**21411.5**	359.0	21638	20151	21473.2	21548.0	21605.8	**0.105**	0.022	0.153	0.072	0.086	0.103	0.118
XAGE5	**20823.0**	1580.5	21638	15812	21213.8	21569.0	21628.2	**0.104**	0.032	0.173	0.054	0.080	0.104	0.125
RNU6-525P	**18190.9**	5344.6	21641	4691	15305.0	21410.5	21631.8	**0.103**	0.052	0.218	0.036	0.053	0.098	0.147
CLEC9A	**21308.2**	832.5	21629	17725	21383.8	21556.5	21597.5	**0.103**	0.022	0.146	0.057	0.087	0.108	0.116
LMO1	**21464.2**	204.5	21634	20835	21462.8	21538.0	21595.2	**0.103**	0.018	0.143	0.079	0.089	0.099	0.115
RN7SL58P	**20709.0**	1896.1	21641	13739	21157.0	21515.0	21621.8	**0.103**	0.031	0.159	0.050	0.082	0.099	0.121
LYPD1	**21418.4**	389.1	21639	19834	21471.8	21513.0	21596.8	**0.102**	0.015	0.137	0.069	0.094	0.102	0.108
RNU6-329P	**21424.4**	244.0	21629	20718	21400.0	21508.0	21604.8	**0.102**	0.020	0.141	0.079	0.084	0.096	0.118
AKAP5	**21433.4**	261.8	21622	20526	21373.0	21549.0	21594.5	**0.102**	0.017	0.137	0.076	0.086	0.103	0.112
GPR15	**21275.1**	612.6	21627	19071	21373.5	21518.0	21599.0	**0.100**	0.019	0.126	0.061	0.088	0.103	0.114
ROBO3	**21404.8**	272.5	21635	20619	21373.5	21523.5	21567.8	**0.100**	0.017	0.125	0.071	0.091	0.101	0.110
GLUD2	**21230.4**	530.8	21640	19743	21155.5	21484.5	21598.8	**0.099**	0.023	0.155	0.064	0.080	0.098	0.113
P2RY2	**20777.4**	1714.1	21633	14403	21108.8	21391.5	21594.0	**0.097**	0.028	0.146	0.051	0.076	0.089	0.123
ANOS2P	**21243.4**	595.8	21626	19568	21308.2	21466.5	21565.0	**0.097**	0.019	0.130	0.062	0.085	0.095	0.108

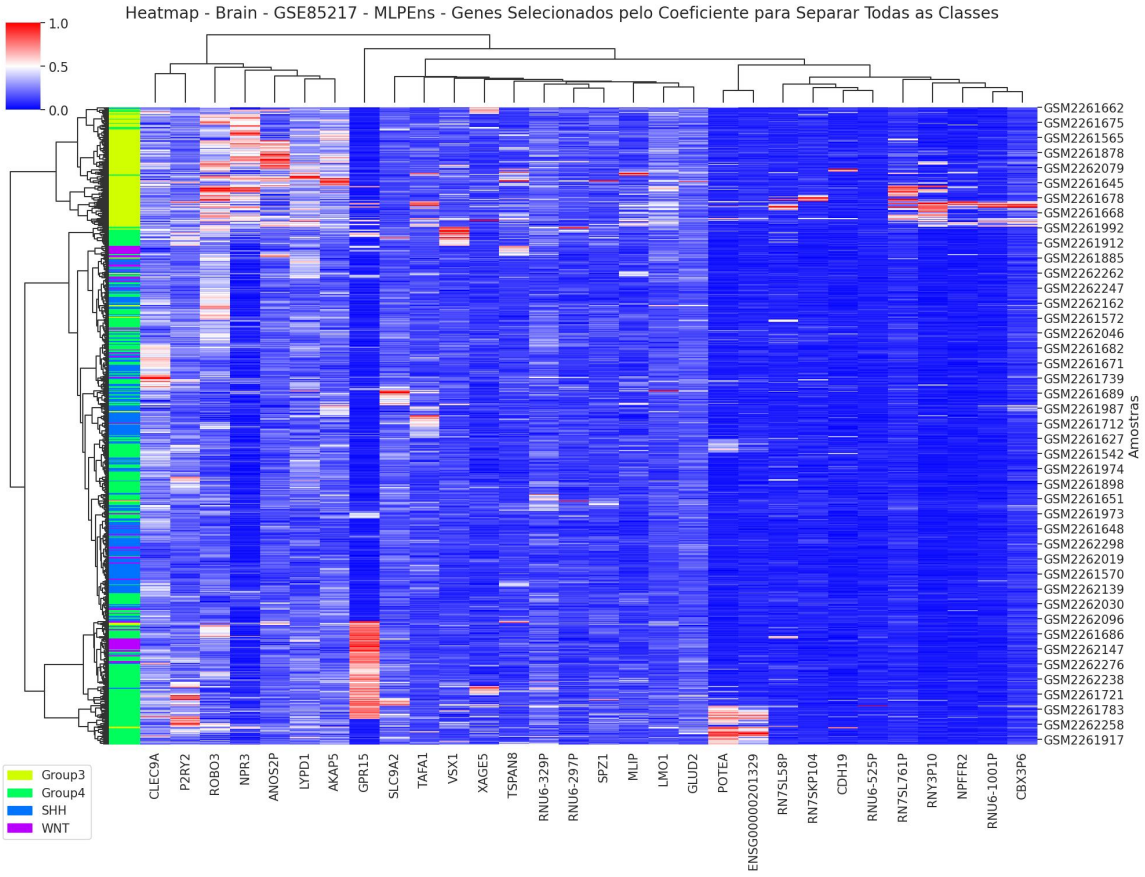

Figura 56 – MLPEns pelo Coeficiente - Melhores Genes - Heatmap.

Como podemos perceber, a separação de grupos não ficou evidente nos *heatmaps* e os genes no topo da tabela de genes (selecionados como importantes) apresentam alto desvio padrão. Para que seja vista uma separação nos *heatmaps* de medulo-blastoma, dos genes selecionados pela MLPEns, é necessário considerar um número maior de genes, cerca de 100 - os *heatmaps* com 100 genes selecionados como mais importantes pela MLPEns pelo critério de posição e coeficientes são mostrados nas Figuras 57 e 58, respectivamente.

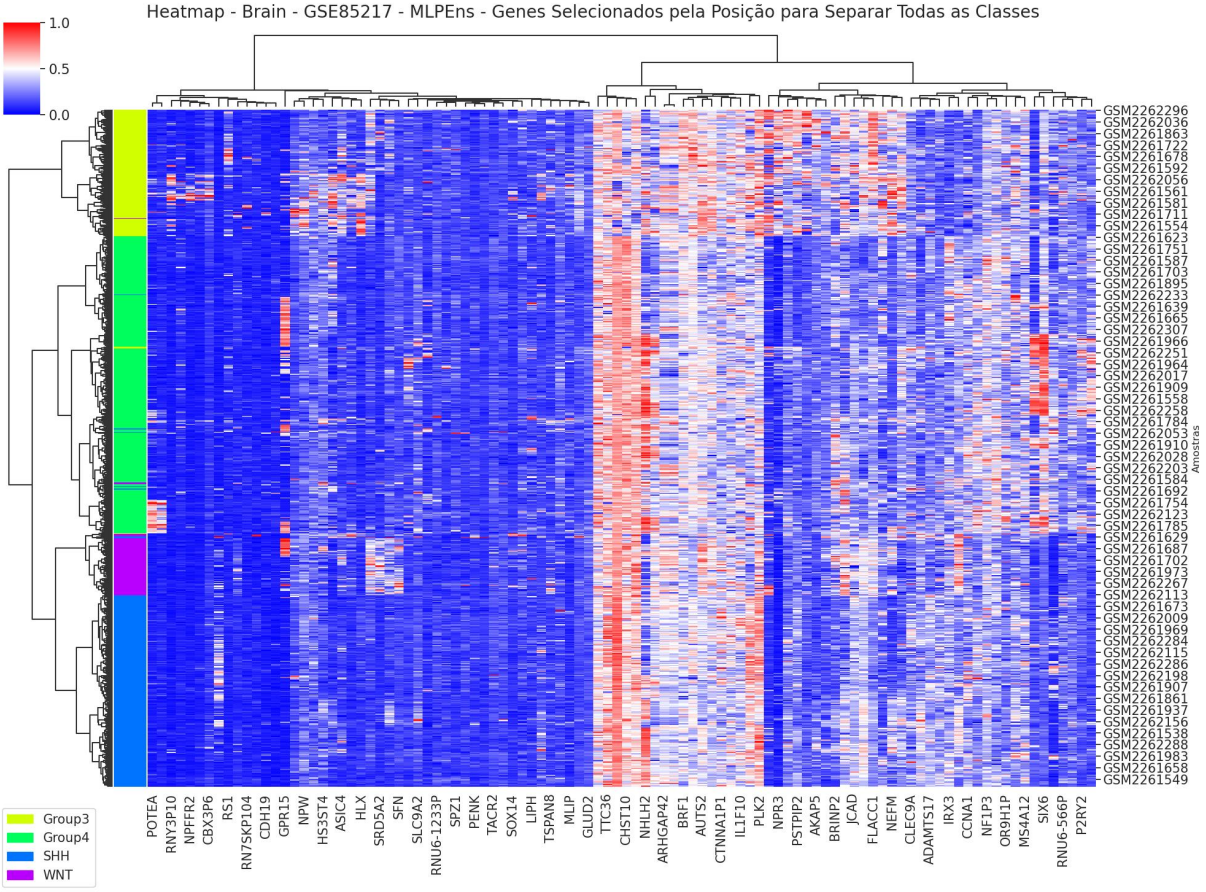

Figura 57 – MLPEns pela Posição - Melhores Genes - Heatmap.

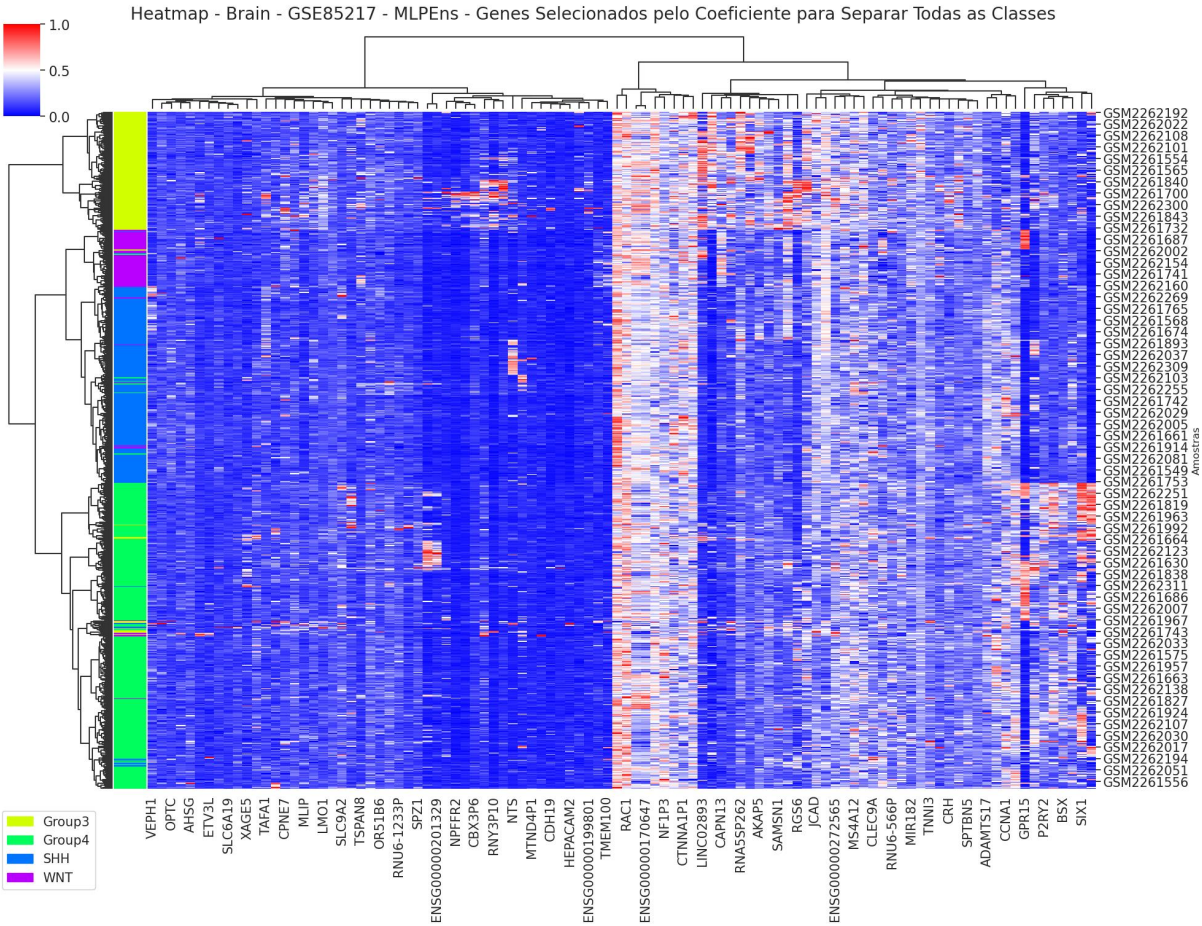

Figura 58 – MLPEns pelo Coeficiente - Melhores Genes - Heatmap.

Isso nos indica que analisar **apenas os pesos da primeira camada da rede neural é insuficiente**. Para ter uma melhor compreensão de genes importantes, em trabalhos futuros, pensamos em utilizar abordagens que investigam os gradientes em todas as camadas da rede neural como o DeepLift (SHRIKUMAR; GREENSIDE; KUNDAJE, 2017).

8.2.1.3 SVM - Biomarcadores por Classe

Como o **SVM** é implementado de forma que um classificador, em problemas multi-classes de $nr\_classes$, seja composto de $nr\_classes$ **modelos OvR** (One versus Rest - uma classe versus as demais), podemos selecionar, também, um conjunto de **genes biomarcadores para cada classe** estudada. As Tabelas 36, 37, 38 e 39 mostram os genes selecionados como importantes para separar as classes *Group 3*, *Group 4*, *SHH* e *WNT*, respectivamente, usando o critério de posição.

Tabela 36 – Separação da classe Group 3 - SVM pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
NPR3	**21640.2**	1.3	21641	21636	21640.0	21641.0	21641.0	**0.419**	0.046	0.516	0.333	0.383	0.420	0.443
GPR15	**21636.7**	6.2	21641	21618	21637.8	21639.0	21640.0	**0.369**	0.055	0.501	0.264	0.340	0.369	0.401
GABRA5	**21628.6**	20.0	21640	21562	21631.0	21636.5	21638.0	**0.319**	0.043	0.398	0.223	0.301	0.328	0.351
RNY3P10	**21624.2**	15.5	21640	21582	21617.0	21628.5	21637.0	**0.295**	0.039	0.360	0.225	0.264	0.294	0.330
POTEA	**21622.4**	34.1	21641	21495	21628.2	21633.0	21637.5	**0.312**	0.056	0.417	0.193	0.283	0.298	0.351
NHLH2	**21619.7**	27.4	21640	21518	21611.8	21632.0	21638.0	**0.298**	0.055	0.381	0.206	0.249	0.288	0.348
NEFM	**21619.0**	33.3	21639	21485	21619.2	21630.5	21635.8	**0.291**	0.040	0.359	0.195	0.278	0.289	0.321
PAX6	**21618.0**	42.2	21639	21446	21622.5	21631.0	21637.0	**0.297**	0.046	0.392	0.191	0.263	0.299	0.331
PPP2R2B	**21609.6**	19.7	21635	21559	21599.5	21613.5	21626.2	**0.263**	0.028	0.326	0.210	0.242	0.262	0.281
SLC16A4	**21608.8**	34.5	21637	21488	21607.2	21624.0	21629.2	**0.269**	0.034	0.329	0.210	0.247	0.270	0.298

Tabela 37 – Separação da classe Group 4 - SVM pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
LMX1A	**21639.3**	1.6	21641	21636	21638.8	21639.5	21641.0	**0.402**	0.047	0.510	0.329	0.365	0.397	0.445
RBM24	**21636.8**	4.0	21641	21629	21635.2	21638.0	21640.0	**0.366**	0.040	0.440	0.299	0.340	0.363	0.400
GPR15	**21636.6**	7.5	21641	21610	21636.8	21639.5	21641.0	**0.383**	0.063	0.500	0.243	0.354	0.375	0.418
TFAP2D	**21634.2**	7.3	21641	21613	21634.0	21636.0	21638.2	**0.345**	0.051	0.487	0.248	0.321	0.345	0.366
POTEA	**21633.8**	9.9	21641	21597	21633.5	21637.0	21639.2	**0.359**	0.062	0.483	0.231	0.308	0.352	0.422
SIX1	**21631.6**	8.9	21639	21604	21629.0	21635.0	21638.0	**0.326**	0.047	0.405	0.241	0.282	0.335	0.367
SIX6	**21631.4**	4.3	21637	21619	21630.0	21632.5	21634.2	**0.313**	0.028	0.394	0.266	0.294	0.307	0.331
ROBO1	**21629.6**	7.9	21640	21609	21626.2	21632.0	21634.8	**0.311**	0.029	0.382	0.258	0.290	0.312	0.332
NPR3	**21623.2**	15.2	21640	21587	21614.5	21630.0	21634.5	**0.295**	0.036	0.357	0.230	0.266	0.299	0.317
ZIC2	**21621.0**	24.0	21640	21561	21614.8	21632.5	21637.0	**0.309**	0.057	0.421	0.217	0.261	0.311	0.348

Tabela 38 – Separação da classe SHH - SVM pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
NEUROG1	**21640.2**	1.0	21641	21638	21639.0	21640.5	21641.0	**0.441**	0.042	0.533	0.384	0.409	0.430	0.468
PEX5L	**21638.7**	2.2	21641	21632	21638.0	21639.0	21640.0	**0.417**	0.026	0.482	0.370	0.397	0.416	0.437
ATOH1	**21638.7**	1.9	21641	21634	21637.8	21639.0	21640.0	**0.423**	0.047	0.533	0.372	0.386	0.402	0.470
TBR1	**21634.5**	5.0	21641	21621	21632.8	21635.0	21637.0	**0.395**	0.038	0.478	0.322	0.364	0.400	0.419
OTX2	**21634.1**	4.8	21640	21625	21632.0	21635.5	21638.0	**0.389**	0.040	0.458	0.322	0.360	0.383	0.422
CYYR1	**21634.1**	3.0	21639	21629	21631.8	21634.5	21636.2	**0.384**	0.026	0.431	0.342	0.362	0.387	0.399
ZFHX4	**21633.2**	5.4	21640	21620	21628.0	21634.0	21638.0	**0.389**	0.045	0.465	0.321	0.353	0.371	0.436
NDP	**21632.8**	4.1	21640	21623	21629.8	21633.0	21636.0	**0.372**	0.026	0.432	0.324	0.360	0.372	0.390
FBXL21P	**21632.6**	3.5	21637	21623	21631.0	21633.0	21635.2	**0.373**	0.016	0.412	0.351	0.363	0.369	0.378
PLPPR4	**21631.0**	5.2	21638	21617	21629.0	21631.0	21635.0	**0.370**	0.023	0.418	0.321	0.357	0.378	0.381

Tabela 39 – Separação da classe WNT - SVM pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
WIF1	**21639.8**	1.4	21641	21636	21640.0	21640.0	21641.0	**0.407**	0.032	0.479	0.361	0.382	0.404	0.423
PAX3	**21638.8**	7.6	21641	21606	21640.0	21641.0	21641.0	**0.429**	0.037	0.471	0.307	0.414	0.439	0.456
TMEM51	**21637.0**	2.4	21640	21630	21636.0	21637.5	21639.0	**0.371**	0.019	0.404	0.328	0.357	0.370	0.385
NOL4	**21634.8**	4.2	21640	21622	21633.8	21635.0	21638.0	**0.358**	0.025	0.425	0.318	0.337	0.354	0.370
DLX3	**21632.2**	5.9	21640	21619	21627.0	21633.5	21637.2	**0.347**	0.031	0.401	0.305	0.324	0.337	0.374
TMEM51-AS1	**21632.0**	6.7	21639	21610	21629.0	21634.5	21636.0	**0.349**	0.023	0.393	0.312	0.333	0.345	0.359
TMEM132C	**21631.7**	5.0	21641	21625	21627.5	21631.5	21635.0	**0.345**	0.034	0.428	0.304	0.318	0.341	0.351
BEND7	**21628.7**	9.2	21638	21604	21621.8	21633.0	21636.0	**0.336**	0.027	0.381	0.290	0.307	0.343	0.356
NKD1	**21628.0**	5.6	21640	21620	21622.8	21627.5	21631.5	**0.326**	0.028	0.391	0.285	0.304	0.322	0.343
PDE11A	**21625.3**	9.2	21638	21603	21620.8	21628.0	21631.0	**0.321**	0.033	0.389	0.267	0.296	0.322	0.340

8.2.1.4 SVM e MLPEns - tSNEs

A seguir, apresentamos **tSNEs** usando apenas os genes selecionados como mais importantes para separação dos subgrupos moleculares. Nas Figuras 59 e 60 são mostrados os tSNEs usando os 10 genes mais importantes selecionados pelo SVM e pela MLPEns, respectivamente.

Figura 59 – SVM - 10 genes - tSNE.

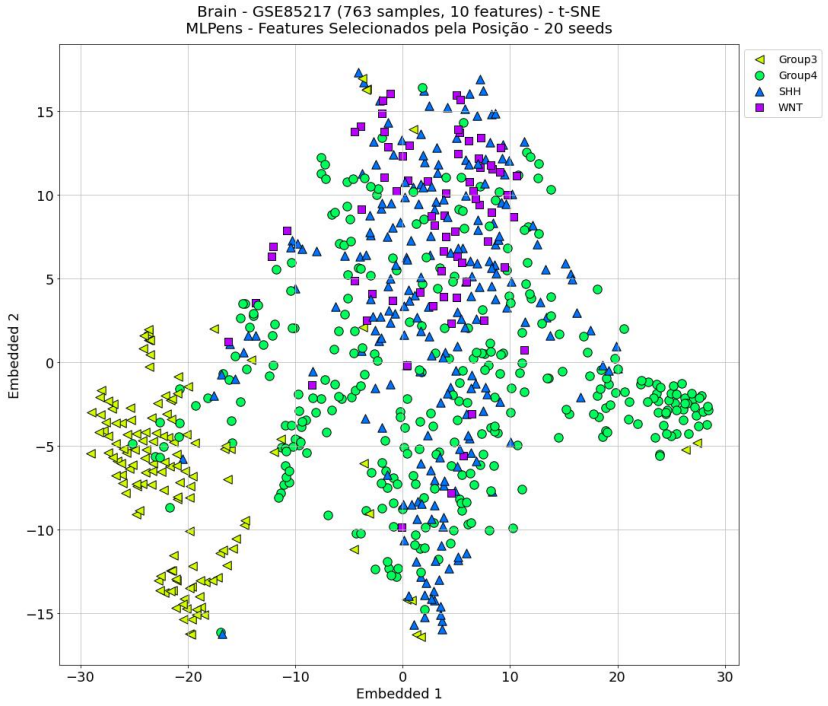

Figura 60 – MLPEns - 10 genes - tSNE.

Nas Figuras 61 e 62 apresentamos os tSNEs usando 30 genes mais importantes selecionados pelo SVM e pela MLPEns, respectivamente.

