UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Departamento de Sistemas de Computação

Variations of Deep Learning for Cancer Diagnosis Using Gene Expression Data

Henrique de Almeida Machado da Silveira

São Carlos – SP

Variations of Deep Learning for Cancer Diagnosis Using Gene Expression Data

Henrique de Almeida Machado da Silveira

Orientador: **Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho**

Monografia referente ao projeto de conclusão de curso dentro do espaço da disciplina SSC0670 - Projeto de Formatura I do Departamento de Sistemas de Computação do Instituto de Ciências Matemáticas e de Computação – ICMC-USP, para obtenção do título de Engenheiro de Computação.

Área de Concentração: Inteligência Computacional, Aprendizado de Máquina

USP – São Carlos
Novembro de 2017

Silveira, Henrique de Almeida Machado da
Variations of Deep Learning for Cancer Diagnosis Using Gene Expression Data / Henrique de Almeida Machado da Silveira. - São Carlos - SP, 2017.
65 p.; 29,7 cm.

Orientador: André Carlos Ponce de Leon Ferreira de Carvalho.
Monografia (Graduação) - Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos - SP, 2017.

1. Aprendizado de Máquina. 2. Aprendizado Profundo. 3. Dados de Expressão Gênica. 4. Classificação. 5. Bioinformática. I. Carvalho, André Carlos Ponce de Leon Ferreira de. II. Instituto de Ciências Matemáticas e de Computação (ICMC/USP). III. Título.

Este trabalho é dedicado à minha família, especialmente
minha mãe e meu pai, que devotaram a totalidade de suas vidas
e nunca mediram esforços na criação de seus filhos.

AGRADECIMENTOS

Primeiramente, agradeço à minha família por todo o amor e confiança depositados em mim. Sem a estrutura dada por minha família não teria sido possível concluir este curso. Obrigado por me apoiarem em todos os momentos e tornarem possível meu sonho de ser um Engenheiro de Computação.

Agradeço a todos os excelentes professores que tive o privilégio de conhecer ao longo da graduação, que não só contribuíram para minha formação e conhecimento acadêmico, mas também para meu amadurecimento e formação pessoal, especialmente os professores e as professoras Maria do Carmo Carbinatto, Irene Onnis, Simone Souza, Paulo Souza, Everaldo Bonotto, Lucas Bueno Oliveira (do ICMC), Evandro Rodrigues (da EESC) e Atílio Tomazini (IFSC). Agradeço especialmente também o meu professor e orientador André Carvalho, que me recebeu de braços abertos e sempre foi muito atencioso e prestativo, me incentivando e motivando e com quem muito aprendi ao longo deste trabalho de conclusão de curso.

A todos os funcionários da USP: aos funcionários das cantinas do campus, dos restaurantes, das secretarias da graduação do ICMC e da EESC, das bibliotecas, aos seguranças e pessoal da limpeza e tantos outros que tornaram possível o dia a dia de todos os estudantes.

A todos os amigos do grupo de pesquisa do professor André, em especial Edésio Alcobaça, Igor Martinelli, Denilson Marques e Henrique Cintra, que muito me ajudaram com discussões e conselhos valiosos acerca dos temas discutidos neste trabalho.

E finalmente agradeço a todos meus amigos e colegas que se tornaram minha família em São Carlos, especialmente Adriano, Cassiano, Elisa, Giuliano, Henrique Grando, Igor Mendes, Igor Ramon, Isabella, Jéssica, Jessika, Lucas Rovere, Lucas Tomazela, Luiza, Luiz, Marcello, Marvin, Moisés, Pedro e Victor. Agradeço ao Esquadrão da Zuera por todos os inúmeros momentos compartilhados, tanto em noites jogando e comendo pizza quanto em vésperas de prova. Sem vocês essa longa jornada não teria o mesmo significado.

“ If money is your hope for independence you will never have it.
The only real security that a man will have in this world is a
reserve of knowledge, experience, and ability.”
(Henry Ford)

RESUMO

O desenvolvimento da área de inteligência artificial e, mais especificamente, da área de aprendizado de máquina, tem contribuído muito para diversos setores da ciência, como a saúde. O aprendizado de máquina permite a manipulação e interpretação de dados que outrora fora inviável (devido, por exemplo, à enorme quantidade de dados a serem analisados). Este trabalho terá como objetivo a investigação do potencial de algoritmos de aprendizado de máquina, em especial de aprendizado profundo, na análise de dados de expressão gênica para diagnóstico de câncer, focando na predição de classes. Para isso, serão realizados experimentos com três arquiteturas diferentes de aprendizado profundo, utilizando dados de expressão gênica de domínio público.

Palavras-chave: Aprendizado de Máquina, Aprendizado Profundo, Dados de Expressão Gênica, Classificação, Bioinformática.

ABSTRACT

The development of artificial intelligence and, more specifically, the area of machine learning, has contributed significantly to the advances of other fields of science, such as health. Machine learning aids in the manipulation and interpretation of data, enabling discoveries and insights that were previously unfeasible (for example, due to very large amounts of data). This project aims to investigate the potential of different machine learning algorithms, especially deep learning algorithms, in analyzing gene expression data for cancer diagnosis, focusing on class prediction. To do that, experiments will be conducted with three different deep learning architectures, using public domain gene expression datasets.

Key-words: Machine Learning, Deep Learning, Gene Expression Data, Classification, Bioinformatics.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM ..... Aprendizado de Máquina
AP ..... Aprendizado Profundo
CNN ..... *Convolutional Neural Network*
DNA ..... Ácido Desoxirribonucleico
GPU ..... *Graphics Processing Unit*
miRNA-Seq Illumina HiSeq 2000 miRNA Sequencing
MLP ..... *Multilayer Perceptron*
RNA ..... Ácido Ribonucleico
RNA-Seq . *RNA Sequencing*
RNA-Seqv2 Illumina HiSeq 2000 RNA Sequencing Version 2
SDAE .... *Stacked Denoising Autoencoder*
SVM ..... *Support Vector Machine*
TCGA .... *The Cancer Genome Atlas*

SUMÁRIO

Capítulo 1

INTRODUÇÃO

1.1 Contextualização e Motivação

A área da ciência da computação denominada inteligência artificial tem contribuído para que carros autônomos, robôs inteligentes e sistemas de reconhecimento facial e de voz, por exemplo, façam parte do cotidiano da sociedade moderna. Embora essa área seja de extrema relevância na atualidade, sendo muito explorada tanto na indústria quanto na academia, a inteligência artificial passou por um longo período em que foi preterida e desacreditada. O ramo do aprendizado de máquina e, mais especificamente, algoritmos de aprendizado profundo (*deep learning*), podem ser considerados os principais vetores que auxiliaram o campo da inteligência artificial na retomada de sua relevância nos âmbitos acadêmico e científico (BENGIO, 2016).

Outras áreas do conhecimento se beneficiam muito com o desenvolvimento da inteligência artificial. Por exemplo, no campo da medicina, a inteligência artificial tem sido de grande auxílio no diagnóstico de doenças, auxiliando profissionais de saúde; na análise de imagens e sinais de exames médicos (BENGIO, 2016); e também no processamento de dados de expressão gênica para diagnóstico de câncer (SAJDA, 2006). Este trabalho terá seu foco justamente nesse último campo: será analisado o potencial de algoritmos de aprendizado profundo na análise de dados de expressão gênica para diagnóstico de câncer.

A relevância da união