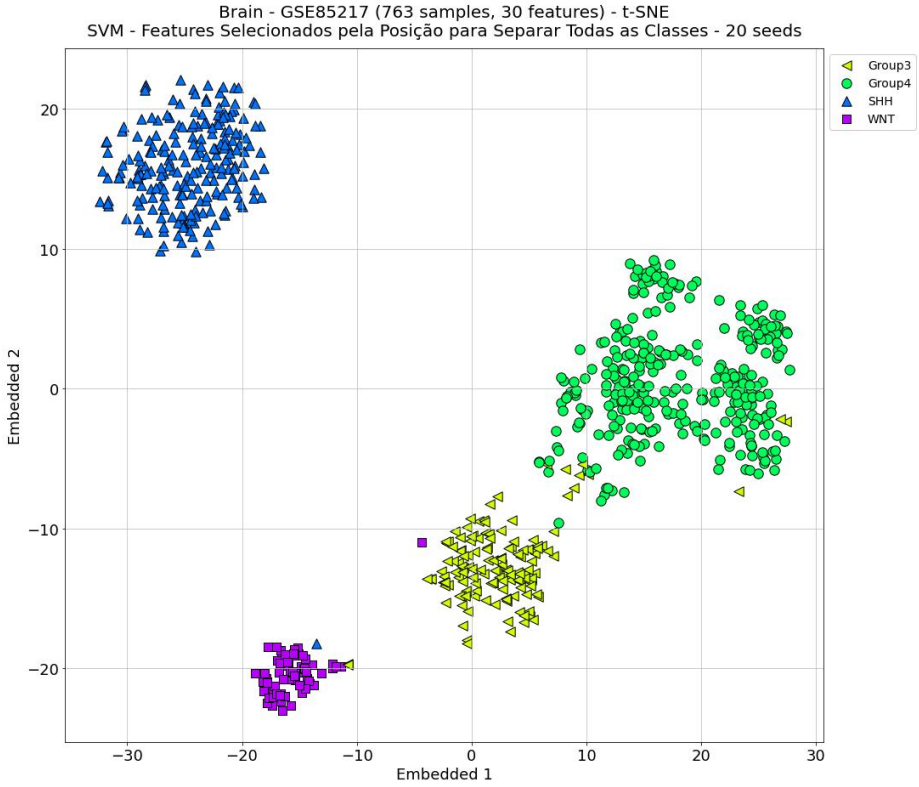

Figura 61 – SVM - 30 genes - tSNE.

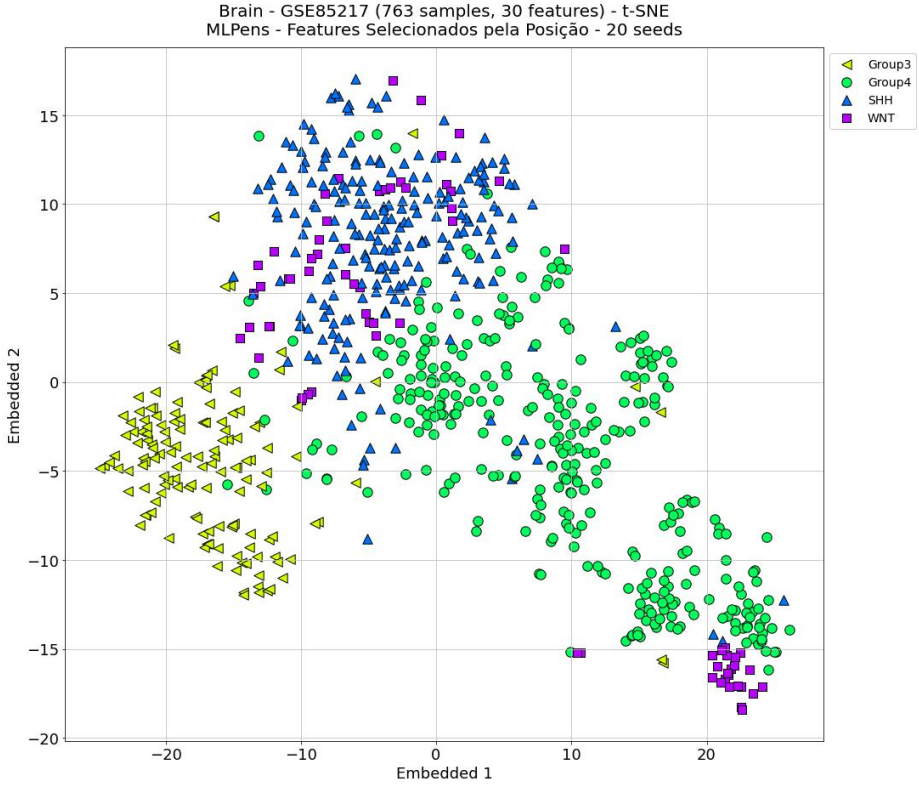

Figura 62 – MLPEns - 30 genes - tSNE.

Nas Figuras 63 e 64 apresentamos os tSNEs usando 100 genes mais importantes selecionados pelo SVM e pela MLPEns, respectivamente.

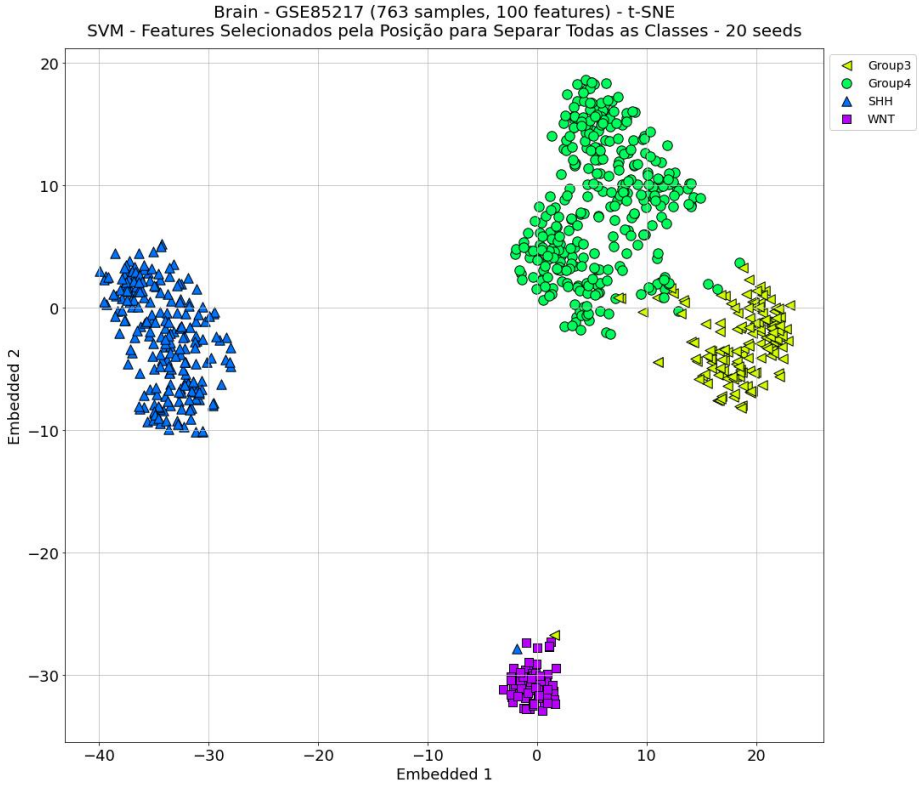

Figura 63 – SVM - 100 genes - tSNE.

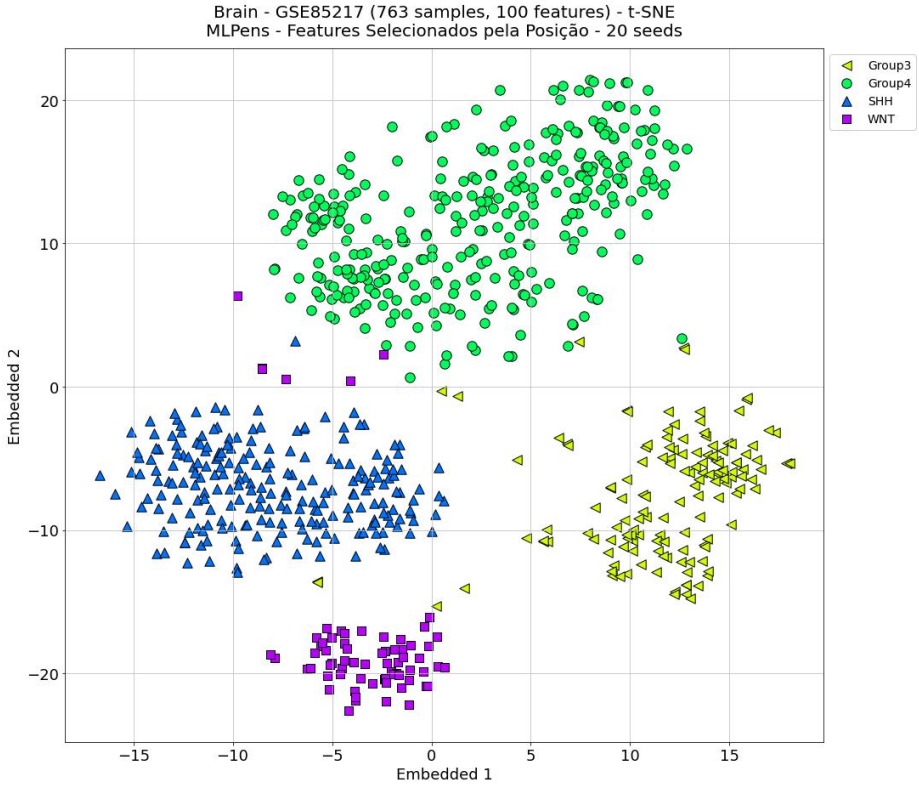

Figura 64 – MLPEns - 100 genes - tSNE.

Nas Figuras 65 e 66 apresentamos os tSNEs usando 300 genes mais importantes selecionados pelo SVM e pela MLPEns, respectivamente.

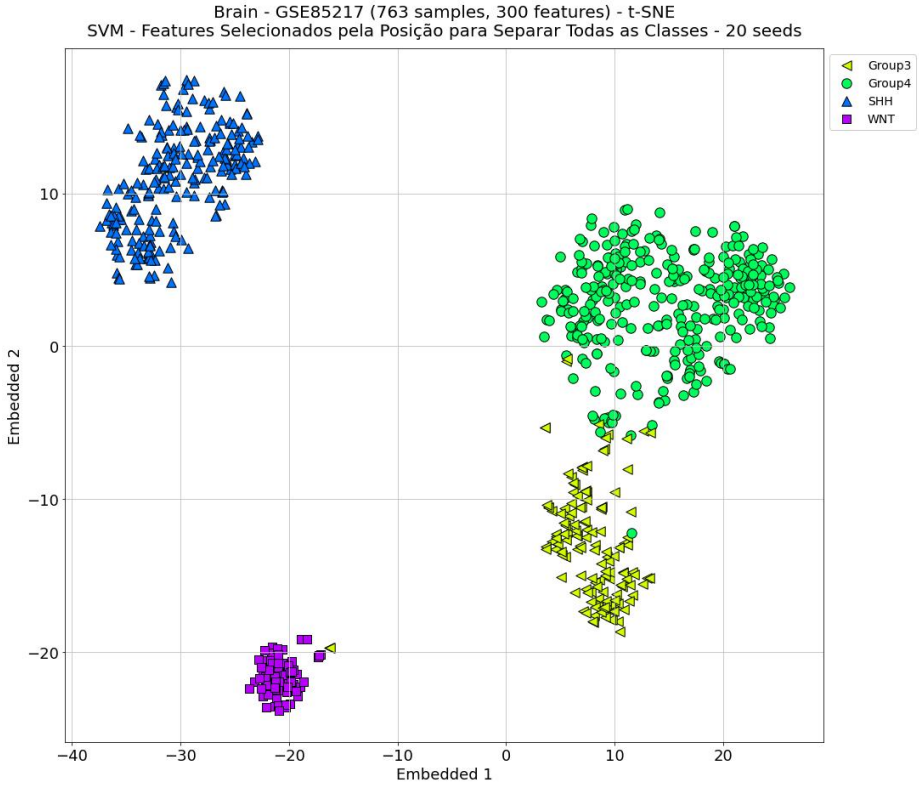

Figura 65 – SVM - 300 genes - tSNE.

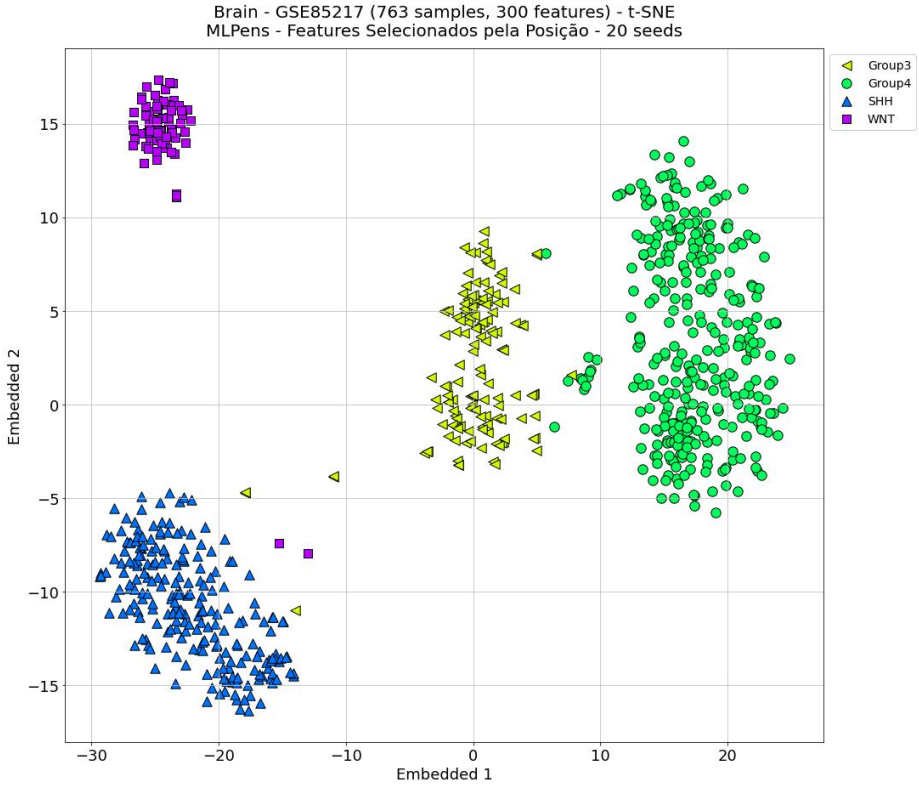

Figura 66 – MLPEns - 300 genes - tSNE.

8.2.2 Sarcoma de Ewing - (1,4) - GSE2553

Nessa seção, investigamos os genes selecionados para separar 19 amostra de Sarcoma de Ewing de outros tumores (38 amostras de *malignant fibrous histiocytoma*; 33 de *liposarcoma*; 17 de *leiomyosarcoma*; 16 de *synovial cell sarcoma*; 10 de *sarcoma NOS (Not Otherwise Specified)*; 7 de *fibrosarcoma*; 6 de *malignant peripheral nerve sheath tumor*; 6 de *rhabdomyosarcoma*; 6 de *malignant hemangiopericytoma*; 5 de *dermatofibrosarcoma*; 5 de *gastrointestinal stromal tumor*; e 5 de *osteosarcoma*).

A Figura 67 apresenta o tSNE do dataset com todos os genes disponíveis (os tipos tumorais com menos de 5 amostras foram removidos). As amostras de Sarcoma de Ewing são mostradas como losangos verdes.

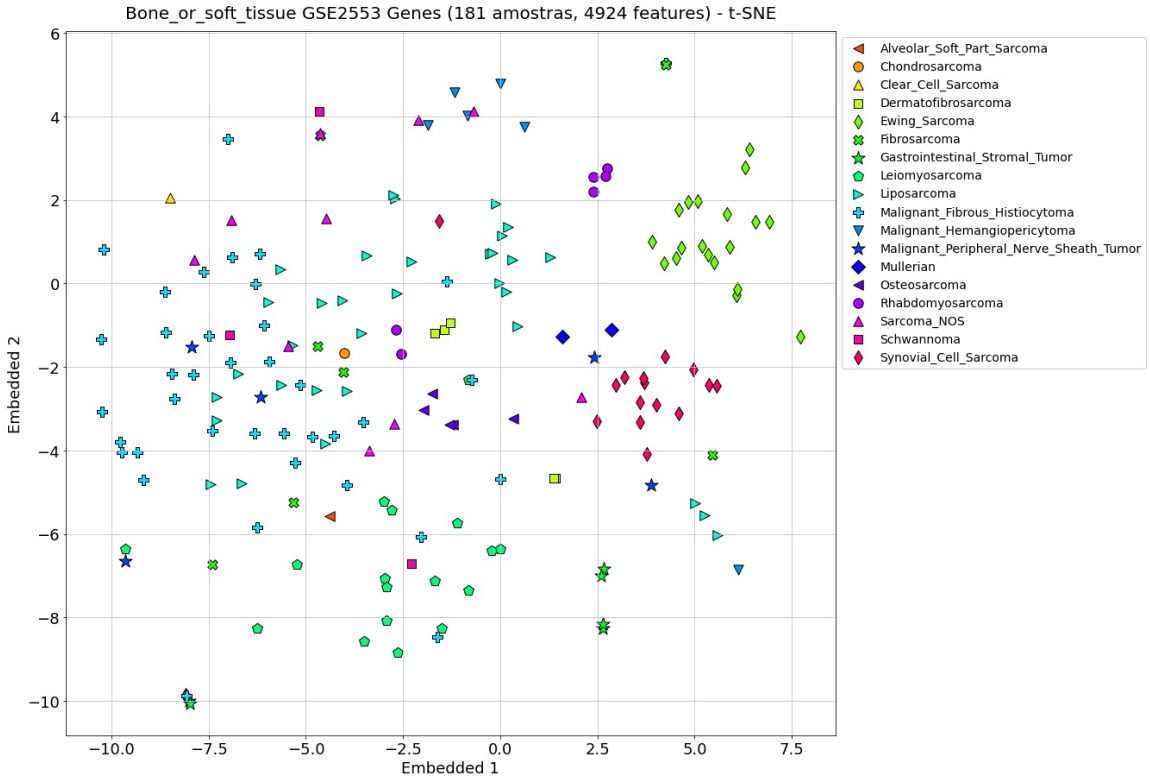

Figura 67 – tSNE - Sarcoma de Ewing e outros tumores - todos features (4.924 genes).

8.2.2.1 SVM

As Tabelas 40 e 41 mostram os 30 genes selecionados pelo SVM, expostos de forma decrescente em importância. Na Tabela 40, empregou-se o critério de posição; enquanto que, na 41, empregou-se o critério de coeficientes. As Figuras 68 e 69 apresentam os heatmap (Agrupamento Hierárquico) usando os 30 genes selecionados pela posição e coeficiente, respectivamente. As tabelas foram montadas combinando a saída de 20 execuções.

Tabela 40 – SVM pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
ATP1A1	**4922.8**	1.0	4924	4921	4922.0	4923.0	4924.0	**2.568**	0.261	3.065	2.144	2.390	2.604	2.694
DYRK3	**4921.8**	1.6	4924	4919	4920.8	4922.0	4923.0	**2.357**	0.135	2.659	2.081	2.299	2.375	2.430
KDSR	**4919.8**	8.4	4924	4885	4920.8	4922.5	4923.2	**2.381**	0.382	3.167	1.520	2.106	2.384	2.697
GLG1	**4911.3**	10.8	4923	4878	4905.5	4915.0	4919.0	**1.952**	0.306	2.545	1.471	1.652	1.897	2.209
FCGRT	**4910.2**	16.4	4924	4865	4903.8	4918.0	4922.0	**2.062**	0.471	2.893	1.357	1.660	2.032	2.452
LOC399959	**4909.4**	15.4	4921	4866	4907.0	4914.0	4918.5	**1.859**	0.248	2.299	1.305	1.730	1.877	2.021
AKAP7	**4906.6**	25.1	4924	4816	4902.8	4916.0	4920.2	**1.966**	0.372	2.736	1.212	1.648	1.992	2.259
CDK6	**4899.6**	21.8	4919	4846	4898.5	4905.0	4915.2	**1.713**	0.214	2.037	1.255	1.641	1.736	1.853
DCC	**4894.4**	18.9	4922	4842	4887.2	4896.0	4901.2	**1.648**	0.257	2.284	1.308	1.518	1.570	1.708
NPY	**4892.8**	22.0	4924	4842	4881.0	4888.0	4915.5	**1.705**	0.401	2.579	1.307	1.460	1.552	1.798
DNAL4	**4892.0**	34.7	4918	4801	4888.8	4908.0	4913.0	**1.685**	0.275	2.104	1.136	1.509	1.715	1.914
RCOR1	**4891.8**	17.3	4917	4858	4881.8	4894.0	4907.8	**1.603**	0.192	2.016	1.305	1.485	1.545	1.774
RNF141	**4889.8**	25.5	4914	4825	4884.0	4898.5	4908.0	**1.610**	0.211	1.902	1.183	1.493	1.610	1.777
ROCK2	**4889.8**	27.9	4920	4828	4872.0	4900.5	4911.2	**1.601**	0.253	2.099	1.150	1.407	1.643	1.768
TBK1	**4888.3**	28.9	4914	4802	4880.5	4902.0	4906.0	**1.587**	0.197	1.897	1.165	1.490	1.610	1.736
ADRB1	**4887.6**	47.8	4920	4734	4891.2	4905.0	4913.0	**1.683**	0.321	2.329	1.048	1.544	1.685	1.901
GLI3	**4884.6**	41.3	4918	4755	4870.2	4905.0	4908.2	**1.596**	0.262	2.042	1.062	1.460	1.650	1.741
ECEL1	**4883.8**	56.2	4921	4683	4879.8	4910.0	4918.0	**1.681**	0.353	2.217	0.915	1.463	1.739	1.929
LECT1	**4883.8**	41.1	4917	4778	4875.2	4898.5	4912.0	**1.629**	0.294	2.133	1.082	1.440	1.628	1.828
CITED2	**4880.4**	51.0	4921	4688	4873.2	4893.0	4912.2	**1.627**	0.347	2.337	0.927	1.403	1.545	1.964
TIPARP	**4878.2**	36.3	4914	4765	4865.0	4888.0	4904.0	**1.524**	0.238	1.941	1.055	1.388	1.529	1.728
PCNXL2	**4876.0**	36.5	4919	4790	4851.8	4889.5	4909.2	**1.531**	0.276	1.993	1.152	1.281	1.498	1.746
KCTD15	**4875.8**	22.0	4915	4826	4860.0	4880.5	4887.0	**1.463**	0.182	2.036	1.208	1.324	1.446	1.545
EXOSC7	**4875.8**	29.3	4916	4796	4867.5	4876.0	4893.5	**1.482**	0.208	1.923	1.105	1.364	1.458	1.571
FCHSD1	**4875.3**	29.8	4910	4810	4858.5	4886.0	4900.0	**1.483**	0.203	1.841	1.150	1.318	1.502	1.655
CCK	**4874.8**	54.4	4920	4681	4857.0	4892.0	4912.2	**1.574**	0.330	2.045	0.938	1.313	1.506	1.840
EIF4E2	**4874.2**	17.1	4897	4828	4863.8	4875.0	4887.5	**1.429**	0.113	1.674	1.220	1.337	1.434	1.520
TRIM35	**4874.2**	43.4	4919	4753	4862.8	4890.0	4905.0	**1.527**	0.254	1.981	1.041	1.334	1.556	1.677
PRKCG	**4873.2**	39.1	4909	4744	4868.5	4885.0	4898.0	**1.475**	0.191	1.752	1.016	1.395	1.500	1.601
TLE1	**4871.8**	42.4	4904	4716	4871.8	4881.5	4897.5	**1.467**	0.184	1.754	0.973	1.422	1.457	1.592

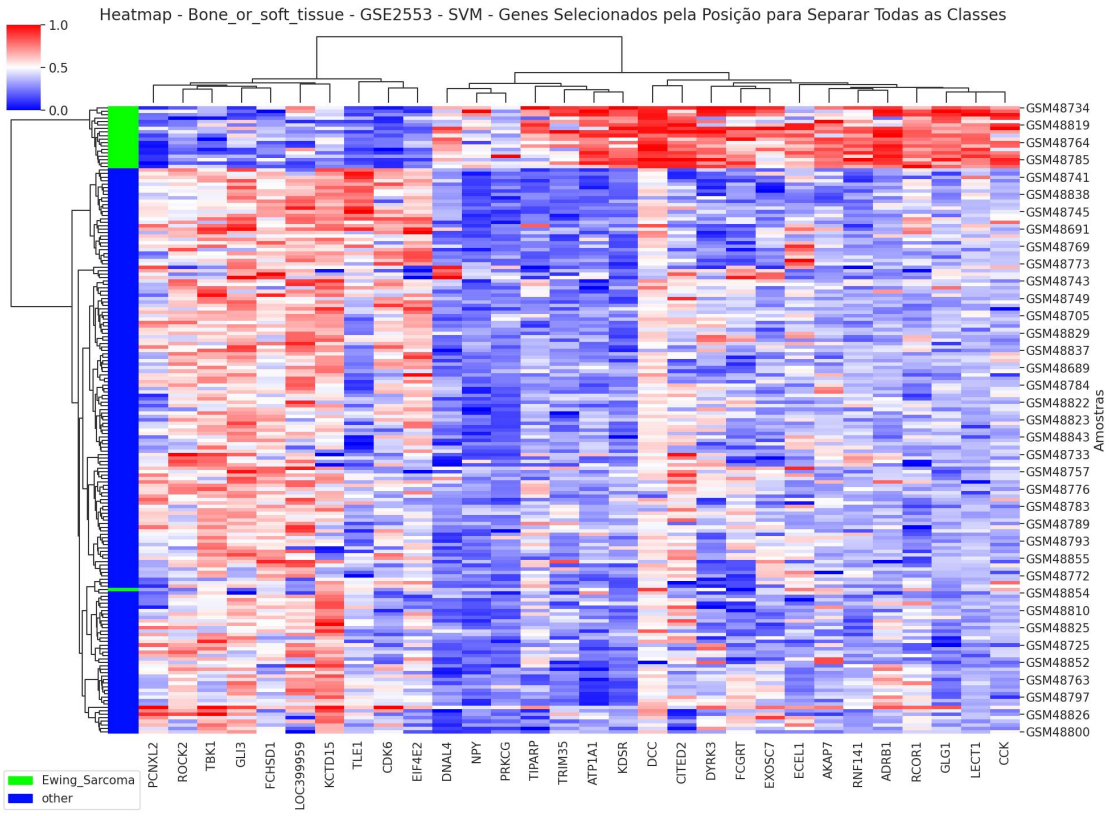

Figura 68 – SVM pela Posição - Melhores Genes- Heatmap.

Tabela 41 – SVM pelo Coeficiente - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
ATP1A1	**4922.8**	1.0	4924	4921	4922.0	4923.0	4924.0	**2.568**	0.261	3.065	2.144	2.390	2.604	2.694
KDSR	**4919.8**	8.4	4924	4885	4920.8	4922.5	4923.2	**2.381**	0.382	3.167	1.520	2.106	2.384	2.697
DYRK3	**4921.8**	1.6	4924	4919	4920.8	4922.0	4923.0	**2.357**	0.135	2.659	2.081	2.299	2.375	2.430
FCGRT	**4910.2**	16.4	4924	4865	4903.8	4918.0	4922.0	**2.062**	0.471	2.893	1.357	1.660	2.032	2.452
AKAP7	**4906.6**	25.1	4924	4816	4902.8	4916.0	4920.2	**1.966**	0.372	2.736	1.212	1.648	1.992	2.259
GLG1	**4911.3**	10.8	4923	4878	4905.5	4915.0	4919.0	**1.952**	0.306	2.545	1.471	1.652	1.897	2.209
LOC399959	**4909.4**	15.4	4921	4866	4907.0	4914.0	4918.5	**1.859**	0.248	2.299	1.305	1.730	1.877	2.021
CDK6	**4899.6**	21.8	4919	4846	4898.5	4905.0	4915.2	**1.713**	0.214	2.037	1.255	1.641	1.736	1.853
NPY	**4892.8**	22.0	4924	4842	4881.0	4888.0	4915.5	**1.705**	0.401	2.579	1.307	1.460	1.552	1.798
DNAL4	**4892.0**	34.7	4918	4801	4888.8	4908.0	4913.0	**1.685**	0.275	2.104	1.136	1.509	1.715	1.914
ADRB1	**4887.6**	47.8	4920	4734	4891.2	4905.0	4913.0	**1.683**	0.321	2.329	1.048	1.544	1.685	1.901
ECEL1	**4883.8**	56.2	4921	4683	4879.8	4910.0	4918.0	**1.681**	0.353	2.217	0.915	1.463	1.739	1.929
DCC	**4894.4**	18.9	4922	4842	4887.2	4896.0	4901.2	**1.648**	0.257	2.284	1.308	1.518	1.570	1.708
LECT1	**4883.8**	41.1	4917	4778	4875.2	4898.5	4912.0	**1.629**	0.294	2.133	1.082	1.440	1.628	1.828
CITED2	**4880.4**	51.0	4921	4688	4873.2	4893.0	4912.2	**1.627**	0.347	2.337	0.927	1.403	1.545	1.964
OAS3	**4871.5**	67.2	4922	4702	4836.5	4910.0	4917.2	**1.621**	0.385	2.152	0.941	1.206	1.709	1.977
RNF141	**4889.8**	25.5	4914	4825	4884.0	4898.5	4908.0	**1.610**	0.211	1.902	1.183	1.493	1.610	1.777
RCOR1	**4891.8**	17.3	4917	4858	4881.8	4894.0	4907.8	**1.603**	0.192	2.016	1.305	1.485	1.545	1.774
ROCK2	**4889.8**	27.9	4920	4828	4872.0	4900.5	4911.2	**1.601**	0.253	2.099	1.150	1.407	1.643	1.768
GLI3	**4884.6**	41.3	4918	4755	4870.2	4905.0	4908.2	**1.596**	0.262	2.042	1.062	1.460	1.650	1.741
TBK1	**4888.3**	28.9	4914	4802	4880.5	4902.0	4906.0	**1.587**	0.197	1.897	1.165	1.490	1.610	1.736
CCK	**4874.8**	54.4	4920	4681	4857.0	4892.0	4912.2	**1.574**	0.330	2.045	0.938	1.313	1.506	1.840
GYG2	**4871.7**	39.3	4919	4763	4849.0	4869.0	4910.0	**1.557**	0.358	2.271	1.069	1.292	1.382	1.914
OLFM1	**4861.0**	77.9	4921	4673	4826.8	4904.0	4911.0	**1.551**	0.366	2.049	0.900	1.228	1.692	1.782
PCNXL2	**4876.0**	36.5	4919	4790	4851.8	4889.5	4909.2	**1.531**	0.276	1.993	1.152	1.281	1.498	1.746
TRIM35	**4874.2**	43.4	4919	4753	4862.8	4890.0	4905.0	**1.527**	0.254	1.981	1.041	1.334	1.556	1.677
TIPARP	**4878.2**	36.3	4914	4765	4865.0	4888.0	4904.0	**1.524**	0.238	1.941	1.055	1.388	1.529	1.728
FCHSD1	**4875.3**	29.8	4910	4810	4858.5	4886.0	4900.0	**1.483**	0.203	1.841	1.150	1.318	1.502	1.655
EXOSC7	**4875.8**	29.3	4916	4796	4867.5	4876.0	4893.5	**1.482**	0.208	1.923	1.105	1.364	1.458	1.571
DAPK1	**4869.6**	41.9	4915	4750	4845.5	4887.0	4903.2	**1.481**	0.247	1.964	1.020	1.263	1.487	1.697

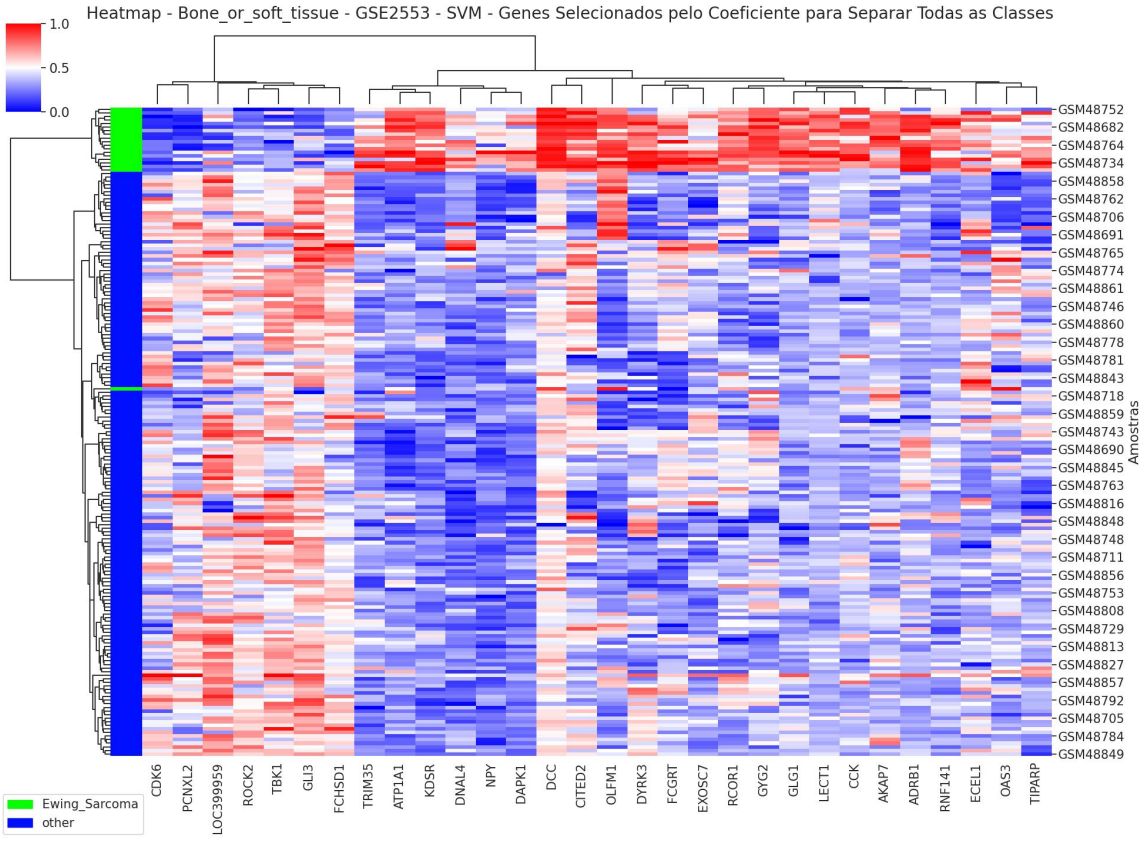

Figura 69 – SVM pelo Coeficiente - Melhores Genes - Heatmap.

8.2.2.2 MLPEns

As Tabelas 42 e 43 mostram os 30 genes selecionados pela MLPEns, expostos de forma decrescente em importância. Na Tabela 42, empregou-se o critério de posição; enquanto que, na 43, empregou-se o critério de coeficientes. As Figuras 70 e 71 apresentam os heatmaps (Agrupamentos Hierárquicos), usando os 30 genes selecionados pela posição e coeficiente, respectivamente.

Tabela 42 – MLPEns pela Posição - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
KDSR	**4920.3**	5.6	4924	4903	4920.8	4922.0	4923.0	**0.741**	0.097	0.950	0.594	0.673	0.723	0.800
DYRK3	**4918.0**	6.7	4924	4900	4915.8	4919.5	4924.0	**0.732**	0.145	1.141	0.591	0.637	0.683	0.777
GLG1	**4916.4**	7.3	4924	4893	4912.8	4916.0	4922.0	**0.697**	0.077	0.833	0.576	0.637	0.694	0.744
ATP1A1	**4914.4**	9.4	4923	4892	4911.0	4917.5	4922.0	**0.687**	0.073	0.828	0.576	0.636	0.671	0.746
FCGRT	**4895.0**	40.7	4924	4744	4894.2	4909.5	4918.2	**0.640**	0.066	0.770	0.531	0.586	0.644	0.685
RNF141	**4892.6**	48.0	4923	4733	4901.5	4908.5	4916.5	**0.650**	0.116	1.013	0.524	0.581	0.615	0.652
DAPK1	**4892.2**	44.5	4924	4737	4893.0	4910.5	4916.5	**0.638**	0.066	0.820	0.525	0.590	0.626	0.660
CCK	**4891.2**	45.3	4923	4767	4884.0	4915.5	4920.5	**0.666**	0.121	0.966	0.498	0.600	0.644	0.704
ENPP1	**4872.1**	74.9	4924	4564	4864.8	4892.5	4904.0	**0.616**	0.081	0.774	0.501	0.551	0.589	0.690
CARHSP1	**4859.8**	45.0	4918	4755	4827.0	4867.5	4898.8	**0.585**	0.051	0.729	0.515	0.549	0.576	0.609
LECT1	**4857.2**	101.7	4918	4468	4864.2	4891.0	4905.5	**0.599**	0.058	0.741	0.516	0.552	0.601	0.629
TRIM35	**4848.6**	79.8	4921	4592	4806.2	4878.5	4911.2	**0.602**	0.085	0.830	0.509	0.529	0.600	0.627
TIPARP	**4845.2**	126.8	4923	4357	4849.8	4876.5	4909.8	**0.601**	0.078	0.854	0.481	0.554	0.592	0.631
ADRB1	**4841.4**	195.9	4921	4002	4856.0	4900.5	4913.8	**0.611**	0.079	0.833	0.486	0.578	0.600	0.621
GRP	**4837.0**	147.6	4924	4231	4825.2	4881.5	4898.2	**0.589**	0.066	0.772	0.481	0.549	0.580	0.614
DCC	**4832.8**	173.7	4922	4228	4879.8	4900.0	4915.0	**0.614**	0.084	0.841	0.443	0.560	0.612	0.662
SLC12A2	**4830.8**	135.3	4921	4318	4841.5	4877.0	4909.5	**0.598**	0.100	0.920	0.460	0.538	0.584	0.636
GLI3	**4828.2**	133.7	4922	4316	4785.0	4885.0	4904.2	**0.599**	0.100	0.878	0.481	0.528	0.570	0.637
FCHSD1	**4826.2**	106.2	4923	4559	4765.0	4877.5	4904.5	**0.588**	0.082	0.811	0.490	0.519	0.578	0.624
CCND1	**4823.5**	168.2	4922	4166	4834.0	4876.5	4911.5	**0.598**	0.070	0.727	0.480	0.563	0.598	0.648
CDH8	**4820.0**	95.2	4919	4586	4793.5	4854.0	4889.5	**0.574**	0.071	0.759	0.485	0.521	0.552	0.592
CITED2	**4815.6**	259.1	4921	3700	4839.8	4885.5	4903.8	**0.598**	0.072	0.761	0.442	0.553	0.587	0.647
AKAP7	**4813.2**	245.5	4921	3769	4805.8	4889.5	4909.8	**0.605**	0.089	0.821	0.462	0.546	0.585	0.627
NPY	**4807.0**	224.3	4920	3933	4855.8	4895.5	4913.8	**0.608**	0.087	0.748	0.430	0.551	0.591	0.665
CDK6	**4802.6**	301.6	4920	3504	4852.2	4885.5	4906.0	**0.598**	0.087	0.780	0.413	0.547	0.576	0.645
SLC39A10	**4782.9**	152.0	4919	4280	4690.2	4825.5	4897.0	**0.573**	0.084	0.839	0.466	0.510	0.555	0.616
IVNS1ABP	**4777.6**	297.6	4917	3809	4826.8	4880.0	4906.0	**0.589**	0.077	0.765	0.433	0.555	0.587	0.640
GYG2	**4776.5**	252.6	4923	4045	4799.8	4875.0	4910.2	**0.600**	0.094	0.848	0.447	0.547	0.588	0.661
EIF1B	**4764.4**	187.1	4921	4216	4683.8	4856.0	4901.2	**0.569**	0.075	0.782	0.456	0.527	0.558	0.594
PCNXL2	**4758.0**	355.2	4918	3277	4799.8	4859.5	4909.2	**0.588**	0.094	0.801	0.416	0.537	0.565	0.640

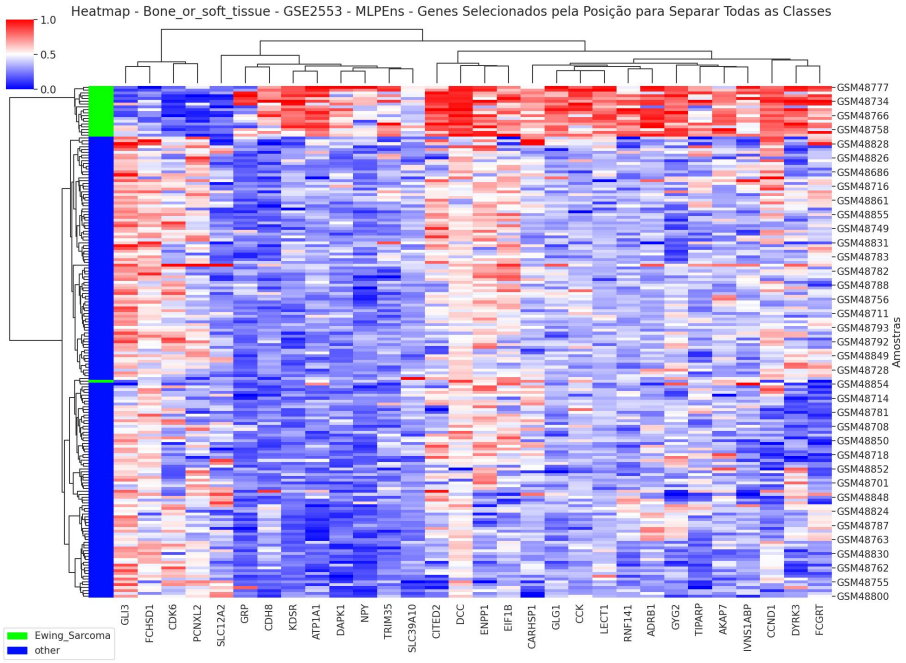

Figura 70 – MLPEns pela Posição - Heatmap.

Tabela 43 – MLPEns pelo Coeficiente - Melhores Genes.

	Média Pos	Std Pos	Max Pos	Min Pos	Q1 Pos	Mediana Pos	Q3 Pos	Média Coef	Std Coef	Max Coef	Min Coef	Q1 Coef	Mediana Coef	Q3 Coef
KDSR	**4920.3**	5.6	4924	4903	4920.8	4922.0	4923.0	**0.741**	0.097	0.950	0.594	0.673	0.723	0.800
DYRK3	**4918.0**	6.7	4924	4900	4915.8	4919.5	4924.0	**0.732**	0.145	1.141	0.591	0.637	0.683	0.777
GLG1	**4916.4**	7.3	4924	4893	4912.8	4916.0	4922.0	**0.697**	0.077	0.833	0.576	0.637	0.694	0.744
ATP1A1	**4914.4**	9.4	4923	4892	4911.0	4917.5	4922.0	**0.687**	0.073	0.828	0.576	0.636	0.671	0.746
CCK	**4891.2**	45.3	4923	4767	4884.0	4915.5	4920.5	**0.666**	0.121	0.966	0.498	0.600	0.644	0.704
RNF141	**4892.6**	48.0	4923	4733	4901.5	4908.5	4916.5	**0.650**	0.116	1.013	0.524	0.581	0.615	0.652
FCGRT	**4895.0**	40.7	4924	4744	4894.2	4909.5	4918.2	**0.640**	0.066	0.770	0.531	0.586	0.644	0.685
DAPK1	**4892.2**	44.5	4924	4737	4893.0	4910.5	4916.5	**0.638**	0.066	0.820	0.525	0.590	0.626	0.660
ENPP1	**4872.1**	74.9	4924	4564	4864.8	4892.5	4904.0	**0.616**	0.081	0.774	0.501	0.551	0.589	0.690
DCC	**4832.8**	173.7	4922	4228	4879.8	4900.0	4915.0	**0.614**	0.084	0.841	0.443	0.560	0.612	0.662
ADRB1	**4841.4**	195.9	4921	4002	4856.0	4900.5	4913.8	**0.611**	0.079	0.833	0.486	0.578	0.600	0.621
NPY	**4807.0**	224.3	4920	3933	4855.8	4895.5	4913.8	**0.608**	0.087	0.748	0.430	0.551	0.591	0.665
AKAP7	**4813.2**	245.5	4921	3769	4805.8	4889.5	4909.8	**0.605**	0.089	0.821	0.462	0.546	0.585	0.627
TRIM35	**4848.6**	79.8	4921	4592	4806.2	4878.5	4911.2	**0.602**	0.085	0.830	0.509	0.529	0.600	0.627
TIPARP	**4845.2**	126.8	4923	4357	4849.8	4876.5	4909.8	**0.601**	0.078	0.854	0.481	0.554	0.592	0.631
GYG2	**4776.5**	252.6	4923	4045	4799.8	4875.0	4910.2	**0.600**	0.094	0.848	0.447	0.547	0.588	0.661
GLI3	**4828.2**	133.7	4922	4316	4785.0	4885.0	4904.2	**0.599**	0.100	0.878	0.481	0.528	0.570	0.637
LECT1	**4857.2**	101.7	4918	4468	4864.2	4891.0	4905.5	**0.599**	0.058	0.741	0.516	0.552	0.601	0.629
CITED2	**4815.6**	259.1	4921	3700	4839.8	4885.5	4903.8	**0.598**	0.072	0.761	0.442	0.553	0.587	0.647
CCND1	**4823.5**	168.2	4922	4166	4834.0	4876.5	4911.5	**0.598**	0.070	0.727	0.480	0.563	0.598	0.648
SLC12A2	**4830.8**	135.3	4921	4318	4841.5	4877.0	4909.5	**0.598**	0.100	0.920	0.460	0.538	0.584	0.636
CDK6	**4802.6**	301.6	4920	3504	4852.2	4885.5	4906.0	**0.598**	0.087	0.780	0.413	0.547	0.576	0.645
GRP	**4837.0**	147.6	4924	4231	4825.2	4881.5	4898.2	**0.589**	0.066	0.772	0.481	0.549	0.580	0.614
IVNS1ABP	**4777.6**	297.6	4917	3809	4826.8	4880.0	4906.0	**0.589**	0.077	0.765	0.433	0.555	0.587	0.640
FCHSD1	**4826.2**	106.2	4923	4559	4765.0	4877.5	4904.5	**0.588**	0.082	0.811	0.490	0.519	0.578	0.624
PCNXL2	**4758.0**	355.2	4918	3277	4799.8	4859.5	4909.2	**0.588**	0.094	0.801	0.416	0.537	0.565	0.640
CARHSP1	**4859.8**	45.0	4918	4755	4827.0	4867.5	4898.8	**0.585**	0.051	0.729	0.515	0.549	0.576	0.609
SLC1A1	**4685.4**	448.7	4922	2992	4733.2	4855.5	4910.5	**0.581**	0.107	0.810	0.417	0.506	0.570	0.630
CDH8	**4820.0**	95.2	4919	4586	4793.5	4854.0	4889.5	**0.574**	0.071	0.759	0.485	0.521	0.552	0.592
SLC39A10	**4782.9**	152.0	4919	4280	4690.2	4825.5	4897.0	**0.573**	0.084	0.839	0.466	0.510	0.555	0.616

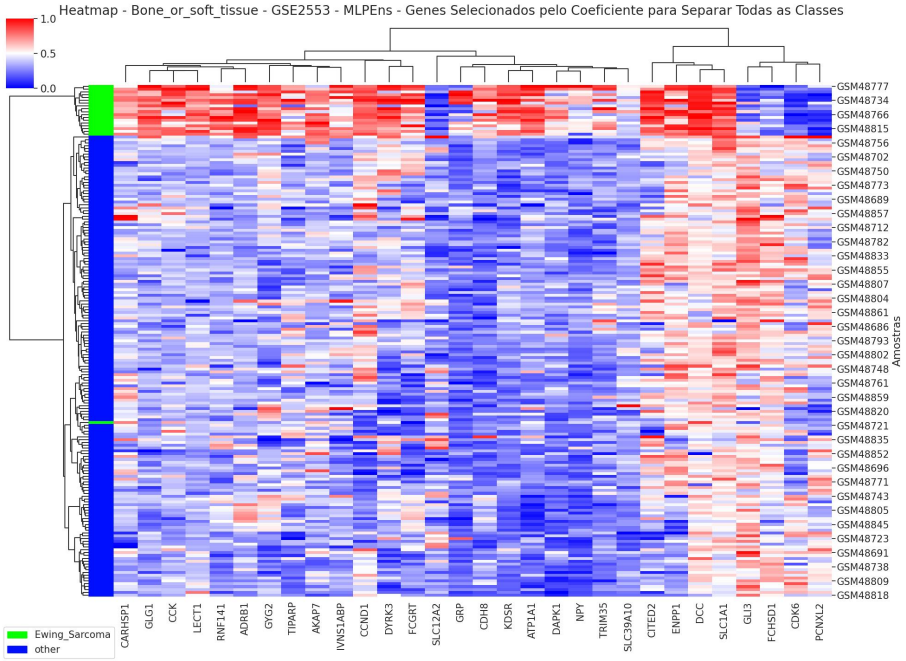

Figura 71 – MLPEns pelo Coeficiente - Melhores Genes - Heatmap.

Como podemos perceber, nesse dataset, os heatmaps obtidos com poucos genes selecionados pela MLPEns, assim como os obtidos pelos genes selecionados pelo SVM, conseguem separar bem as classes. Os genes KDSR, DYRK3, GLG1, ATP1A1 e FCGRT aparecem no topo como mais importantes para ambas as metodologias

estudadas.

8.2.2.3 Contextualização e Relevância Biológica

O **sarcoma de Ewing** é caracterizado pela presença da fusão entre o gene **EWSR1** e um gene da família ETS, na maioria das vezes **FLI1**.

O **diagnóstico** do sarcoma de Ewing é um grande **desafio**, pois os tumores do sarcoma de Ewing são, em grande parte, compostos de células indiferenciadas, exibindo um fenótipo de pequenas células redondas. Este **fenótipo é compartilhado** por muitas outras entidades tumorais, como rabdomiossarcoma e neuroblastoma.

Recentemente, **vários subtipos de sarcomas do tipo Ewing** foram identificados. Esses tumores são caracterizados por oncogenes de fusão distintos e diferentes assinaturas transcriptômicas.

Atualmente **não há um marcador robusto** que possa ser utilizado nas análises citogenéticas de rotina. Além disso, **erros de diagnóstico** podem surgir, uma vez que técnicas mais sofisticadas de diagnóstico citogenético e de sequenciamento molecular não são disponíveis na grande maioria dos centros ou são muito caras (particularmente para países em desenvolvimento).

É importante salientar que o marcador CD99, muito sensível para Sarcoma de Ewing, apresenta uma baixa especificidade, uma vez que ele apresenta elevada expressão em tumores com semelhanças morfológicas e em certos tipos de linfomas e outros. Assim, são utilizados outros marcadores juntamente com CD99 para um diagnóstico mais preciso (ZÖLLNER et al., 2021).

8.2.2.4 Análise dos Genes Selecionados

Nos resultados que obtivemos, os genes **KDSR, DYRK3, GLG1, ATP1A1 e FCGRT** apareceram entre os sete primeiros genes selecionados como mais importantes pelas quatro abordagens de ranqueamento que empregamos (como apresentado na Tabela 44 a seguir), mostrando que esses cinco genes tem um grande potencial no diagnóstico do Sarcoma de Ewing.

Tabela 44 – Sarcoma de Ewing - Primeiros 7 genes selecionados pelos diferentes critérios com SVM e MLPEns.

Ranking	MLPEns Posição	MLPEns Coeficiente	SVM Posição	SVM Coeficiente
1°	KDSR	KDSR	ATP1A1	ATP1A1
2°	DYRK3	DYRK3	DYRK3	KDSR
3°	GLG1	GLG1	KDSR	DYRK3
4°	ATP1A1	ATP1A1	GLG1	FCGRT
5°	FCGRT	CCK	FCGRT	AKAP7
6°	RNF141	RNF141	LOC399959	GLG1
7°	DAPK1	FCGRT	AKAP7	LOC399959

Buscando por conhecimento disponível na literatura, verificamos que existe base para defender que esses genes realmente se mostram como importantes.

GLG1 tem sido proposto como marcador de imuno-histoquímica em combinação com outros marcadores CAV1, NKX2.2 e BCL11B, os quais são utilizados para auxiliar no diagnóstico de Sarcoma de Ewing, especialmente nos casos em que a expressão de CD99 é negativa (BALDAUF et al., 2018; MACHADO et al., 2018; LLOMBART-BOSCH et al., 2009).

Recentemente foi demonstrado que **ATP1A1** e **GLG1** são marcadores com alta especificidade para sarcoma de Ewing e podem oferecer um diagnóstico rápido e eficiente através de imuno-histoquímica. Esses marcadores juntamente com BCL11B podem **diminuir o número de pacientes com erros de diagnóstico** (BALDAUF et al., 2018).

Por sua vez, **KDSR e FCGRT** são genes cujas expressões são **reguladas por EWSR1-FLI1** (CIDRE-ARANAZ; ALONSO, 2015).

8.2.2.5 SVM e MLPEns - tSNEs

A seguir, apresentamos tSNEs usando apenas os genes selecionados como mais importantes para separação das classes. Nas Figuras 72 e 73 são mostrados os tSNEs usando os 5 genes mais importantes selecionados pelo SVM e pela MLPEns, respectivamente.

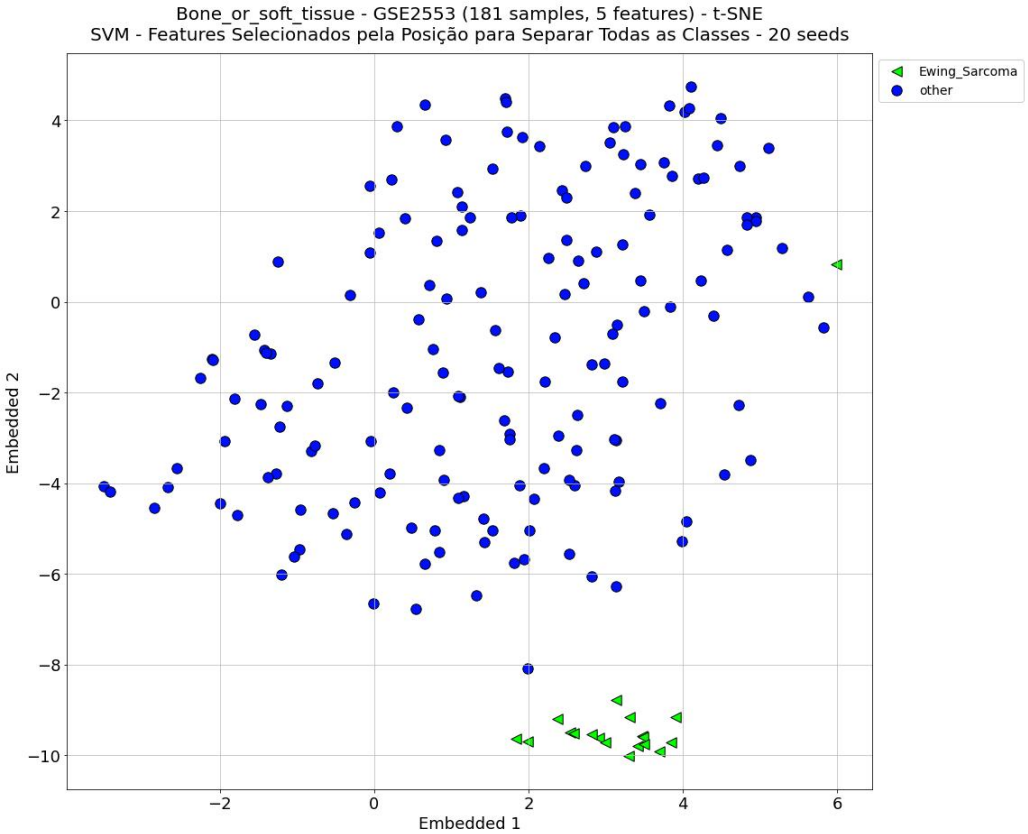

Figura 72 – SVM - 5 genes - tSNE

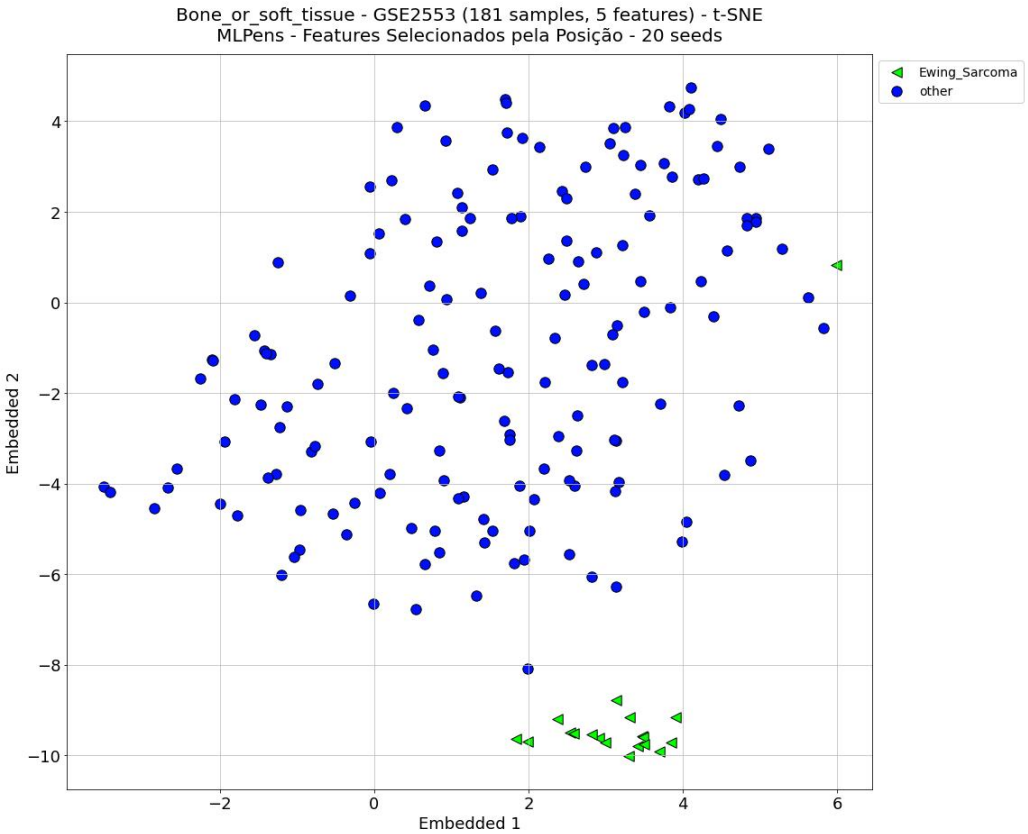

Figura 73 – MLPEns - 5 genes - tSNE

Nas Figuras 74 e 75 são apresentados os tSNEs usando os 100 genes mais importantes selecionados pelo SVM e pela MLPEns, respectivamente.

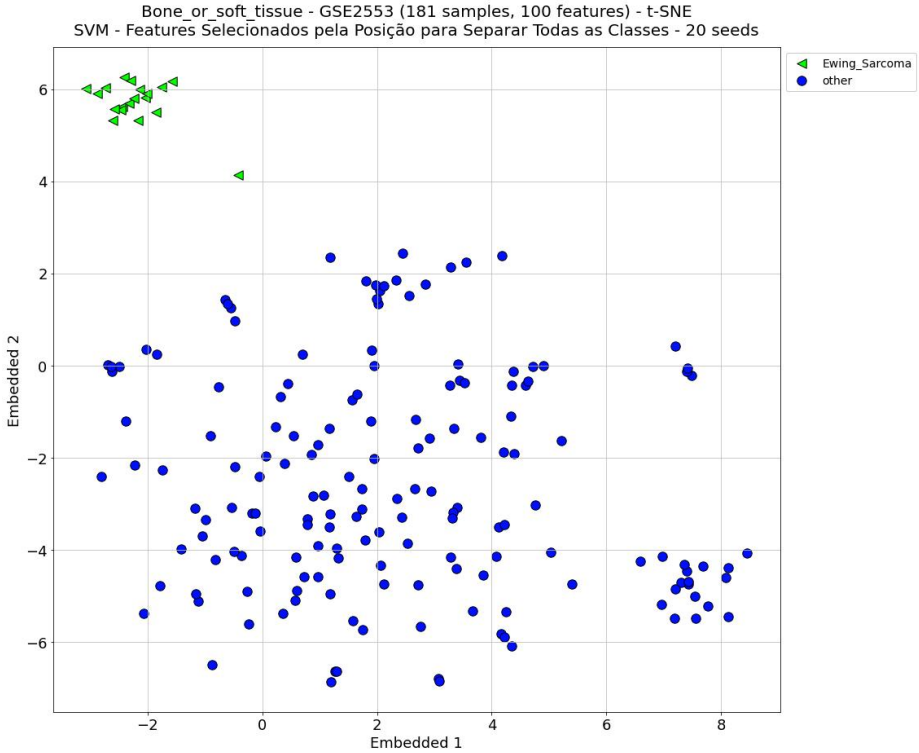

Figura 74 – SVM - 100 genes - tSNE

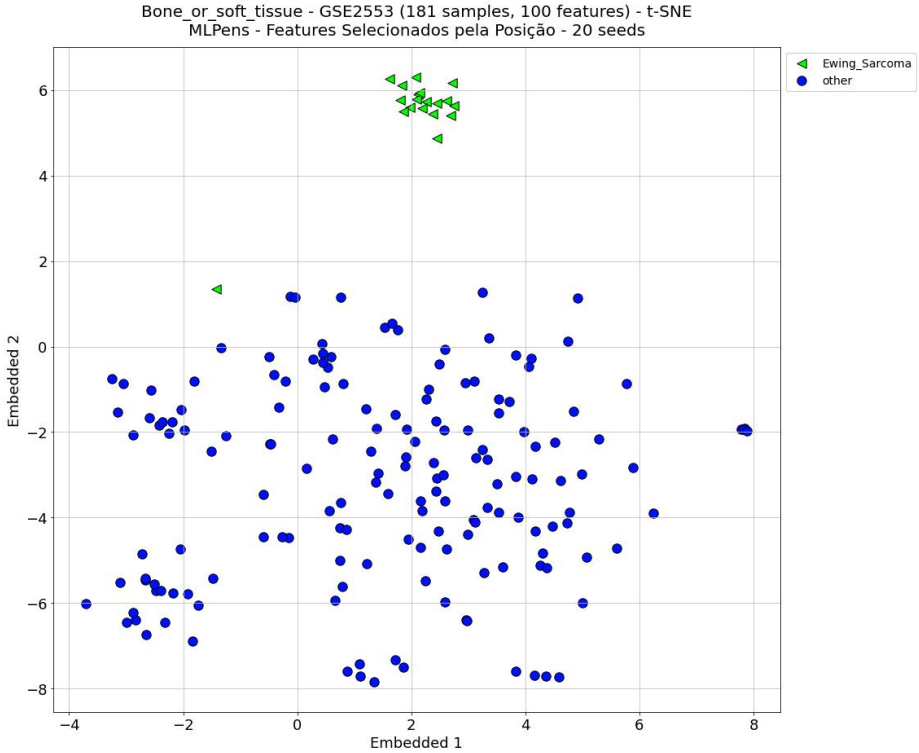

Figura 75 – MLPEns - 100 genes - tSNE

8.3 BorutaShap

O Método **BorutaShap** (KEANY, 2020) foi investigado para seleção de genes relevantes em conjunto com três métodos de classificação baseados em árvores de decisão: Árvores Aleatórias (**Random Forest**); **Extra Trees**; e **XGBoost**. Para ambos métodos empregou-se os hiperparâmetros *default*.

Nesse experimento, foi utilizado apenas o dataset piloto de meduloblastoma GSE85217 com $n = 763$. O dataset foi separado com 70% das amostras para treino e 30% para teste. A seleção dos genes relevantes foi feita analisando apenas o conjunto de treinamento.

Como entrada, utilizou-se a expressão gênica de todos os genes com relações estabelecidas no BioGRID (um total de 15.445 genes), uma vez que se desejava selecionar genes que apresentavam conhecimento biológico disponível na literatura. Então, esses dados foram escalados com *mini-max feature scaling*. Foi executada apenas uma semente aleatória por modelo de classificador.

A Figura 76 mostra um diagrama de Venn dos genes selecionados pelos métodos investigados. Como pode ser percebido, o **conjunto de genes selecionados difere de método para método**. A interseção dos conjuntos de ambos métodos forneceu um grupo de 58 genes.

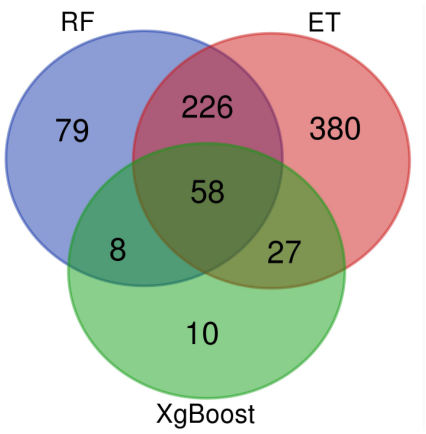

Figura 76 – Diagrama de Venn dos genes selecionados pelos métodos Extra Trees (ET), Random Forest (RF) e XGBoost.

Após obter a seleção de genes relevantes, executamos o treinamos de classificadores dos três métodos usados (*Random Forest*, *Extra Trees* e XGBoost) e avaliamos no conjunto de teste. Como entrada dos classificadores, nós usamos os seguintes seis conjuntos de *features* distintos:

- 371 genes selecionados pela *Random Forest* (RF);
- 691 genes selecionados pela *Extra Trees* (ET);
- 103 genes selecionados pelo XGBoost;
- 58 genes selecionados por ambos métodos (interseção);

- 788 genes selecionados por qualquer um dos métodos (união);
- todos 15.445 genes analisados.

Os resultados são apresentados na Tabela 45 abaixo. Foi utilizado 10 execuções para cada um dos três métodos de classificação (10 sementes aleatórias) e para cada um dos seis conjuntos de *features* mencionado anteriormente. Cada linha representa um dos conjuntos de *feaures* distintos e as colunas especificam o método usado para a classificação.

Tabela 45 – Resultado de classificadores usando, como *features*, os genes selecionados pelo BorutaShap.

Genes selecionados por:	Nr de Genes	RF Acc	ET Acc	XgBoost Acc	Média Acc	Std Acc
RF	371	99.09	98.63	97.26	98.33	0.78
ET	691	99.09	99.09	97.72	98.63	0.65
XgBoost	103	99.54	99.54	98.17	99.08	0.65
Interseção	58	99.09	99.09	97.72	98.63	0.65
Todos genes selecionados	788	99.09	99.54	98.17	98.93	0.57
Todos genes	15445	99.09	98.63	98.63	98.78	0.22

Como pode ser percebido, a opção de maior acurácia foi a seleção de genes realizadas pelo XGBoost e a classificação pela Extra Trees. O conjunto de apenas 58 genes, obtidos como interseção, alcançou média de acurácia de 98.63%, similar a do conjunto de todos os 15.445 genes, que obteve 98.78% de média de acurácia.

As Figuras 77 e 78 mostram o t-SNE e o heatmap usando apenas os 58 genes obtidos na interseção dos genes selecionados pelos métodos.

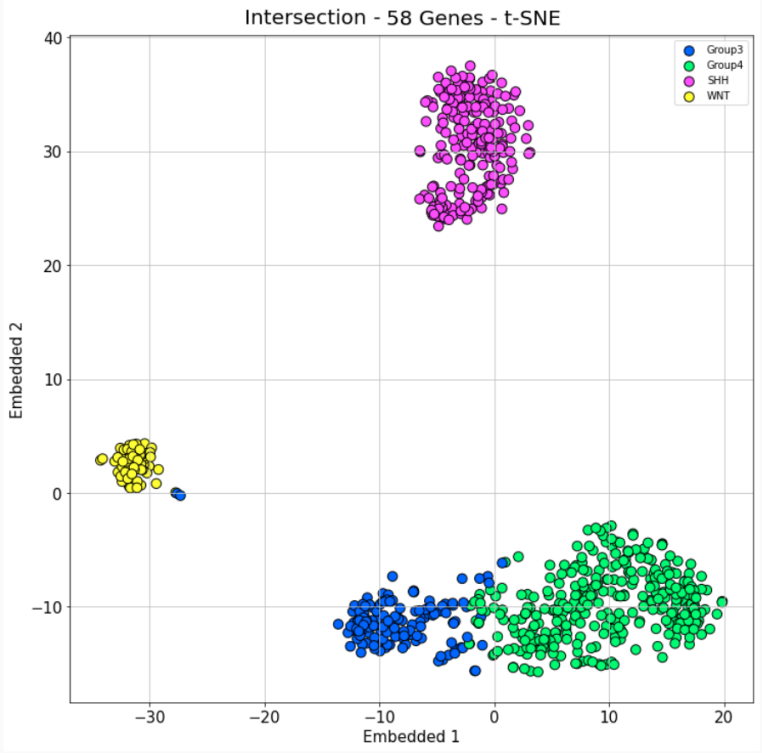

Figura 77 – Meduloblastoma - tSNE com os 58 genes selecionados pelo BorutaShap.

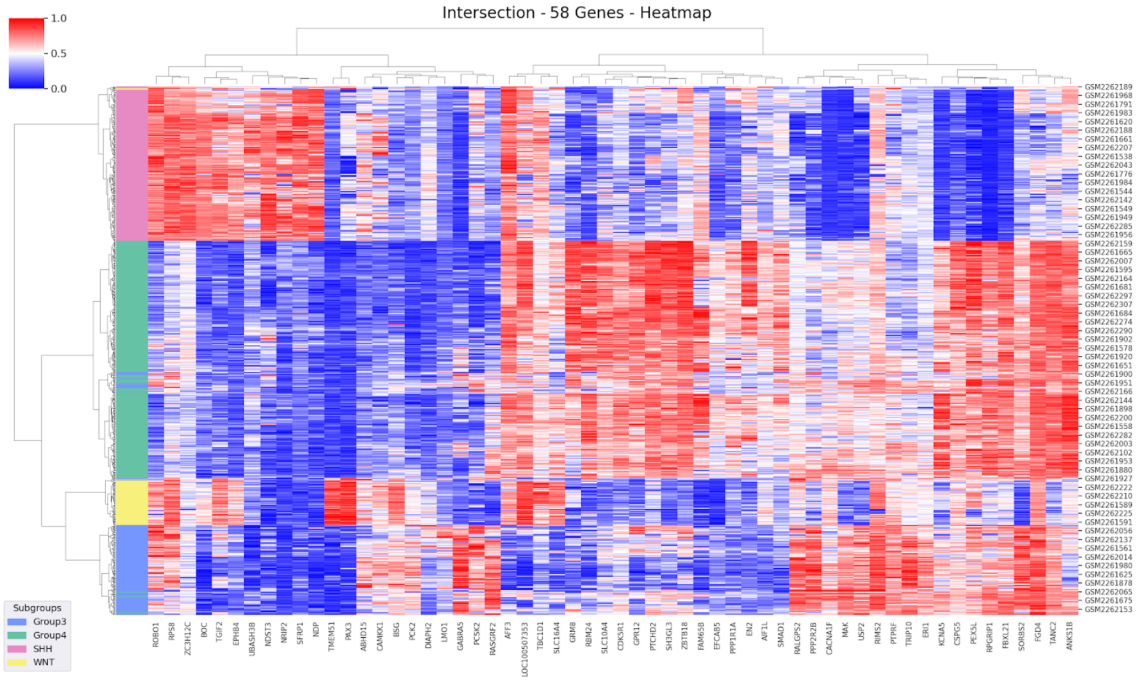

Figura 78 – Meduloblastoma - Heatmap com os 58 genes selecionados pelo BorutaShap.

Consideramos esses **resultados positivos**. Entretanto, temos que observar uma desvantagem: o **BorutaShap exige alto tempo de processamento**. Pois, para tomar decisões sobre todos genes, precisa executar o treinamento de vários modelos de um mesmo classificador.

8.4 Comparação entre as Abordagens para Seleção de Genes Relevantes

Ao comparar as formas de seleção de genes estudadas, concluímos como a melhor pelo **SVM critério de posição**. Pois, além de proporcionar bons resultados com essa forma de seleção, pela inspeção visual dos tSNEs e *Heatmaps* (Agrupamentos Hierárquicos); o classificador SVM apresentou um dos melhores resultados em acurácia de classificação (como apresentado na Seção 7.12); e possui baixo tempo de processamento para treino, em relação ao treinamento de modelos de redes neurais ou a execução da seleção pelo BorutaShap.

Além disso, o SVM é implementado de forma que um classificador, em problemas multi-classes de $nr\_classes$, seja composto de $nr\_classes$ **modelos OvR** (One versus Rest - uma classe versus demais). Essa característica favorece na interpretabilidade dos resultados. Pois, podemos obter, também, uma seleção de **genes biomarcadores para cada classe** estudada (exemplo, subtipo tumoral).

É importante destacar, também, que a metodologia adotada para seleção de genes com o SVM pode ser facilmente **estendida** a métodos de regressão ou **a demais métodos** de Aprendizagem de Máquina que possam atribuir valores de importância a cada *feature*, como o *Ridge Classifier* e o *Logistic Regression*, que obtiveram altas médias de acurácia nos experimentos de classificação (Seção 7.12).

Concluímos, também, a importância de, independentemente da metodologia adoada para seleção de genes, **analisar mais do que um modelo** de Aprendizagem de Máquina; devido à alta redundância de informações nos *features* e **correlações** existentes em dados de expressão gênica, especialmente quando o intuito da seleção de genes seja compreender os mecanismos subadjacentes a uma doença.

9 CONCLUSÃO

Uma primeira conclusão desse trabalho é a grande importância e potenciais contribuições que análises de dados de expressão gênica podem trazer, indo desde possibilitar o diagnóstico correto e rápido; como a definição de potenciais alvos de fármacos; compreensão de doenças; e medicina personalizada.

Entretanto, percebe-se desafios importantes. Um primeiro é o fato de que esses tipos de dados estão sujeitos à maldição de dimensionalidade (número de amostras muito menor ao número de *features*). Além disso, apresentam grande redundância de informação e correlações entre *features*. Um segundo é a dificuldade de analisar *datasets* disponíveis em conjunto, fazendo com que cada *dataset* seja considerado em separado.

Destacamos, também, a importância de trabalhos que façam análises comparativas de métodos. Como pôde ser visto, no Capítulo 3 dessa tese, ao termos as *Deep Neural Networks* (DNNs) despontando no campo de Aprendizagem de Máquina, muitos trabalhos começaram a surgir, na literatura, investigando e, principalmente, prospectando as possíveis contribuições das DNNs na área de análise de dados de expressão gênica. Entretanto, esses trabalhos não realizavam comparações com abordagens mais tradicionais de Aprendizagem de Máquina que, pelos nossos experimentos, demonstraram ser inclusive melhores para a maioria dos datasets analisados.

Além disso, ressaltamos a importância de, ao se selecionar genes relevantes, analisar uma combinação de modelos, principalmente, se o objetivo for compreensão de doenças ou montagem de redes gênicas, pois modelos treinados com sementes aleatórias distintas e diferentes métodos obtém soluções de assinaturas gênicas distintas, pela alta correlação entre os *features* (genes).

A abordagem proposta de explorar o *dropout* para tratar um modelo de rede neural como um *ensemble* de modelos (e a arquitetura MLPEns) nos pareceu interessante ao percebê-la como um caminho diferente da maioria dos trabalhos em redes neurais existentes. Essa metodologia conseguiu fornecer resultados positivos em análise de expressão gênica, sendo estatisticamente comparável aos melhores métodos de Aprendizagem de Máquina que encontramos para esse tipo de dado. Assim a perce-

bemos como promissora para mais investigações em trabalhos futuros.

Entretanto, mesmo com os resultados positivos da MLPEns, após exaustivos experimentos com abordagens de Redes Neurais Profundas, ainda percebemos os métodos de Aprendizagem Rasa como os mais adequados para classificações de amostras de expressão gênica e apresentam, também, como vantagens um menor tempo de processamento e facilidades na interpretabilidade dos modelos.

Segundo os nossos experimentos, os melhores métodos de Aprendizagem Rasa para classificação foram o Ridge Classifier, a Regressão Logística e o SVM, especialmente, o com kernel linear L2. Entre os métodos de Redes Neurais, os melhores foram a MLPEns e a MLP.

Obtemos como melhor forma de entrada para os classificadores tratar os dados como expressão de genes (mapeando cada gene à expressão da sonda de maior variância associada a esse gene) e, antes da utilização dos classificadores, empregar o filtro de Kruskal Wallis. Essa filtragem, também, facilita a realização de vários experimentos, pois reduz consideravelmente o tempo de processamento.

Dentre as abordagens de seleção de genes relevantes, concluímos como melhores as abordagens *embedded* com o emprego de um classificadores de Aprendizagem Rasa cuja associação de um valor de importância por gene seja direta, preferencialmente, algum dos métodos que mostraram obter maior acurácia nos experimentos que realizamos, como o Ridge Classifier, a Regressão Logística ou o SVM com kernel Linear L2. O tempo de processamento reduzido das abordagens *embedded*, em relação às abordagens *wrapper* de *feature selection* (como o BorutaShap), favorecem a execução de vários modelos, para se obter um ranqueamento de genes mais robusto, que considere vários modelos.

Nós verificamos, pela grande variabilidade vista nos valores de importância dados aos genes pelos classificadores, que a resposta de um modelo pode ser muito diferente de outro executado apenas alterando a semente aleatória, pois cada gene apresenta expressão correlacionada com outros, que podem ser tomados de forma equivalente pelos classificadores.

Devemos, também, destacar a decepção nos resultados de Redes Neurais Profundas, que mostraram fornecer piores resultados que os métodos de Aprendizagem Rasa. Conclui-se, portanto, que existe uma lacuna nessa área de pesquisa: a comunidade científica ainda precisa trabalhar no sentido de estabelecer soluções mais adequadas para a análise dados tabulares sujeitos à maldição de dimensionalidade, como os dados de expressão gênica.

Nesse contexto, percebemos as abordagens generativas e de aprendizagem não supervisionada como promissoras. Além disso, abordagens que tratem de favorecer generalização dos modelos, como mecanismos de regularização e a nossa abordagem de tratar um modelo de rede neural como *ensemble* de modelos, também, podem

trazer contribuições.

Entretanto, essas são linhas de pesquisas mais computacionais. Para a análise de expressão genica, com o arcabouço computacional que se tem hoje, os métodos de Aprendizagem Rasa ainda parecem ser as melhores soluções.

Percebemos que, dependendo do dataset de expressão gênica utilizado, um método fornecerá resultado melhor que outro. Assim, notamos, também, a importância de que trabalhos que visem estabelecer abordagens computacionais, para tratar dados de expressão gênica, devam analisar suas abordagens em vários dataset.

9.1 Contribuições do Trabalho

A seguir, listamos as principais contribuições do presente trabalho.

- Realizou-se uma pesquisa de trabalhos relacionados à área de análise de expressão gênica e utilização de Aprendizagem Profunda - *Deep Learning* (DL) em problemas de bioinformática, bem como, de técnicas computacionais propostas no campo de DL.
- Realizaram-se experimentos em diferentes datasets e tecidos com diferentes métodos de Aprendizagem de Máquina, tanto métodos de Aprendizagem Profunda, quanto métodos de Aprendizagem Rasa.
- Realizou-se uma comparação dos métodos estudados e especificação das melhores abordagens para classificação de amostras de expressão gênica.
- Analisaram-se três abordagens para seleção de genes relevantes na classificação, um exemplo de *wrapper feature selection* e duas abordagens *embedded* com classificador SVM e com a redes neurais MLPEs. Para as abordagens *embedded* experimentaram-se dois critérios para o estabelecimento de um ranqueamento de genes para a combinação dos resultados de vários modelos.
- Compararam-se as abordagens de seleção de genes estudadas, especificando qual se mostrou melhor.
- Propôs-se uma metodologia que visa contribuir para a generalidade de modelos de redes neurais que apresentou bom desempenho nos experimentos realizados - a metodologia para explorar o *dropout* e a MLPEns.
- Propôs-se uma abordagem para organizar os *features* de entrada de uma rede neural CNN que apresentou resultados superiores à disposição dos *features* em ordem aleatória - o Transcriptograma.
- Mostramos que resultados em seleção de genes relevantes variam de modelo para modelo, dependendo da semente aleatória e do método.
- Mostrou-se, também, que o melhor método de classificação, melhor conjunto de hiperparâmetros e melhor forma de entrada dos dados variam com o dataset /

problema especificado, concluindo que as melhores abordagens para a análise de expressão gênica são aquelas que fornecem os melhores resultados quando se considera um conjunto de dadasets.

9.2 Trabalhos Futuros

Percebemos que as possibilidades de linhas de pesquisa para trabalhos possíveis na análise de dados de bioinformática são inúmeras, mas destacamos alguns pontos como trabalhos futuros relevantes:

- construir redes de iteração gênica de doenças, para maior interpretabilidade dos genes selecionados como relevantes pelas abordagens estabelecidas neste trabalho;
- realizar experimentos da abordagem de rede neural proposta - que trata um modelo como um ensemble, explorando o *dropout* - em outras arquiteturas e datasets;
- analisar os gradientes de todas camadas da rede neural *MLPEns* para maior interpretabilidade dos modelos aprendidos;
- empregar a abordagem de rede neural proposta em um *framework Generative Adversarial Network* (GAN);
- analisar se o ordenamento 2D do Transcriptograma pode fornecer resultados superiores ao 1D;
- analisar a abordagem especificada de ranqueamento de genes importantes em outros datasets e em conjunto de outros métodos de classificação, especialmente, com o *Ridge Classifier* e com a Regressão Logística que apresentaram alta acurácia nas tarefas de classificação;
- investigar métodos de Aprendizagem de Máquina para predição de sobrevida e seleção de genes relevantes nessa tarefa;
- realizar experimentos em bancada, para validação dos genes selecionados como relevantes pela abordagem estabelecida de ranqueamento de genes;
- investigar as abordagens estabelecidas em outros tipos de dados como dados de *single cell* e dados de metilação;
- experimentar as abordagens estabelecidas em análises integradas de diferentes tipos de dados e plataformas.

9.3 Resumo do Texto

Primeiramente, o Capítulo 1 introduziu a contextualização do escopo do presente trabalho, destacando os desafios envolvidos em análises de dados de expressão gênica e ressaltando sua importância em estudos relacionados a cânceres. Bem como, foram estabelecidos os objetivos do presente trabalho e a organização do texto.

O Capítulo 2 apresentou o referencial teórico, tratando alguns conceitos-chave relacionados a Redes Neurais Profundas - *Deep Neural Networks (DNNs)*; explicou as *Convolutional Neural Networks (CNNs)* e o *framework Generative Adversarial Networks (GAN)*. Após, foi apresentado a técnica usada para mapeamento de dados organizados como um conjunto de variáveis relacionadas para um *multiarray*, visando favorecer a análise desse tipo de dados por redes neurais CNNs.

Seguindo, o Capítulo 3 apresentou os trabalhos relacionados, destacando pesquisas na área *Deep Learning (DL)* em problemas tratados em Bioinformática e, especialmente, em análises envolvendo dados de expressão gênica e câncer.

Então, o Capítulo 4 apresentou um esquema da metodologia adotada para investigação dos métodos de Aprendizagem de Máquina; descreveu os datasets empregados; o procedimento para a escolha dos hiperparâmetros; e a comparação dos métodos.

No Capítulo 5, propôs-se uma abordagem que utiliza o mecanismo de *dropout* para explorar um modelo de Rede Neural como um *ensemble* de modelos, visando contribuir na capacidade de generalização dos modelos aprendidos.

O Capítulo 6, então, abordou as metodologias estabelecidas para a seleção de genes relevantes nas tarefas de classificação.

Os Capítulos 7 e 8 apresentaram os resultados e discussões dos experimentos realizados. O Capítulo 7 apresentou os resultados obtidos com as abordagens de classificação; enquanto que, o Capítulo 8 apresentou os resultados das metodologias de seleção de genes relevantes.

Finalizando, este Capítulo 9 resumiu os principais aspectos do trabalho, discutindo as contribuições e percepções alcançadas com o trabalho, bem como, relatando possibilidades de trabalhos futuros.

REFERÊNCIAS

ALOM, M. Z. et al. The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches. **arXiv preprint arXiv:1803.01164**, [S.l.], 2018.

AMIDI, A. et al. EnzyNet: enzyme classification using 3D convolutional neural networks on spatial representation. **PeerJ**, [S.l.], v.6, p.e4750, may 2018.

ANGERMUELLER, C.; PÄRNAMAA, T.; PARTS, L.; STEGLE, O. Deep learning for computational biology. **Molecular systems biology**, [S.l.], v.12, n.7, p.878, 2016.

ARRAYEXPRESS. **ArrayExpress – functional genomics data**. Disponível em: <https://www.ebi.ac.uk/arrayexpress/>.

BACCIU, D. et al. Bioinformatics and Medicine in the Era of Deep Learning. **arXiv preprint arXiv:1802.09791**, [S.l.], 2018.

BACH, S. et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. **PloS one**, [S.l.], v.10, n.7, p.e0130140, 2015.

BAEK, J.; LEE, B.; KWON, S.; YOON, S. LncRNAnet: long non-coding RNA identification using deep learning. **Bioinformatics**, [S.l.], may 2018.

BALDAUF, M. C. et al. Robust diagnosis of Ewing sarcoma by immunohistochemical detection of super-enhancer-driven EWSR1-ETS targets. **Oncotarget**, [S.l.], v.9, n.2, p.1587, 2018.

BALDI, P. Deep learning in biomedical data science. **Annual Review of Biomedical Data Science**, [S.l.], v.1, p.181–205, 2018.

BENGIO, Y.; COURVILLE, A.; VINCENT, P. Representation learning: a review and new perspectives. arXiv. org. **arXiv preprint arXiv:1206.5538**, [S.l.], 2012.

BHAT, R. R.; VISWANATH, V.; LI, X. DeepCancer: Detecting Cancer through Gene Expressions via Deep Generative Learning. **arXiv preprint arXiv:1612.03211**, [S.l.], 2016.

BIOGRID. **BioGRID - Biological General Repository for Interaction Datasets**. Disponível em: <https://thebiogrid.org/>.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.]: springer, 2006.

BRAY, F. et al. Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. **CA: a cancer journal for clinicians**, [S.l.], v.68, n.6, p.394–424, 2018.

BRETSCHNEIDER, H. et al. COSSMO: predicting competitive alternative splice site selection using deep learning. **Bioinformatics**, [S.l.], v.34, n.13, p.i429–i437, jun 2018.

BRUCE, A.; BRUCE, P. **Estatística Prática para Cientistas de Dados**. [S.l.]: Alta Books, 2019.

CAMACHO, D. M. et al. Next-Generation Machine Learning for Biological Networks. **Cell**, [S.l.], 2018.

CANCER, I. A. for Research on. The GLOBOCAN project. **https://gco.iarc.fr/**, [S.l.], 2019.

CANCER, I. I. A. for Research on. **Latest global cancer data**: Cancer burden rises to 18.1 million new cases and 9.6 million cancer deaths in 2018.

CAO, Z. et al. The lncLocator: a subcellular localization predictor for long non-coding RNAs based on a stacked ensemble classifier. **Bioinformatics**, [S.l.], v.1, p.10, 2018.

CELESTI, F.; CELESTI, A.; WAN, J.; VILLARI, M. Why Deep Learning Is Changing the Way to Approach NGS Data Processing: a Review. **IEEE Reviews in Biomedical Engineering**, [S.l.], 2018.

CHANG, Y. et al. Cancer Drug Response Profile scan (CDRscan): A Deep Learning Model That Predicts Drug Effectiveness from Cancer Genomic Signature. **Scientific Reports**, [S.l.], v.8, n.1, jun 2018.

CHAUDHARY, K.; POIRION, O. B.; LU, L.; GARMIRE, L. Deep Learning based multi-omics integration robustly predicts survival in liver cancer. **bioRxiv**, [S.l.], p.114892, 2017.

CHEN, H. et al. The rise of deep learning in drug discovery. **Drug discovery today**, [S.l.], 2018.

CHENG, B.; LIU, L. yun; QI, Z. hui; YANG, H. guang. Prediction of Continuous B-cell Epitopes Using Long Short Term Memory Networks. In: INTERNATIONAL CONFERENCE ON BIOINFORMATICS AND COMPUTATIONAL BIOLOGY - ICBCB 2018, 2018., 2018. **Proceedings...** ACM Press, 2018.

CHING, T. et al. Opportunities and obstacles for deep learning in biology and medicine. **Journal of The Royal Society Interface**, [S.l.], v.15, n.141, p.20170387, 2018.

CHIU, Y.-C. et al. **Predicting drug response of tumors from integrated genomic profiles by deep neural networks**. Disponível em: <https://arxiv.org/abs/1805.07702>.

CHUAI, G. et al. DeepCRISPR: optimized CRISPR guide RNA design by deep learning. **Genome Biology**, [S.l.], v.19, n.1, jun 2018.

CIDRE-ARANAZ, F.; ALONSO, J. EWS/FLI1 target genes and therapeutic opportunities in Ewing sarcoma. **Frontiers in oncology**, [S.l.], v.5, p.162, 2015.

CUI, P. et al. Identification of human circadian genes based on time course gene expression profiles by using a deep learning method. **Biochimica et Biophysica Acta (BBA) - Molecular Basis of Disease**, [S.l.], v.1864, n.6, p.2274–2283, jun 2018.

DANAEE, P.; GHAEINI, R.; HENDRIX, D. A. A DEEP LEARNING APPROACH FOR CANCER DETECTION AND RELEVANT GENE IDENTIFICATION. In: PACIFIC SYMPOSIUM ON BIOCOMPUTING. PACIFIC SYMPOSIUM ON BIOCOMPUTING, 2016. **Anais...** [S.l.: s.n.], 2016. v.22, p.219.

DINCER, A. B.; CELIK, S.; HIRANUMA, N.; LEE, S.-I. DeepProfile: Deep learning of cancer molecular profiles for precision medicine. **bioRxiv**, [S.l.], mar 2018.

DIZAJI, K. G.; WANG, X.; HUANG, H. Semi-Supervised Generative Adversarial Network for Gene Expression Inference. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY & DATA MINING, 24., 2018. **Proceedings...** ACM Press, 2018.

DOSOVITSKIY, A. et al. An image is worth 16x16 words: Transformers for image recognition at scale. **arXiv preprint arXiv:2010.11929**, [S.l.], 2020.

DU, Y. et al. Predicting Drug-target Interaction via Wide and Deep Learning. In: INTERNATIONAL CONFERENCE ON BIOINFORMATICS AND COMPUTATIONAL BIOLOGY - ICBCB 2018, 2018., 2018. **Proceedings...** ACM Press, 2018.

DUTIL, F. et al. **Towards Gene Expression Convolutions using Gene Interaction Graphs**. Disponível em: <https://arxiv.org/pdf/1806.06975.pdf>.

EKINS, S. The next era: Deep learning in pharmaceutical research. **Pharmaceutical research**, [S.l.], v.33, n.11, p.2594–2603, 2016.

FAKOOR, R.; LADHAK, F.; NAZI, A.; HUBER, M. Using deep learning to enhance cancer diagnosis and classification. **Proceedings of the ICML Workshop on the Role of Machine Learning in Transforming Healthcare**, [S.l.], 06 2013.

FENG, Q.; DUEVA, E.; CHERKASOV, A.; ESTER, M. **PADME**: A Deep Learning-based Framework for Drug-Target Interaction Prediction.

FINN, C.; ABBEEL, P.; LEVINE, S. Model-agnostic meta-learning for fast adaptation of deep networks. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING-VOLUME 70, 34., 2017. **Proceedings...** [S.l.: s.n.], 2017. p.1126–1135.

FIORAVANTI, D. et al. Phylogenetic convolutional neural networks in metagenomics. **BMC Bioinformatics**, [S.l.], v.19, n.S2, mar 2018.

FONTES, J. d. A. **Abordagens de selecao de variáveis para classificação e regressão em dados espectrais para controle da qualidade**. 2020. Tese (Doutorado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul.

FUKUSHIMA, K.; MIYAKE, S. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In: **Competition and cooperation in neural nets**. [S.l.]: Springer, 1982. p.267–285.

GAO, J.; YANG, Y.; ZHOU, Y. Grid-based prediction of torsion angle probabilities of protein backbone and its application to discrimination of protein intrinsic disorder regions and selection of model structures. **BMC Bioinformatics**, [S.l.], v.19, n.1, feb 2018.

GAO, K. Y. et al. Interpretable Drug Target Prediction Using Deep Neural Representation. In: IJCAI, 2018. **Anais...** [S.l.: s.n.], 2018. p.3371–3377.

GAO, X.; ZHANG, J.; WEI, Z.; HAKONARSON, H. DeepPolyA: A Convolutional Neural Network Approach for Polyadenylation Site Prediction. **IEEE Access**, [S.l.], v.6, p.24340–24349, 2018.

GARCIA, E. A. C. **Biofísica**. 2. ed..ed. [S.l.]: Sarvier, 2015.

GAWEHN, E.; HISS, J. A.; SCHNEIDER, G. Deep learning in drug discovery. **Molecular informatics**, [S.l.], v.35, n.1, p.3–14, 2016.

GEO. **GEO - Gene Expression Omnibus**. Disponível em: <https://www.ncbi.nlm.nih.gov/gds>.

GÉRON, A. **Mãos à Obra**: Aprendizado de Máquina com Scikit-Learn & TensorFlow. [S.l.]: Alta Books, 2019.

GHASEMI, F.; MEHRIDEHNAVI, A.; PÉREZ-GARRIDO, A.; PÉREZ-SÁNCHEZ, H. Neural network and deep-learning algorithms used in QSAR studies: merits and drawbacks. **Drug Discovery Today**, [S.l.], 2018.

GHEISARI, M.; WANG, G.; BHUIYAN, M. Z. A. A Survey on Deep Learning in Big Data. In: COMPUTATIONAL SCIENCE AND ENGINEERING (CSE) AND EMBEDDED AND UBIQUITOUS COMPUTING (EUC), 2017 IEEE INTERNATIONAL CONFERENCE ON, 2017. **Anais...** [S.l.: s.n.], 2017. v.2, p.173–180.

GOODFELLOW, I. NIPS 2016 tutorial: Generative adversarial networks. **arXiv preprint arXiv:1701.00160**, [S.l.], 2016.

GOODFELLOW, I. et al. Generative adversarial nets. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 2014. **Anais...** [S.l.: s.n.], 2014. p.2672–2680.

GRISCI, B.; CHANDELIER, E.; FELTES, B.; DORN, M. CuMiDa: An Extensively Curated Microarray Database for Benchmarking and Testing of Machine Learning Approaches in Cancer Research. **Journal of Computational Biology**, [S.l.], 2019.

GRUS, J. **Data Science do zero**: Primeiras regras com o Python. [S.l.]: Alta books, 2019.

GTEX. **GTEx - Genotype-Tissue Expression**. Disponível em: <https://gtexportal.org/>.

GUAN, R. et al. Multi-label Deep Learning for Gene Function Annotation in Cancer Pathways. **Scientific Reports**, [S.l.], v.8, n.1, jan 2018.

GUO, Y.; LIU, S.; LI, Z.; SHANG, X. BCDForest: a boosting cascade deep forest model towards the classification of cancer subtypes based on gene expression data. **BMC Bioinformatics**, [S.l.], v.19, n.5, p.118, Apr 2018.

GUO, Y.; SHANG, X.; LI, Z. Identification of cancer subtypes by integrating multiple types of transcriptomics data with deep learning in breast cancer. **Neurocomputing**, [S.l.], may 2018.

GUPTA, A.; WANG, H.; GANAPATHIRAJU, M. Learning structure in gene expression data using deep architectures, with an application to gene clustering. In: BIOINFORMATICS AND BIOMEDICINE (BIBM), 2015 IEEE INTERNATIONAL CONFERENCE ON, 2015. **Anais...** [S.l.: s.n.], 2015. p.1328–1335.

HAMID, M. N.; FRIEDBERG, I. Identifying antimicrobial peptides using word embedding with deep recurrent neural networks. **BioRxiv**, [S.l.], p.255505, 2018.

HARRISON, M. **Machine Learning–Guia de Referência Rápida**: Trabalhando com dados estruturados em Python. [S.l.]: Novatec Editora, 2019.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 2016. **Proceedings...** [S.l.: s.n.], 2016. p.770–778.

HERON, M. P. Deaths: Leading causes for 2016. **National Vital Statistics Reports**, [S.l.], 2018.

HGNC. **HGNC - HUGO Gene Nomenclature Committee**: Home. Disponível em: <https://www.genenames.org/>.

HUA, Y.; GUO, J.; ZHAO, H. Deep Belief Networks and deep learning. In: INTELLIGENT COMPUTING AND INTERNET OF THINGS (ICIT), 2014 INTERNATIONAL CONFERENCE ON, 2015. **Anais...** [S.l.: s.n.], 2015. p.1–4.

HUA, Y.-J. et al. Comparison of normalization methods with microRNA microarray. **Genomics**, [S.l.], v.92, n.2, p.122–128, 2008.

HUANG, L.; LIAO, L.; WU, C. H. Completing sparse and disconnected protein-protein network by deep learning. **BMC Bioinformatics**, [S.l.], v.19, n.1, mar 2018.

IBRAHIM, R.; YOUSRI, N. A.; ISMAIL, M. A.; EL-MAKKY, N. M. Multi-level gene/MiRNA feature selection using deep belief nets and active learning. In: ENGINEERING IN MEDICINE AND BIOLOGY SOCIETY (EMBC), 2014 36TH ANNUAL INTERNATIONAL CONFERENCE OF THE IEEE, 2014. **Anais...** [S.l.: s.n.], 2014. p.3957–3960.

INCA. **INCA - Instituto Nacional de Câncer**. Disponível em: <https://www.inca.gov.br/numeros-de-cancer>.

IORIO, F. et al. A landscape of pharmacogenomic interactions in cancer. **Cell**, [S.l.], v.166, n.3, p.740–754, 2016.

JACOBS, S. et al. **Scaling Deep Learning for Cancer Drug Discovery on HPC Systems**. [S.l.]: Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2018.

JÄRVINEN, A.-K. et al. Are data from different gene expression microarray platforms comparable? **Genomics**, [S.l.], v.83, n.6, p.1164–1168, 2004.

JIMENEZ-CARRETERO, D. et al. Tox_(R)CNN: Deep Learning-Based Nuclei Profiling tool For Drug Toxicity Screening. **bioRxiv**, [S.l.], 2018.

JING, F.; ZHANG, S.-W.; CAO, Z.; ZHANG, S. Combining Sequence and Epigenomic Data to Predict Transcription Factor Binding Sites Using Deep Learning. In: **Bioinformatics Research and Applications**. [S.l.]: Springer International Publishing, 2018. p.241–252.

JING, Y. et al. Deep learning for drug design: An artificial intelligence paradigm for drug discovery in the big data era. **The AAPS journal**, [S.l.], v.20, n.3, p.58, 2018.

JORNAL oglobo. **Câncer é a principal causa de morte em quase 10% das cidades brasileiras**. Disponível em: <https://oglobo.globo.com/sociedade/saude/cancer-a-principal-causa-de-morte-em-quase-10-das-cidades-brasileiras-22595871>.

JUMPER, J. et al. Highly accurate protein structure prediction with AlphaFold. **Nature**, [S.l.], v.596, n.7873, p.583–589, 2021.

KABACOFF, R. I. **R in action**: data analysis and graphics with R. [S.l.]: Simon and Schuster, 2015.

KALININ, A. A. et al. Deep learning in pharmacogenomics: from gene regulation to patient stratification. **Pharmacogenomics**, [S.l.], v.19, n.7, p.629–650, 2018.

KAMARUDIN, J. A. M.; AHYAD, N. A. A.; ABDULLAH, A.; SALLEHUDDIN, R. an enhance cnn-rnn model for predicting functional non-coding variants. **EBSCO Industries**, [S.l.], 2018.

KARIMI, M.; WU, D.; WANG, Z.; SHEN, Y. **DeepAffinity**: Interpretable Deep Learning of Compound-Protein Affinity through Unified Recurrent and Convolutional Neural Networks.

KEANY, E. BorutaShap: A wrapper feature selection method which combines the Boruta feature selection algorithm with Shapley values. **Zenodo https://zenodo.org/record/4247618 (Accessed October 25, 2021)**, [S.l.], 2020.

KHALID, S.; KHALIL, T.; NASREEN, S. A survey of feature selection and feature extraction techniques in machine learning. In: SCIENCE AND INFORMATION CONFERENCE, 2014., 2014. **Anais...** [S.l.: s.n.], 2014. p.372–378.

KHAZE, S. R.; MASDARI, M.; HOJJATKHAH, S. Application of Artificial Neural Networks in estimating participation in elections. **arXiv preprint arXiv:1309.2183**, [S.l.], 2013.

KIM, H. K. et al. Deep learning improves prediction of CRISPR-Cpf1 guide RNA activity. **Nature Biotechnology**, [S.l.], v.36, p.239 EP –, Jan 2018.

KIM, S.; LEE, H.; KIM, K.; KANG, J. Mut2Vec: distributed representation of cancerous mutations. **BMC Medical Genomics**, [S.l.], v.11, n.S2, apr 2018.

KISHAN, K. et al. GNE: A deep learning framework for gene network inference by aggregating biological information. **bioRxiv**, [S.l.], p.300996, 2018.

KUENTZER, F. Otimização e análise de algoritmos de ordenamento de redes proteicas. **Master thesis**, Hardware Design Support Group , Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, RS,Brazil, 2014.

LAB, S. V. **ImageNet database**. Disponível em: <http://image-net.org/>.

LAB, S. V. **ImageNet - Large Scale Visual Recognition Challenge (ILSVRC)**. Disponível em: <http://www.image-net.org/challenges/LSVRC/>.

LAMBERT, S. A. et al. The human transcription factors. **Cell**, [S.l.], v.172, n.4, p.650–665, 2018.

LAN, K. et al. A Survey of Data Mining and Deep Learning in Bioinformatics. **Journal of medical systems**, [S.l.], v.42, n.8, p.139, 2018.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, [S.l.], v.521, n.7553, p.436–444, 2015.

LEI, H. et al. Protein-protein Interactions Prediction via Multimodal Deep Polynomial Network and Regularized Extreme Learning Machine. **IEEE Journal of Biomedical and Health Informatics**, [S.l.], p.1–1, 2018.

LEUNG, M. K.; DELONG, A.; ALIPANAHI, B.; FREY, B. J. Machine learning in genomic medicine: a review of computational problems and data sets. **Proceedings of the IEEE**, [S.l.], v.104, n.1, p.176–197, 2016.

LI, Y.; CHEN, C.-Y.; WASSERMAN, W. W. Deep feature selection: theory and application to identify enhancers and promoters. **Journal of Computational Biology**, [S.l.], v.23, n.5, p.322–336, 2016.

LIANG, M.; LI, Z.; CHEN, T.; ZENG, J. Integrative data analysis of multi-platform cancer data with a multimodal deep learning approach. **IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)**, [S.l.], v.12, n.4, p.928–937, 2015.

LIBBRECHT, M. W.; NOBLE, W. S. Machine learning applications in genetics and genomics. **Nature Reviews Genetics**, [S.l.], v.16, n.6, p.321–332, 2015.

LIN, E.; LANE, H.-Y. Machine learning and systems genomics approaches for multi-omics data. **Biomarker research**, [S.l.], v.5, n.1, p.2, 2017.

LIN, H. et al. Learning rate dropout. **IEEE Transactions on Neural Networks and Learning Systems**, [S.l.], 2022.

LIPTON, Z. C.; BERKOWITZ, J.; ELKAN, C. A critical review of recurrent neural networks for sequence learning. **arXiv preprint arXiv:1506.00019**, [S.l.], 2015.

LIU, J. et al. Cancer Characteristic Gene Selection via Sample Learning Based on Deep Sparse Filtering. **Scientific Reports**, [S.l.], v.8, n.1, may 2018.

LLOMBART-BOSCH, A. et al. Histological heterogeneity of Ewing's sarcoma/PNET: an immunohistochemical analysis of 415 genetically confirmed cases with clinical support. **Virchows Archiv**, [S.l.], v.455, n.5, p.397–411, 2009.

LOPEZ-RINCON, A. et al. Evolutionary optimization of convolutional neural networks for cancer miRNA biomarkers classification. **Applied Soft Computing**, [S.l.], v.65, p.91–100, apr 2018.

LYU, B.; HAQUE, A. Deep Learning Based Tumor Type Classification Using Gene Expression Data. **bioRxiv**, [S.l.], jul 2018.

MA, J. et al. Using deep learning to model the hierarchical structure and function of a cell. **Nature methods**, [S.l.], v.15, n.4, p.290, 2018.

MAATEN, L. v. d.; HINTON, G. Visualizing data using t-SNE. **Journal of machine learning research**, [S.l.], v.9, n.Nov, p.2579–2605, 2008.

MACHADO, I. et al. Review with novel markers facilitates precise categorization of 41 cases of diagnostically challenging,"undifferentiated small round cell tumors". A clinicopathologic, immunophenotypic and molecular analysis. **Annals of Diagnostic Pathology**, [S.l.], v.34, p.1–12, 2018.

MAMOSHINA, P. et al. Machine Learning on Human Muscle Transcriptomic Data for Biomarker Discovery and Tissue-Specific Drug Target Identification. **Frontiers in Genetics**, [S.l.], v.9, jul 2018.

MAMOSHINA, P.; VIEIRA, A.; PUTIN, E.; ZHAVORONKOV, A. Applications of deep learning in biomedicine. **Molecular pharmaceutics**, [S.l.], v.13, n.5, p.1445–1454, 2016.

MENG, L.; DING, S.; XUE, Y. Research on denoising sparse autoencoder. **International Journal of Machine Learning and Cybernetics**, [S.l.], v.8, n.5, p.1719–1729, 2017.

MERK, D.; FRIEDRICH, L.; GRISONI, F.; SCHNEIDER, G. De Novo Design of Bioactive Small Molecules by Artificial Intelligence. **Molecular Informatics**, [S.l.], v.37, n.1-2, p.1700153, jan 2018.

MIN, S.; LEE, B.; YOON, S. Deep learning in bioinformatics. **Briefings in bioinformatics**, [S.l.], p.bbw068, 2016.

MIRANDA, L. J.; HU, J. A Deep Learning Approach Based on Stacked Denoising Autoencoders for Protein Function Prediction. In: IEEE 42ND ANNUAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE (COMPSAC), 2018., 2018. **Anais...** IEEE, 2018.

MORAIS, D. A.; ALMEIDA, R.; DALMOLIN, R. J. Transcriptogramer: an R/Bioconductor package for transcriptional analysis based on protein–protein interaction. **Bioinformatics**, [S.l.], 2019.

MüLLER, A. T.; HISS, J. A.; SCHNEIDER, G. Recurrent Neural Network Model for Constructive Peptide Design. **Journal of Chemical Information and Modeling**, [S.l.], v.58, n.2, p.472–479, jan 2018.

NAZARI, I.; CHONG, K. T. Deep Learning Approaches For Prediction of Sequence Specificities of DNA-Binding Proteins. **33st ICROS Annual Conference (ICROS 2018)**, [S.l.], 2018.

NCBI. **NCBI - National Center for Biotechnology Information**. Disponível em: <https://ftp.ncbi.nlm.nih.gov>.

NCI, N. C. I. **TARGET**: Therapeutically Applicable Research to Generate Effective Treatments.

NG, A. **Machine learning yearning**. [S.l.: s.n.], 2017. v.139.

NGIAM, J. et al. Sparse filtering. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 2011. **Anais...** [S.l.: s.n.], 2011. p.1125–1133.

NGUYEN, T. H. et al. **Disease Classification in Metagenomics with 2D Embeddings and Deep Learning**.

NIEPERT, M.; AHMED, M.; KUTZKOV, K. Learning convolutional neural networks for graphs. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 2016. **Anais...** [S.l.: s.n.], 2016. p.2014–2023.

ONIMARU, K.; NISHIMURA, O.; KURAKU, S. A regulatory-sequence classifier with a neural network for genomic information processing. **bioRxiv**, [S.l.], p.355974, 2018.

OUGHTRED, R. et al. The BioGRID interaction database: 2019 update. **Nucleic acids research**, [S.l.], v.47, n.D1, p.D529–D541, 2019.

OUYANG, Z. et al. Accurate identification of RNA editing sites from primitive sequence with deep neural networks. **Scientific Reports**, [S.l.], v.8, n.1, apr 2018.

PAN, X.; RIJNBEEK, P.; YAN, J.; SHEN, H.-B. Prediction of RNA-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks. **BMC Genomics**, [S.l.], v.19, n.1, jul 2018.

PANTELEEV, J.; GAO, H.; JIA, L. Recent Applications of Machine Learning in Medicinal Chemistry. **Bioorganic & Medicinal Chemistry Letters**, [S.l.], 2018.

PÉREZ-SIANES, J.; PÉREZ-SÁNCHEZ, H.; DÍAZ, F. Virtual screening: a challenge for deep learning. In: INTERNATIONAL CONFERENCE ON PRACTICAL APPLICATIONS OF COMPUTATIONAL BIOLOGY & BIOINFORMATICS, 10., 2016. **Anais...** [S.l.: s.n.], 2016. p.13–22.

PERRONE, G. Transcriptograma em duas dimensões. **Master thesis**, Instituto de Física, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS,Brazil, 2013.

PFEIFFENBERGER, E.; BATES, P. A. Predicting improved protein conformations with a temporal deep recurrent neural network. **PloS one**, [S.l.], v.13, n.9, p.e0202652, 2018.

PRABHU. **Understanding of Convolutional Neural Network (CNN) - Deep Learning**. Disponível em: <https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>.

PRAT, A. et al. Clinical implications of the intrinsic molecular subtypes of breast cancer. **The Breast**, [S.l.], v.24, p.S26–S35, 2015.

PUTIN, E. et al. Deep biomarkers of human aging: application of deep neural networks to biomarker development. **Aging (Albany NY)**, [S.l.], v.8, n.5, p.1021, 2016.

RAMPASEK, L.; GOLDENBERG, A. Tensorflow: Biology's gateway to deep learning? **Cell systems**, [S.l.], v.2, n.1, p.12–14, 2016.

RAVÌ, D. et al. Deep learning for health informatics. **IEEE journal of biomedical and health informatics**, [S.l.], v.21, n.1, p.4–21, 2017.

REN, J. et al. **Identifying viruses from metagenomic data by deep learning**.

RENHULDT, N. T. **Protein contact prediction based on the Tiramisu deep learning architecture (Dissertation)**. Kth royal institute of technology, School of electrical engineering and computer science: [s.n.], 2018.

RYBARCZYK-FILHO, J. L. et al. Towards a genome-wide transcriptogram: the Saccharomyces cerevisiae case. **Nucleic acids research**, [S.l.], v.39, n.8, p.3005–3016, 2010.

RYU, S.; LIM, J.; KIM, W. Y. **Deeply learning molecular structure-property relationships using graph attention neural network**.

SALEKIN, S. **Deep Learning Models for Predicting Genomic and Transcriptomic Functional Sites**. 2018. Tese (Doutorado em Ciência da Computação) — The University of Texas at San Antonio.

SERRANO, A. et al. Accelerating Drugs Discovery with Deep Reinforcement Learning. In: INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING COMPANION, 47., 2018. **Proceedings...** ACM Press, 2018.

SHARIFI-NOGHABI, H. et al. Deep Genomic Signature for early metastasis prediction in prostate cancer. **BioRxiv**, [S.l.], p.276055, 2018.

SHRIKUMAR, A.; GREENSIDE, P.; KUNDAJE, A. Learning important features through propagating activation differences. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING-VOLUME 70, 34., 2017. **Proceedings...** [S.l.: s.n.], 2017. p.3145–3153.

SILVA, S. R. da; PERRONE, G. C.; DINIS, J. M.; ALMEIDA, R. M. de. Reproducibility enhancement and differential expression of non predefined functional gene sets in human genome. **BMC genomics**, [S.l.], v.15, n.1, p.1181, 2014.

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

SMYTH, G. K.; YANG, Y. H.; SPEED, T. Statistical issues in cDNA microarray data analysis. **Functional genomics: methods and protocols**, [S.l.], p.111–136, 2003.

SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. **Journal of machine learning research**, [S.l.], v.15, n.1, p.1929–1958, 2014.

STRING. **STRING**: functional protein association networks. Disponível em: <https://string-db.org/>.

SUN, D.; WANG, M.; LI, A. A multimodal deep neural network for human breast cancer prognosis prediction by integrating multi-dimensional data. **IEEE/ACM Transactions on Computational Biology and Bioinformatics**, [S.l.], p.1–1, 2018.

SUN, K. et al. GeneCT: a generalizable cancerous status and tissue origin classifier for pan-cancer biopsies. **Bioinformatics (Oxford, England)**, [S.l.], 2018.

SUNG, F. et al. Learning to compare: Relation network for few-shot learning. In: IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 2018. **Proceedings...** [S.l.: s.n.], 2018. p.1199–1208.

SUNSERI, J.; KING, J. E.; FRANCOEUR, P. G.; KOES, D. R. Convolutional neural network scoring and minimization in the D3R 2017 community challenge. **Journal of Computer-Aided Molecular Design**, [S.l.], jul 2018.

SZKLARCZYK, D. et al. The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets. **Nucleic acids research**, [S.l.], v.49, n.D1, p.D605–D612, 2021.

TARTAGLIONE, E.; PERLO, D.; GRANGETTO, M. Post-synaptic potential regularization has potential. In: INTERNATIONAL CONFERENCE ON ARTIFICIAL NEURAL NETWORKS, 2019. **Anais...** [S.l.: s.n.], 2019. p.187–200.

TCGA. **TCGA - The Genome Cancer Atlas**. Disponível em: <https://cancergenome.nih.gov/>.

UNTERTHINER, T. et al. Deep learning as an opportunity in virtual screening. In: DEEP LEARNING WORKSHOP AT NIPS, 2014. **Proceedings...** [S.l.: s.n.], 2014.

WANG, J.; CAO, H.; ZHANG, J. Z. H.; QI, Y. Computational Protein Design with Deep Learning Neural Networks. **Scientific Reports**, [S.l.], v.8, n.1, apr 2018.

WANG, K.; HOEKSEMA, J.; LIANG, C. piRNN: deep learning algorithm for piRNA prediction. **PeerJ**, [S.l.], v.6, p.e5429, aug 2018.

WANG, L.; CHENG, J. Protein Secondary Structure Prediction Using AutoEncoder Network and Bayes Classifier. **IOP Conference Series: Materials Science and Engineering**, [S.l.], v.322, n.6, p.062008, mar 2018.

WANG, S.; FU, L.; YAO, J.; LI, Y. The application of deep learning in biomedical informatics. In: INTERNATIONAL CONFERENCE ON ROBOTS & INTELLIGENT SYSTEM (ICRIS), 2018., 2018. **Anais...** [S.l.: s.n.], 2018.

WHO, W. H. O. **World Health Organization - Cancer - Key facts**. Disponível em: <http://www.who.int/news-room/fact-sheets/detail/cancer>.

WICK, R. R.; JUDD, L. M.; HOLT, K. E. Deepbinner: Demultiplexing barcoded Oxford Nanopore reads with deep convolutional neural networks. **PLoS computational biology**, [S.l.], v.14, n.11, p.e1006583, 2018.

WU, K.; WEI, G.-W. Quantitative Toxicity Prediction Using Topology Based Multitask Deep Neural Networks. **Journal of Chemical Information and Modeling**, [S.l.], v.58, n.2, p.520–531, jan 2018.

XIAO, Y.; WU, J.; LIN, Z.; ZHAO, X. A deep learning-based multi-model ensemble method for cancer prediction. **Computer Methods and Programs in Biomedicine**, [S.l.], v.153, p.1–9, jan 2018.

XIE, R.; QUITADAMO, A.; CHENG, J.; SHI, X. A predictive model of gene expression using a deep learning framework. In: BIOINFORMATICS AND BIOMEDICINE (BIBM), 2016 IEEE INTERNATIONAL CONFERENCE ON, 2016. **Anais...** [S.l.: s.n.], 2016. p.676–681.

XU, J. **An Intuitive Guide to Deep Network Architectures**. Disponível em: <https://towardsdatascience.com/an-intuitive-guide-to-deep-network-architectures-65fdc477db41>.

YADAV, S.; GUPTA, M.; BIST, A. S. Prediction of Ubiquitination Sites Using UbiNets. **Advances in Fuzzy Systems**, [S.l.], v.2018, p.1–10, 2018.

YANG, C. et al. LncADeep: an ab initio lncRNA identification and functional annotation tool based on deep learning. **Bioinformatics**, [S.l.], may 2018.

YUE, T.; WANG, H. Deep Learning for Genomics: A Concise Overview. **arXiv preprint arXiv:1802.00810**, [S.l.], 2018.

ZENG, W.; GLICKSBERG, B.; LI, Y.; CHEN, B. Selecting precise reference normal tissue samples for cancer research using a deep learning approach. **bioRxiv**, [S.l.], jun 2018.

ZHANG, D.; ZOU, L.; ZHOU, X.; HE, F. Integrating Feature Selection and Feature Extraction Methods With Deep Learning to Predict Clinical Outcome of Breast Cancer. **IEEE Access**, [S.l.], v.6, p.28936–28944, 2018.

ZHANG, H.; GOODFELLOW, I.; METAXAS, D.; ODENA, A. Self-attention generative adversarial networks. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 2019. **Anais...** [S.l.: s.n.], 2019. p.7354–7363.

ZHANG, Q.; YANG, L. T.; CHEN, Z.; LI, P. A survey on deep learning for big data. **Information Fusion**, [S.l.], v.42, p.146–157, 2018.

ZHANG, R. et al. MetaGAN: An Adversarial Approach to Few-Shot Learning. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 2018. **Anais...** [S.l.: s.n.], 2018. p.2371–2380.

ZHOU, J. et al. CNNH_PSS: protein 8-class secondary structure prediction by convolutional neural network with highway. **BMC Bioinformatics**, [S.l.], v.19, n.S4, may 2018.

ZHOU, J. et al. Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk. **Nature Genetics**, [S.l.], v.50, n.8, p.1171–1179, 2018.

ZÖLLNER, S. K. et al. Ewing sarcoma—diagnosis, treatment, clinical challenges and future perspectives. **Journal of clinical medicine**, [S.l.], v.10, n.8, p.1685, 2021.

ÖZTüRK, H.; OZKIRIMLI, E.; ÖZGüR, A. **DeepDTA**: Deep Drug-Target Binding Affinity Prediction.

Apêndices

APÊNDICE A – Trabalhos Relacionados - Aplicação de Deep Learning em Bioinformática

As Tabelas 46, 47, 48, 49 e 50, a seguir, apresentam alguns dos trabalhos selecionados na pesquisa bibliográfica realizada (ver Capítulo 3), especificando as técnicas de *Deep Learning* (DL) utilizadas (abreviações: CNN - Convolutional Neural Network; GAN - Generative Adversarial Network; MGC - Molecular Graph Convolution; NN - Neural Network; RNN - Recurrent Neural Network; SVM - Support Vector Machine; TF - Transcriptional Factor).

Tabela 46 – Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 1/5.

Área	**Fonte**	**Método Proposto**	**Objetivo**	**Técnicas**	**Diferenciais**	**Entrada**	**Saída**
Transcriptômica	(GUO et al., 2018)	BCDForest	Diagnóstico - Classificação de subtipos de câncer baseado em dados de expressão gênica	Ensemble de florestas de árvores de decisão profundas com as modificações: 1) possui método multi-class-grained scanning que favorece a diversidade nos ensembles; 2) apresenta uma estratégia boosting para priorizar features importantes	Visa poder ser aplicável em datasets com poucas amostras	Expressão gênica - RNAseq e microarray	Multi-classes - Classes de subtipos de câncer
Genômica - Metagenômica	(REN et al., 2018)	DeepVirFinder	Diagnóstico - Identifica vírus a partir de dados metagenômicos	CNN que extrai intensidades de motifs nas sequências e as usa para a predição de existência de sequência viral	Não requer a realização de alinhamento (ou da existência de sequências homólogas) e de genoma de referência	Sequenciamento de amostras metagenômicas	Probabilidade de existência de sequência viral na amostra
Genômica	(ZHOU et al., 2018)	ExPecto	Compreensão de efeitos de mutações - predição de expressão gênica a partir da sequência de DNA - predição de SNPs funcionais	1) CNN analisa as sequências de DNA com janela deslizante e retorna padrões epigenéticos (marcas de histonas, ligação TF e perfis de acessibilidade da cromatina). 2) Transformação espacial para redução dimensional. 3) regressão linear e gradient boosting algorithm. Predições específicas para cada tipo de célula.	Trabalha não apenas com organismos modelos com poucas regiões não codificantes - focado para humanos	Sequência de DNA de regiões regulatórias próximas a promotores	Níveis de expressão gênica e risco de doenças. Validado com dados de expression quantitative trait loci eQTL
Genômica	(KAMARUDIN et al., 2018)	-	Predição se sequências de DNA não codificantes são funcionais	CNN, RNN e swarm optimization. Sequência de DNA de entrada é transformada em uma matriz binária de 4 linhas. 1) CNN para aprender motivos regulatórios. 2) RNN- bidirectional long term memory (BLSTM) para capturar dependências de 'longo termo'	Não requer seleção de parâmetros manuais. Autores reportam ser um sistema melhorado em relação a prévios DanQ e DeepSEA	Sequências de DNA	Score (probabilidade) da sequência ser funcional

Tabela 47 – Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 2/5.

Área	Fonte	Método Proposto	Objetivo	Técnicas	Diferenciais	Entrada	Saída
Genômica - Epigenômica	(JING et al., 2018)	-	Predição de sítios de ligação de fatores de transcrição (Transcription Factor - TF)	CNN. Sequência de DNA é codificada como uma imagem de 4 canais que assumem valores binários (0 ou 1) para indicar a presença das bases (A,T,C,G)	Utiliza também dados epigenéticos (modificações de histonas e acessibilidade de cromatina) além da sequência de DNA. DL automaticamente extrai features	Sequência de DNA; ChIP-seq; DNase-seq	Probabilidade da sequência de DNA conter um sítio de ligação de fatores de transcrição (TFBS)
Transcriptômica	(SALEKIN, 2018)	DeepSNR	Parte 1- Predição de sítios de ligação de fatores de transcrição (TF). Parte 2 - extrair padrões de sequências de pre-mRNA	Parte 1 - Nova arquitetura inspirada na Deconvolutional neural network (deconvNet): DeepSNR. Parte 2 - GAN	Parte 1 - melhor acurácia do que métodos baseados em busca de motivos. Parte 2 - Permite classificação mesmo quando a base de dados possua poucas amostras e seja altamente não balanceada	Parte 1 - ChIP-exonuclease. Parte 2 - sequência de pre-mRNA	Parte1 - Probabilidade da sequência de RNA ter um sítio de ligação de TF. Parte 2 - Predição de diferentes tipos de modificações de RNA
Transcriptômica	(OUYANG et al., 2018)	DeepRed	Identificação de sítios de edição de RNA	Múltiplos ensembles de DNNs; sequência codificada com One-hot-encoding	Não requer conhecimento a priori e genomas de referência anotados	Sequências de RNA - RNAseq	Probabilidade da sequência ter um sítio de edição de RNA
Transcriptômica	(BAEK et al., 2018)	lncRNA-net	Identificação de RNA longo não codificante (elementos regulatórios)	RNN para modelagem de sequências de RNA e CNN para detectar stop codons para indicar ORFs (open reading frame)	Melhor acurácia que métodos prévios	Sequências de RNA	Probabilidade da sequência ser um lncRNA
Transcriptômica - Proteômica	(YANG et al., 2018)	lncADeep	Identificação e anotação funcional de RNAs longos não codificantes	Deep belief network, enriquecimento funcional com KEGG e Reactome	Prediz interações com proteínas e integra análises de enriquecimento KEGG e Reactome. Supera estado de arte	Sequência, informações de homologia, atributos estruturais (estrutura secundária, pontes de hidrogênio e forças de Van der Waals) de RNAs e proteínas	Identificação de RNA não codificante; predição de interação com determinada proteína; funcionalidade de lncRNA através do enriquecimento funcional com KEGG e Reactome
Transcriptômica	(CAO et al., 2018)	lncLocator	Predição de localização subcelular de lncRNA	Padrões de frequência de k-mers e features aprendidos de forma não supervisionada através de modelo profundo (stacked autoencoder) são utilizados em classificadores SVM e florestas aleatórias. Resultados dos classificadores combinados com estratégia de stacked ensemble	Primeiro método para predição de localização de lncRNA	Sequência de lncRNA	Localização do lncRNA (entre cinco classes: citoplasma, núcleo, citosol, ribossomo, exossomo)
Transcriptômica	(PAN et al., 2018)	iDeepS	Identificar sequências de sítios de ligação RNA-proteína e predição de motivos estruturais de RNA	CNN e RNN (Bidirectional Long Short Term Memory - BLSTM); sequência e estrutura secundária predita pelo método RNAshape são codificadas com one-hot encoding	Obtém simultaneamente a identificação de sítios de ligação RNA-proteína e motivos estruturais. Supera estado da arte na predição de sítios de ligação	Sequência e estrutura secundária de RNA. Avaliações feitas com dados de CLIP-seq	Probabilidade da sequência de RNA conter sítio de ligação de proteínas. Os features aprendidos pelas CNNs podem ser convertidos em PWM (position weight matrices) de forma a permitir a identificação de motivos estruturais
Fármacos	(RYU; LIM; KIM, 2018)	-	Aprender relações entre estruturas e propriedades moleculares para permitir o projeto de novas moléculas	Graph attention neural network (permite considerar a vizinhança de átomos)	Utiliza mecanismo de atenção. Problema desafiador, pois existe limitada quantidade de dados químicos	Estrutura química de molécula	Propriedades químicas da molécula (polaridade, solubilidade, energia)
Proteômica - Fármacos	(JIMENEZ-CARRETERO et al., 2018)	Kdeep	Predição de afinidade entre proteína e ligante - acelerar virtual screening (triagem virtual)	3D CNN	Alta velocidade de predição; fácil uso (disponível para download); boa performance	Representação 3D em voxels da proteína e do ligante	Afinidades entre proteína e ligante

Tabela 48 – Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 3/5.

Área	Fonte	Método Proposto	Objetivo	Técnicas	Diferenciais	Entrada	Saída
Proteômica - Fármacos	(SUNSERI et al., 2018)	-	Predição de atividade do ligante e refinamento da pose do ligante	CNN	Obtém melhor performance que o Vina sem requerer inspeção manual de um usuário especialista	Informação estrutural do ligante e receptor	Atividade do ligante e identificação se a pose do ligante está correta (próxima ou distante) da verdadeira posição de ligação
Fármacos	(JACOBS et al., 2018)	Livermore Big Artificial Neural Network (LBANN)	Resultados preliminares de aprendizagem de features de fármacos visando posterior descoberta de resposta de linhagens celulares de cânceres a fármacos	Autoencoders	Desenvolvimento de toolkit	Atributos de fármacos (informações químicas)	Extração de features relevantes de fármacos
Proteômica - Fármacos	(SERRANO et al., 2018)	DQN-Docking	Resultados preliminares - Predição de docking proteína-ligante	Deep reinforcement learning - Deep Q Network. Visa ensinar um agente como o mesmo deve ligar um ligante ao receptor. Um esquema de metaheurística METADOCK pontua os movimentos	Resultados promissores (preliminares)	Estrutura química do ligante e receptor	Pose do ligante de ligação ao receptor
Genômica	(GAO et al., 2018)	DeepPolyA	Predição de sítio de poliadenilação (papel em regulação gênica)	CNN	Supera métodos prévios DanQ, DeepSEA, VGG. Não requer engenharia manual para obter features	Sequências de DNA de Arabidopsis	Probabilidade da sequência ter sítio de poliadenilação
Proteômica	(YADAV; GUPTA; BIST, 2018)	UbiNets	Predição de sítio de ubiquitinação (controla a atividade de várias proteínas)	Nova arquitetura de rede neural multi-layer perceptron denominada de UbiNets baseada em Densely Connected CNNs (DenseNet)	Explora o uso de DL. Propõe nova arquitetura de rede neural	Sequências de proteínas	Probabilidade da sequência ter um sítio de ubiquitinação
Proteômica - Fármacos	(FENG et al., 2018)	PADME	Predição de interação entre fármaco e proteína alvo	Molecular Graph Convolution (MGC) - Graph convolutional network	Visa também poder ser utilizado quando a proteína alvo não está presente no conjunto de treinamento. Primeiro trabalho a usar MGC para aprender features de compostos	Informações de sequências de proteínas (descritores PSC) e informações de sequência dos compostos (representação SMILES)	Força de interação entre compostos e proteínas
Proteômica - Fármacos	(DU et al., 2018)	-	Predição de interação entre fármaco e proteína alvo	Deep feed forward neural network e modelo linear generalizado	Acurácia superior a prévios métodos	Representações numéricas de compostos químicos e de sequências de proteínas obtidas pelo PyDPI (pacote python)	Score de interação entre composto e proteína
Proteômica - Fármacos	(GAO et al., 2018)	-	Predição de interação entre fármaco e proteína alvo - fornecer insights biológicos de interpretação da predição	Mecanismo de atenção two way; Long Short Term Memory RNN; Graph CNN	Não requer conhecimento especialista sobre o tema e engenharia de features. Fornece meios para interpretar a predição do método. Resultados superam métodos prévios	Sequência de aminoácidos de proteínas e estrutura química dos compostos	Score de interação entre composto e proteína
Genômica - Fármacos	(CHANG et al., 2018)	CDRscan	Predição de eficácia de fármacos a partir de assinatura genômica de cânceres (medicina de precisão)	CNNs e docking virtual	Primeira aplicação de DL em predição de viabilidade de reposicionamento de fármacos	Atributos de fármacos (descritor PaDEL) e lista de mutações somáticas de linhagens celulares de cânceres	Eficácia do fármaco
Proteômica - Fármacos	(ÖZTÜRK; OZKIRIMLI; ÖZGÜR, 2018)	DeepDTA	Predição de afinidade de ligação entre fármaco e proteína alvo	CNN	Supera estado-da-arte, usa apenas informação da sequência da proteína e fármaco	Informação da sequência da proteína e do fármaco	Score de afinidade de ligação proteína-fármaco
Proteômica	(CHENG et al., 2018)	-	Predição de epitopos de células B (projeto de vacinas)	Long Short Term Memory Networks; SVM; ELMAN - RNN; feature relevância de pares de aminoácidos	O problema de predição de epitopos ainda não está resolvido. Obtém alta acurácia	Sequência de aminoácidos - relevância de pares de aminoácidos	Classificação epitopo ou não

Tabela 49 – Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 4/5.

Área	Fonte	Método Proposto	Objetivo	Técnicas	Diferenciais	Entrada	Saída
Genômica	(WICK; JUDD; HOLT, 2018)	Deepbinner	Demultiplexação de leituras de Oxford Nanopore entre diferentes amostras - barcode DNA (sequenciadas simultaneamente)	CNN	Menor taxa de leituras não classificadas e maior precisão de demultiplexação. Open source	Sinais elétricos das leituras	Classificação - barcode da amostra
Genômica	(KIM et al., 2018)	Mut2Vec	Obter representação embedded de mutações em cânceres. Diferir entre mutações passageiras e aquelas que causam cânceres	Skip-Gram; representação que pode ser utilizada em classificadores DL de tipo de câncer (mas não é demonstrado no trabalho)	Desafio pouca quantidade de dados	Perfis de mutações genéticas de cânceres, literatura biomédica, redes PPI	Representação de mutações em cânceres. Determinação de mutações passageiras e aquelas que causam cânceres
Transcriptômica	(LOPEZ-RINCON et al., 2018)	-	Classificação de biomarcadores de microRNA em cânceres (diagnóstico de tecido de origem do tumor e subtipos tumorais)	CNN com otimização dos hiperparâmetros por métodos evolucionários	Problema complexo, comparação com outros métodos	Dados de sequenciamento de isoformas de microRNA de 29 tipos de cânceres	Classificação de tipo de câncer
Proteômica-Fármacos	(KARIMI et al., 2018)	DeepAffinity	Predição de afinidade entre composto e proteína	Atoencoder, RNN e CNN; transfer learning para novas classes de proteínas com poucos dados rotulados; mecanismo de atenção para interpretabilidade	Usa apenas informações das sequências. Método para codificar representação e realizar predição. Usa dados rotulados e não rotulados. Emprega mecanismo de atenção para interpretabilidade	Sequência de proteínas e estrutura de compostos (SMILES)	Afinidade entre composto e proteína
Proteômica	(WANG et al., 2018)	-	Predição de probabilidade de cada um dos 20 aminoácidos estarem em cada resíduo de uma proteína. Projeto de proteínas	Deep multi-layer network	Tarefa desafiadora	Propriedades estruturais de proteínas obtidas por cristalografia	Aminoácido em cada resíduo da proteína
Proteômica	(MIRANDA; HU, 2018)	-	Predição de funcionalidade de proteínas	Stacking denoising autoencoders; multilabel SVM	A aprendizagem de representações por autoencoders melhora a performance resolvendo o problema de existência de ruído em datasets de proteínas	Dados de proteínas	Se a proteína (apresentada como entrada) possui determinada funcionalidade (predição para vários tipos de funcionalidades)
Genômica	(NAZARI; CHONG, 2018)	residual-inception	Predição de síto de ligação proteína-DNA	Novo método inspirado em inception v4 CNN	Propõe novo método inspirado na DNN que obteve melhor acurácia no Large Scale Visual Recognition Challenge - ILSVRC2015	Sequência de DNA	Probabilidade da sequência conter sítio de ligação DNA-proteína
Proteômica - fármacos	(HAMID; FRIEDBERG, 2018)	-	Identificação de peptídeos antimicrobianos (desenvolvimento de antibióticos)	Word embedding e RNN	Identificação de peptídeos antimicrobianos putativos; método não emprega buscas por similaridade entre sequências	Sequências de proteínas	Classificação se a sequência apresenta peptídeos antimicrobianos
Transcriptômica - Genômica	(SHARIFI-NOGHABI et al., 2018)	Deep Genomic Signature (DGS)	Identificação de assinatura para predição precoce de metástase	Denoising Autoencoder; transfer learning entre dados rotulados e não rotulados; regressão logística para predição de metástase	Desafio - alta dimensionalidade de dados transcriptômicos, poucas informações sobre sobrevida de pacientes. Busca explorar dados genômicos sem respectiva informação de sobrevida clínica. Realizam análise de vias para avaliar a assinatura genômica obtida	Dados de expressão de microarray de tumor e anotação genômica	Predição de metástase e assinatura genômica

Tabela 50 – Relação de trabalhos que aplicam DL em problemas de bioinformática - Parte 5/5.

Área	Fonte	Método Proposto	Objetivo	Técnicas	Diferenciais	Entrada	Saída
Proteômica	(GAO; YANG; ZHOU, 2018)	Parte do pacote SPIDER2	Predição de probabilidades de ângulos de torção (útil para predição de estrutura). Identificação de regiões intrinsecamente desordenadas em proteínas	Predição baseada em grid (bins de 5º graus). Stacked Autoencoders e Deep feed forward NN	Melhor acurácia que métodos prévios. Disponível online	Sequência de aminoácidos da proteína e atributos de cada resíduo (propriedades do aminoácido e matriz PSSM gerada pelo PSI-BLAST).	Bin referente ao ângulo de torção. Maiores flutuações entre ângulos de torção são usadas para determinar regiões intrinsecamente desordenadas
Transcriptômica	(WANG; HOEKSEMA; LIANG, 2018)	piRNN	Identificação de piRNA (Piwi-interacting RNA) - classe de RNAs não codificantes pequenos	CNN	Melhor acurácia que métodos prévios. Disponível online. Tarefa desafiadora: falta de sequências conservadas e padrões estruturais	Matriz de frequência de k-mer representam a sequência de RNA	Probabilidade da sequência ser um piRNA
Transcriptômica - Genômica	(BRETSCHNEIDER et al., 2018)	COSSMO - Competitive Splice Site Model	Predição de sítios de splicing alternativo	Quatro arquiteturas de DNNs: CNN, Communication NN, Long Short Term Memory NN, Residual NN	Analisa efeitos competitivos na seleção de sítios de splicing (a probabilidade de um sítio de splicing ser usado também depende da força dos sítios vizinhos). Disponível online. Aprendeu a reconhecer sequências consenso referentes a sítios de splicing, apresenta boa acurácia.	Dados de RNA-seq. Valores de Percent selected index (PSI) de eventos de splicing obtidos a partir de anotação de genomas são usados para o treinamento	Predição de percent selected index (PSI) de splicing
Transcriptômica	(DIZAJI; WANG; HUANG, 2018)	-	Predição de expressão de genes alvos a partir de genes de referência (menor custo para obtenção de dados de expressão gênica - avaliar apenas genes de referência)	GAN	Melhor performance que métodos prévios.	Expressão de genes de referência	Expressão de genes alvos
Proteômica	(PFEIFFENBERGER; BATES, 2018)	DeepTrajectory	Predição de conformação de proteínas otimizada com informação de dinâmica molecular	Temporal RNN	Primeira implementação temporal de DL para dados de dinâmica molecular. Supera estado da arte de métodos que não consideram dependência temporal	Dados de dinâmica molecular (DM)	A cada snapshot da DM o método determina se houve melhora no estado conformacional da proteína; piora; ou se não houveram diferenças
Proteômica	(RENHULDT, 2018)	TiramiProt	Predição de contato entre resíduos de uma proteína (ajuda a prever a estrutura da proteína)	Tiramisu NN	Supera métodos do estado da arte	Sequência de proteínas, dados de alinhamento de proteínas	Probabilidade de contato entre resíduos

APÊNDICE B – Problemas e Datasets de Expressão Gênica

As Tabelas 51, 52, 53, 54, 55, 56, 57, 58 e 59, a seguir, apresentam os problemas (divisões de classes) para cada um dos datasets de expressão gênica (*microarray*) usados. Os experimentos com SVM foram feitos com todos os problemas citados nas figuras; o XGBoost foi feito em apenas 31 problemas (ver coluna 'XGBoost'); e as NNs foram executadas em 11 problemas (ver coluna 'NNs').

Tabela 51 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 1/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(1, 1)			GSE2553	Bone_or_soft_tissue	entre as 5 classes com mais amostras (acima de 10 amostras)	123	5	Malignant_Fibrous_Histiocytoma: 38 Liposarcoma: 33 Ewing_Sarcoma: 19 Leiomyosarcoma: 17 Synovial_Cell_Sarcoma: 16
(1, 2)			GSE2553	Bone_or_soft_tissue	Malignant_Fibrous_Histiocytoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 143 Malignant_Fibrous_Histiocytoma: 38
(1, 3)	x		GSE2553	Bone_or_soft_tissue	Liposarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 148 Liposarcoma: 33
(1, 4)	x	x	GSE2553	Bone_or_soft_tissue	Ewing_Sarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 162 Ewing_Sarcoma: 19
(1, 5)			GSE2553	Bone_or_soft_tissue	Leiomyosarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 164 Leiomyosarcoma: 17
(1, 6)			GSE2553	Bone_or_soft_tissue	Synovial_Cell_Sarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 165 Synovial_Cell_Sarcoma: 16
(1, 7)	x		GSE2553	Bone_or_soft_tissue	Sarcoma_NOS x outros (todas as classes com 5 ou mais amostras)	181	2	other: 171 Sarcoma_NOS: 10
(1, 8)			GSE2553	Bone_or_soft_tissue	Fibrosarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 174 Fibrosarcoma: 7
(1, 9)			GSE2553	Bone_or_soft_tissue	Malignant_Peripheral_Nerve_Sheath_Tumor x outros (todas as classes com 5 ou mais amostras)	181	2	other: 175 Malignant_Peripheral_Nerve_Sheath_Tumor: 6
(1, 10)			GSE2553	Bone_or_soft_tissue	Rhabdomyosarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 175 Rhabdomyosarcoma: 6
(1, 11)			GSE2553	Bone_or_soft_tissue	Malignant_Hemangiopericytoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 175 Malignant_Hemangiopericytoma: 6
(1, 12)			GSE2553	Bone_or_soft_tissue	Dermatofibrosarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 176 Dermatofibrosarcoma: 5
(1, 13)	x		GSE2553	Bone_or_soft_tissue	Gastrointestinal_Stromal_Tumor x outros (todas as classes com 5 ou mais amostras)	181	2	other: 176 Gastrointestinal_Stromal_Tumor: 5
(1, 14)	x		GSE2553	Bone_or_soft_tissue	Osteosarcoma x outros (todas as classes com 5 ou mais amostras)	181	2	other: 176 Osteosarcoma: 5
(2, 1)			GSE17679	Bone_or_soft_tissue	Ewing, cell_line_Ewing, normal	73	3	Ewing: 44 normal_skeletal_muscle: 18 cell_line_Ewing: 11
(2, 2)			GSE17679	Bone_or_soft_tissue	câncer x normal - juntando as cell line	73	2	cancer: 55 normal_skeletal_muscle: 18
(2, 3)			GSE17679	Bone_or_soft_tissue	câncer x normal - sem as cell line	62	2	cancer: 44 normal_skeletal_muscle: 18
(4, 1)			GSE45544	Bone_or_soft_tissue	câncer x normal (com amostra do câncer de estomago)	44	2	cancer: 22 normal: 22
(4, 2)			GSE45544	Bone_or_soft_tissue	ewing x normal (removendo amostra do câncer de estomago)	43	2	normal: 22 ewing: 21
(4, 3)			GSE45544	Bone_or_soft_tissue	ewing x outros	44	2	other: 23 ewing: 21
(7, 1)			GSE68776	Bone_or_soft_tissue	ewing x outros	74	2	other: 42 cancer_ewing_sarcoma: 32

Tabela 52 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 2/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(8, 1)			GSE85217	Brain	subtipos	763	12	Group4_gamma: 119 Group4_beta: 109 Group4_alpha: 98 SHH_delta: 76 Group3_alpha: 67 SHH_alpha: 65 WNT_alpha: 49 SHH_gamma: 47 Group3_gamma: 40 Group3_beta: 37 SHH_beta: 35 WNT_beta: 21
(8, 2)	x	x	GSE85217	Brain	grupos moleculares	763	4	Group4: 326 SHH: 223 Group3: 144 WNT: 70
(9, 1)			GSE37418	Brain	grupos moleculares	73	4	G4: 39 G3: 16 SHH: 10 WNT: 8
(10, 1)			GSE16515	Pancreatic	tumor x normal	52	2	tumor: 36 normal: 16
(11, 1)			GSE33447	Breast	câncer x normal	16	2	breast_cancer: 8 normal_breast: 8
(12, 1)	x		GSE59246	Breast	IBC, DCIS, normal	105	3	Invasive_breast_cancer_(IBC): 56 Ductal_carcinoma_in_situ_(DCIS): 46 Normal_breast_tissue: 3
(12, 2)	x		GSE59246	Breast	IBC x DCIS	102	2	Invasive_breast_cancer_(IBC): 56 Ductal_carcinoma_in_situ_(DCIS): 46
(13, 1)			GSE26910	Breast_prostate	normal x câncer	24	2	normal: 12 invasive_tumor: 12
(13, 2)	x	x	GSE26910	Breast_prostate	normal_breast x cancer_breast	12	2	Normal_breast_stroma: 6 Stroma_associated_to_breast_invasive_tumor: 6
(13, 3)			GSE26910	Breast_prostate	normal_prostate x cancer_prostate	12	2	Normal_prostate_stroma: 6 Stroma_associated_to_prostate_invasive_tumor: 6
(13, 4)			GSE26910	Breast_prostate	cancer_prostate x cancer_breast	12	2	Stroma_associated_to_prostate_invasive_tumor: 6 Stroma_associated_to_breast_invasive_tumor: 6
(13, 5)			GSE26910	Breast_prostate	normal_prostate x normal_breast	12	2	Normal_prostate_stroma: 6 Normal_breast_stroma: 6
(14, 1)			GSE57297	Breast	câncer x normal	32	2	breast_cancer: 25 normal_breast: 7
(15, 1)			GSE22820	Breast	tumor x normal	186	2	primary_breast_tumor: 176 normal_breast_tissue: 10
(16, 1)			GSE42568	Breast	câncer x normal	121	2	breast_cancer: 104 normal_breast: 17
(17, 1)			GSE26304	Breast	todas 4 classes	115	4	MIXED: 42 IDC: 36 DCIS: 31 NORMAL: 6
(17, 2)	x		GSE26304	Breast	todas 3 classes de câncer	109	3	MIXED: 42 IDC: 36 DCIS: 31
(17, 3)			GSE26304	Breast	câncer x normal	115	2	cancer: 109 normal: 6
(18, 1)	x	x	GSE70947	Breast	tumor x normal	296	2	normal: 148 tumor: 148

Tabela 53 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 3/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(19, 1)			GSE7904	Breast	todas 5 classes	62	5	Non-BLC: 21 Basal: 18 Normal_organelle: 12 Normal_breast: 7 BRCA1: 4
(19, 2)			GSE7904	Breast	3 tipos de câncer, normal	62	4	Non-BLC: 21 normal: 19 Basal: 18 BRCA1: 4
(19, 3)			GSE7904	Breast	3 tipos de câncer	43	3	Non-BLC: 21 Basal: 18 BRCA1: 4
(19, 4)	x		GSE7904	Breast	câncer x normal	62	2	cancer: 43 normal: 19
(20, 1)			GSE38959	Breast	câncer x normal	47	2	triple_negative_breast_cancer: 30 normal: 17
(21, 1)			GSE45827	Breast	todas 6 classes	155	6	Basal_Tumor_Sample: 41 Her2_Tumor_Sample: 30 Luminal_B_Tumor_Sample: 30 Luminal_A_Tumor_Sample: 29 different_type_origin_breast_tumor_cell_line: 14 Normal: 11
(21, 2)			GSE45827	Breast	todas 6 classes, normal+cell_line	155	5	Basal_Tumor_Sample: 41 Her2_Tumor_Sample: 30 Luminal_B_Tumor_Sample: 30 Luminal_A_Tumor_Sample: 29 Normal: 25
(21, 3)			GSE45827	Breast	todas 6 classea, sem cell line	141	5	Basal_Tumor_Sample: 41 Her2_Tumor_Sample: 30 Luminal_B_Tumor_Sample: 30 Luminal_A_Tumor_Sample: 29 Normal: 11
(21, 4)	x		GSE45827	Breast	tumor x normal	155	2	Tumor: 130 Normal: 25
(21, 5)			GSE45827	Breast	tumor x normal, sem cell line	141	2	Tumor: 130 Normal: 11
(22, 1)			GSE10797	Breast	todas 4 classes	66	4	cancer_epithelium: 28 cancer_stromal_cells: 28 normal_epithelium: 5 normal_stromal_cells: 5
(22, 2)			GSE10797	Breast	câncer x normal	66	2	cancer: 56 normal: 10
(22, 3)			GSE10797	Breast	cancer_epithelium x normal_epithelium	33	2	cancer_epithelium: 28 normal_epithelium: 5
(22, 4)			GSE10797	Breast	cancer_stromal_cells x normal_stromal_cells	33	2	cancer_stromal_cells: 28 normal_stromal_cells: 5
(22, 5)			GSE10797	Breast	cancer_epithelium x cancer_stromal_cells	56	2	cancer_epithelium: 28 cancer_stromal_cells: 28
(22, 6)			GSE10797	Breast	normal_epithelium x normal_stromal_cells	10	2	normal_epithelium: 5 normal_stromal_cells: 5
(23, 1)	x		GSE89116	Breast	todas 4 classes	33	4	LATE_TUMOR: 13 EARLY_TUMOR: 11 LATE_NORMAL: 5 EARLY_NORMAL: 4
(23, 2)			GSE89116	Breast	early_tumor x late_tumor	24	2	LATE_TUMOR: 13 EARLY_TUMOR: 11
(23, 3)			GSE89116	Breast	tumor x normal	33	2	tumor: 24 normal: 9
(24, 1)			GSE62232	Liver	carcinoma x normal	91	2	Hepatocellular_Carcinoma: 81 Normal: 10
(25, 1)			GSE50579	Liver	câncer x normal (com cell line junto ao normal)	80	2	hepatocellular_carcinoma_(HCC): 67 normal: 13
(25, 2)			GSE50579	Liver	câncer x normal (removendo as cell line)	77	2	hepatocellular_carcinoma_(HCC): 67 normal: 10

Tabela 54 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 4/9.

XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(26, 1)		GSE14520_U133A	Liver	HCC x normal	445	2	HCC: 225 Normal: 220
(27, 1)		GSE46408	Liver	câncer x normal	12	2	primary_hepatocellular_carcinomas_(HCC): 6 non-tumourous_liver_parenchyma: 6
(28, 1)		GSE22405	Liver	tumor x normal	48	2	non-tumor: 24 tumor: 24
(29, 1)		GSE57957	Liver	tumor x normal	78	2	hepatocellular_carcinoma_tumor: 39 adjacent_non-tumorous: 39
(30, 1)		GSE60502	Liver	carcinoma x normal	36	2	adjacent_non-tumorous_liver: 18 hepatocellular_carcinoma: 18
(31, 1)		GSE76427	Liver	tumor x normal	167	2	primary_hepatocellular_carcinoma_tumor: 115 adjacent_non-tumor_liver_tissue: 52
(32, 1)		GSE14520_U133_2	Liver	HCC x normal (juntando amostras dos doadores)	43	2	HCC: 22 normal: 21
(32, 2)		GSE14520_U133_2	Liver	HCC x normal (removendo amostras dos doadores)	41	2	HCC: 22 Normal: 19
(33, 1)		GSE42743	Throat	tumor x normal	103	2	oral_cavity_cancer: 74 normal: 29
(34, 1)		GSE59102	Throat	todas 3 classes	42	3	advanced: 15 beginning: 14 normal: 13
(34, 2)		GSE59102	Throat	advanced x beginning	29	2	advanced: 15 beginning: 14
(34, 3)		GSE59102	Throat	câncer x normal	42	2	cancer: 29 normal: 13
(35, 1)		GSE12452	Throat	carcinoma x normal	41	2	nasopharyngeal_carcinoma: 31 normal: 10
(36, 1)		GSE53819	Throat	carcinoma x normal	36	2	Nasopharyngeal_carcinoma_primary_tumor: 18 normal: 18
(37, 1)		GSE33615	Leukemia	ATL x normal	73	2	ATL: 52 Normal: 21
(38, 1)		GSE71935	Leukemia	leucemia x normal	47	2	Juvenile_myelomonocytic_leukemia_(JMML): 38 healthy: 9
(39, 1)		GSE22529_U133B	Leukemia	CLL x normal	52	2	CLL: 41 normal: 11
(40, 1)		GSE14317	Leukemia	leucemia x normal (com amostras tratadas, com progressão de doença linfomática)	25	2	leukemia: 18 normal: 7
(40, 2)		GSE14317	Leukemia	leucemia x normal (com amostras tratadas, sem progressão de doença linfomática)	22	2	leukemia: 15 normal: 7
(40, 3)		GSE14317	Leukemia	leucemia x normal (sem amostras tratadas, com progressão de doença linfomática)	21	2	leukemia: 14 normal: 7
(40, 4)		GSE14317	Leukemia	leucemia x normal (sem amostras tratadas, sem progressão de doença linfomática)	18	2	acute_leukemia: 11 normal: 7
(40, 5)		GSE14317	Leukemia	leucemia x outros	26	2	leukemia: 18 other: 8
(41, 1)		GSE63270	Leukemia	AML x normal	104	2	Bone_marrow_of_AML_patient: 62 Bone_marrow_of_healthy_donor: 42

Tabela 55 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 5/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(42, 1)	x	x	GSE28497	Leukemia	8 subtipos de ALL, normal	288	9	7_B-with-miscellaneous-karyotype: 59 3_ETV6_RUNX1: 56 1_Hyperdiploid: 52 8_T-ALL: 45 2_TCF3-PBX1: 22 4_MLL: 18 5_Ph: 16 6_Hypo: 16 9_CD10CD19: 4
(42, 2)			GSE28497	Leukemia	8 subtipos de ALL	284	8	7_B-with-miscellaneous-karyotype: 59 3_ETV6_RUNX1: 56 1_Hyperdiploid: 52 8_T-ALL: 45 2_TCF3-PBX1: 22 4_MLL: 18 5_Ph: 16 6_Hypo: 16
(42, 3)			GSE28497	Leukemia	ALL x normal	288	2	ALL: 284 normal: 4
(43, 1)	x		GSE71449	Leukemia	leucemia x normal	51	2	Patient: 44 Healthy: 7
(44, 1)			GSE22529_U133A	Leukemia	CLL x normal	52	2	CLL: 41 normal: 11
(45, 1)			GSE9476	Leukemia	todas 5 classes	64	5	AML: 26 Normal_Bone_Marrow: 10 Normal_Peripheral_Blood: 10 Normal_Peripheral_Blood_Stem_Cell_CD34: 10 Normal_Bone_Marrow_CD34: 8
(45, 2)			GSE9476	Leukemia	AML, normal_bone_marrow, normal_peripheral_blood	64	3	AML: 26 Normal_Peripheral_Blood: 20 Normal_Bone_Marrow: 18
(45, 3)			GSE9476	Leukemia	AML x normal	64	2	normal: 38 AML: 26
(46, 1)			GSE8511	Prostate	todas 3 classes	41	3	benign: 16 metastatic_cancer: 13 local_cancer: 12
(46, 2)			GSE8511	Prostate	local x metastático	25	2	metastatic_cancer: 13 local_cancer: 12
(46, 3)			GSE8511	Prostate	câncer x benigno	41	2	cancer: 25 benign: 16
(47, 1)			GSE6919_U95C	Prostate	tumor primario, normal de doador sem câncer, normal adj, metástase	165	4	primary_prostate_tumor: 65 normal_prostate_tissue_adjacent_to_tumor: 58 metastases: 25 normal_prostate_tissue_free_of_any_pathological_alteration: 17
(47, 2)			GSE6919_U95C	Prostate	tumor primario, normal, metástase	165	3	normal: 75 primary_prostate_tumor: 65 metastases: 25
(47, 3)			GSE6919_U95C	Prostate	tumor+metástase, normal	165	2	tumor: 90 normal: 75
(47, 4)	x		GSE6919_U95C	Prostate	tumor, normal	140	2	normal: 75 primary_prostate_tumor: 65
(48, 1)	x		GSE11682	Prostate	câncer x normal	34	2	grade_3_reactive_stroma_prostate_cancer_tissues: 17 normal_peripheral_zone_prostate_stroma: 17
(49, 1)			GSE46602	Prostate	câncer x normal	50	2	prostate_tumor: 36 benign_prostate_glands: 14
(50, 1)			GSE38241	Prostate	metástase x normal	39	2	Normal_Prostate: 21 Prostate_cancer_metastasis: 18
(51, 1)			GSE6919_U95B	Prostate	tumor primario, normal de doador sem câncer, normal adj, metástase	168	4	primary_prostate_tumor: 66 normal_prostate_tissue_adjacent_to_tumor: 60 metastases: 25 normal_prostate_tissue_free_of_any_pathological_alteration: 17
(51, 2)			GSE6919_U95B	Prostate	tumor primario, normal, metástase	168	3	normal: 77 primary_prostate_tumor: 66 metastases: 25
(51, 3)	x		GSE6919_U95B	Prostate	tumor+metástase, normal	168	2	tumor: 91 normal: 77
(51, 4)	x	x	GSE6919_U95B	Prostate	tumor x normal	143	2	normal: 77 primary_prostate_tumor: 66

Tabela 56 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 6/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(52, 1)			GSE55945	Prostate	maligno x benigno	21	2	Malignant_prostate_tissue: 13 Benign_prostate_tissue: 8
(53, 1)			GSE6919_U95Av2	Prostate	tumor primario, normal de doador sem câncer, normal adj, metástase	171	4	primary_prostate_tumor: 65 normal_prostate_tissue_adjacent_to_tumor: 63 metastases: 25 normal_prostate_tissue_free_of_any_pathological_alteration: 18
(53, 2)			GSE6919_U95Av2	Prostate	tumor primario, normal, metástase	171	3	normal: 81 primary_prostate_tumor: 65 metastases: 25
(53, 3)	x		GSE6919_U95Av2	Prostate	tumor+metástase, normal	171	2	tumor: 90 normal: 81
(53, 4)			GSE6919_U95Av2	Prostate	tumor, normal	146	2	normal: 81 primary_prostate_tumor: 65
(54, 1)			GSE12470	Ovary	todas 3 classes	53	3	Advanced_serous_ovarian_cancer: 35 Peritoneum_normal: 10 Early_serous_ovarian_cancer: 8
(54, 2)			GSE12470	Ovary	early x advanced	43	2	Advanced_serous_ovarian_cancer: 35 Early_serous_ovarian_cancer: 8
(54, 3)			GSE12470	Ovary	câncer x normal	53	2	cancer: 43 normal: 10
(55, 1)			GSE16708	Ovary	todas 3 classes	33	3	serous_ovarian_adenocarcinoma: 17 normal_cell_line: 9 adenocarcinoma_serous_cell_line: 7
(55, 2)			GSE16708	Ovary	adenocarcinoma x normal	33	2	adenocarcinoma: 24 normal_cell_line: 9
(55, 3)			GSE16708	Ovary	adenocarcinoma_cell_line x normal_cell_line	16	2	normal_cell_line: 9 adenocarcinoma_serous_cell_line: 7
(56, 1)			GSE16570	Ovary	todas 3 classes	16	3	tumor_clear_cell_line_confluent: 7 normal_cell_line: 6 tumor_clear_cell_line_subconfluent: 3
(56, 2)			GSE16570	Ovary	tumor x normal	16	2	tumor_clear_cell_line: 10 normal_cell_line: 6
(57, 1)			GSE6008	Ovary	4 tipos de tumor, normal	103	5	Serous: 41 Endometrioid: 37 Mucinous: 13 Clear_Cell: 8 Normal: 4
(57, 2)			GSE6008	Ovary	4 tipos de tumor	99	4	Serous: 41 Endometrioid: 37 Mucinous: 13 Clear_Cell: 8
(57, 3)			GSE6008	Ovary	tumor x normal	103	2	ovary_cancer: 99 Normal: 4
(58, 1)			GSE50161	Brain	4 tipos de tumor, normal	130	5	brain_tumor_(ependymoma): 46 brain_tumor_(glioblastoma): 34 brain_tumor_(medulloblastoma): 22 brain_tumor_(pilocytic_astrocytoma): 15 normal: 13
(58, 2)			GSE50161	Brain	4 tipos de tumor	117	4	brain_tumor_(ependymoma): 46 brain_tumor_(glioblastoma): 34 brain_tumor_(medulloblastoma): 22 brain_tumor_(pilocytic_astrocytoma): 15
(58, 3)			GSE50161	Brain	tumor x normal	130	2	brain_tumor: 117 normal: 13
(59, 1)			GSE15824	Brain	glioblastoma (juntando secundário), glioblastoma_cell_line, astrocitoma, oligodendrioglioma, normal	45	5	glioblastoma: 15 glioblastoma_cell_line: 10 astrocytoma: 8 oligodendrioglioma: 7 normal: 5
(59, 2)			GSE15824	Brain	glioblastoma (juntando secundário, cell line), astrocitoma, oligodendrioglioma, normal	45	4	glioblastoma: 25 astrocytoma: 8 oligodendrioglioma: 7 normal: 5
(59, 3)			GSE15824	Brain	glioblastoma (sem secundário), glioblastoma_cell_line, astrocitoma, oligodendrioglioma, normal	42	5	glioblastoma: 12 glioblastoma_cell_line: 10 astrocytoma: 8 oligodendrioglioma: 7 normal: 5

Tabela 57 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 7/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(59, 4)	x	x	GSE15824	Brain	glioblastoma (sem secundário, com cell line), astrocitoma, oligodendrioglioma, normal	42	4	glioblastoma: 22 astrocytoma: 8 oligodendrioglioma: 7 normal: 5
(59, 5)			GSE15824	Brain	glioblastoma (sem secundário, sem cell line), astrocitoma, oligodendrioglioma, normal	32	4	glioblastoma: 12 astrocytoma: 8 oligodendrioglioma: 7 normal: 5
(59, 6)			GSE15824	Brain	glioblastoma (juntando secundário), glioblastoma_cell_line, astrocitoma, oligodendrioglioma	40	4	glioblastoma: 15 glioblastoma_cell_line: 10 astrocytoma: 8 oligodendrioglioma: 7
(59, 7)			GSE15824	Brain	glioblastoma (juntando secundário, cell line), astrocitoma, oligodendrioglioma	40	3	glioblastoma: 25 astrocytoma: 8 oligodendrioglioma: 7
(59, 8)			GSE15824	Brain	glioblastoma (sem secundário), glioblastoma_cell_line, astrocitoma, oligodendrioglioma	37	4	glioblastoma: 12 glioblastoma_cell_line: 10 astrocytoma: 8 oligodendrioglioma: 7
(59, 9)	x		GSE15824	Brain	glioblastoma (sem secundário, com cell line), astrocitoma, oligodendrioglioma	37	3	glioblastoma: 22 astrocytoma: 8 oligodendrioglioma: 7
(59, 10)	x		GSE15824	Brain	glioblastoma (sem secundário, sem cell line), astrocitoma, oligodendrioglioma	27	3	glioblastoma: 12 astrocytoma: 8 oligodendrioglioma: 7
(59, 11)			GSE15824	Brain	câncer x normal (com cell line)	45	2	cancer: 40 normal: 5
(59, 12)			GSE15824	Brain	câncer x normal (sem cell line)	35	2	cancer: 30 normal: 5
(60, 1)			GSE31189	Bladder	câncer x normal	92	2	cancer_urothelial: 52 normal_urothelial: 40
(61, 1)			GSE40355	Bladder	todas 3 classes	24	3	nonmaligant_bladder: 8 low-grade_papillary_urothelial_carcinoma: 8 high-grade_papillary_urothelial_carcinoma: 8
(61, 2)			GSE40355	Bladder	carcinoma x não-maligno	24	2	carcinoma: 16 nonmaligant_bladder: 8
(62, 1)			GSE7670	Lung	adenocarcinoma, large_cell_carcinoma, normal (juntando cell lines)	66	3	cancer_adenocarcinoma: 32 normal: 31 cancer_large_cell_carcinoma: 3
(62, 2)	x	x	GSE7670	Lung	adenocarcinoma x normal adj adenocarcinoma (sem cell line)	54	2	normal_adjacent_adenocarcinoma: 27 cancer_adenocarcinoma: 27
(62, 3)			GSE7670	Lung	adenocarcinoma x normal adenocarcinoma (juntando cell line)	62	2	adenocarcinoma: 32 normal: 30
(62, 4)			GSE7670	Lung	adenocarcinoma x outros (juntando cell line)	66	2	other: 34 adenocarcinoma: 32
(63, 1)			GSE27262	Lung	adenocarcinoma x normal	50	2	normal_adjacent: 25 tumor_adenocarcinoma: 25
(64, 1)			GSE19804	Lung	tumor x normal	120	2	tumor_primary: 60 normal_adjacent: 60
(65, 1)			GSE74706	Lung	NSCLC x normal	36	2	normal: 18 NSCLC: 18
(66, 1)			GSE63459	Lung	adenocarcinoma x normal	65	2	adenocarcinoma: 33 nontumor_adjacent: 32
(67, 1)			GSE18842	Lung	tumor x normal	91	2	tumor: 46 normal_adjacent: 45
(68, 1)			GSE66270	Renal	carcinoma x normal	28	2	clear_cell_carcinoma: 14 normal_adjacent: 14

Tabela 58 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 8/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(69, 1)			GSE6344_U133A	Renal	tumor x normal	20	2	Normal: 10 Tumor: 10
(70, 1)			GSE53757	Renal	4 estagios câncer, 4 estagios normal	144	8	Stage_1_ccRCC: 24 normal_match_to_Stage_1_ccRCC: 24 Stage_2_ccRCC: 19 normal_match_to_Stage_2_ccRCC: 19 Stage_4_ccRCC: 15 normal_match_to_Stage_4_ccRCC: 15 Stage_3_ccRCC: 14 normal_match_to_Stage_3_ccRCC: 14
(70, 2)			GSE53757	Renal	4 estagios, normal	144	5	normal: 72 Stage_1_ccRCC: 24 Stage_2_ccRCC: 19 Stage_4_ccRCC: 15 Stage_3_ccRCC: 14
(70, 3)			GSE53757	Renal	câncer x normal	144	2	ccRCC: 72 normal: 72
(71, 1)	x	x	GSE6344_U133B	Renal	tumor x normal	20	2	Normal: 10 Tumor: 10
(72, 1)			GSE79973	Gastric	adenocarcinoma x normal	20	2	gastric_adenocarcinoma_tumor: 10 normal_adjacent: 10
(73, 1)			GSE19826	Gastric	câncer x normal	27	2	normal: 15 cancer: 12
(73, 2)			GSE19826	Gastric	câncer x normal (apenas com normal_adjacent)	24	2	normal_adjacent: 12 cancer: 12
(74, 1)			GSE44861	Colorectal	tumor x normal	111	2	tumor: 56 normal_adjacent: 55
(75, 1)			GSE75548	Colorectal	câncer x normal	12	2	rectal_cancer: 6 adjacent_normal_tissue: 6
(76, 1)			GSE41328	Colorectal	adenocarcinoma x normal	20	2	normal_adjacent: 10 adenocarcinoma: 10
(77, 1)	x		GSE77953	Colorectal	todas 5 classes	58	5	Adenoma: 17 Carcinoma: 17 Metastasis_in_other_tissue: 11 Normal_crypt_epithelium: 7 Normal_surface_epithelium: 6
(77, 2)			GSE77953	Colorectal	adenoma, carcinoma, metástase, normal	58	4	Adenoma: 17 Carcinoma: 17 Normal: 13 Metastasis_in_other_tissue: 11
(77, 3)			GSE77953	Colorectal	adenoma, carcinoma, normal	47	3	Adenoma: 17 Carcinoma: 17 Normal: 13
(77, 4)			GSE77953	Colorectal	câncer x normal (incluindo amostras de metástase)	58	2	cancer: 45 normal: 13
(77, 5)	x	x	GSE77953	Colorectal	câncer x normal (removendo amostras de metástase)	47	2	cancer: 34 normal: 13
(77, 6)			GSE77953	Colorectal	câncer x metástase x normal	58	3	cancer: 34 normal: 13 Metastasis_in_other_tissue: 11
(78, 1)			GSE25070	Colorectal	adenocarcinoma x normal	52	2	adenocarcinoma: 26 normal_adjacent: 26
(79, 1)			GSE8671	Colorectal	adenoma x normal	64	2	normal: 32 adenoma: 32
(80, 1)			GSE41657	Colorectal	todas 4 classes	88	4	High-grade_dysplasia: 30 Adenocarcinoma: 25 Low-grade_dysplasia: 21 Normal_epithelium: 12
(80, 2)			GSE41657	Colorectal	adenocarcinoma, adenoma (high e low grade), normal	88	3	Adenoma: 51 Adenocarcinoma: 25 Normal_epithelium: 12
(80, 3)			GSE41657	Colorectal	câncer x normal	88	2	Cancer: 76 Normal: 12

Tabela 59 – Problemas (divisões de classes) e datasets de expressão gênica (GSEs) empregados - Parte 9/9.

	XGBoost	NNs	GSE	Tecido	Problema	Nr de Amostras	Nr de Classes	Nr de Amostras por Classe
(81, 1)	x		GSE32323	Colorectal	câncer, normal, cancer_cell_line_tratado, cancer_cell_line_nao_tratado	44	4	normal: 17 cancer: 17 cancer_cell_line_treated: 5 cancer_cell_line: 5
(81, 2)			GSE32323	Colorectal	câncer x normal (com cell_line)	44	2	cancer: 27 normal: 17
(81, 3)			GSE32323	Colorectal	câncer x normal (sem cell_line)	34	2	normal: 17 cancer: 17
(82, 1)			GSE21510	Colorectal	câncer estagios (sem amostras homogenizadas)	104	7	cancer_LCM_3b: 24 cancer_LCM_4: 20 cancer_LCM_2a: 20 cancer_LCM_2b: 17 cancer_LCM_1: 13 cancer_LCM_3c: 6 cancer_LCM_3a: 4
(82, 2)			GSE21510	Colorectal	câncer estagios (juntando LCM e homogenização)	123	7	cancer_3b: 27 cancer_2b: 23 cancer_4: 23 cancer_2a: 23 cancer_1: 15 cancer_3c: 8 cancer_3a: 4
(82, 3)			GSE21510	Colorectal	câncer homogenizado x normal homogenizado	44	2	normal: 25 cancer: 19
(82, 4)	x	x	GSE21510	Colorectal	câncer (juntando LCM e homogenizado) x normal homogenizado	148	2	cancer: 123 normal: 25
(82, 5)			GSE21510	Colorectal	estágios de câncer (juntando LCM e homogenizado), normal homogenizado	148	8	cancer_3b: 27 normal: 25 cancer_2b: 23 cancer_4: 23 cancer_2a: 23 cancer_1: 15 cancer_3c: 8 cancer_3a: 4
(83, 1)	x		GSE44076	Colorectal	adenocarcinoma 2 estagios, normal adj, normal	246	4	normal_paired: 98 adenocarcinoma_IIA: 90 normal_healthy: 50 adenocarcinoma_IIB: 8
(83, 2)			GSE44076	Colorectal	adenocarcinoma, normal adj, normal	246	3	normal_paired: 98 adenocarcinoma: 98 normal_healthy: 50
(83, 3)			GSE44076	Colorectal	adenocarcinoma 2 estagios, normal	246	3	normal: 148 adenocarcinoma_IIA: 90 adenocarcinoma_IIB: 8
(83, 4)			GSE44076	Colorectal	adenocarcinoma x normal (juntando adj e normal)	246	2	normal: 148 adenocarcinoma: 98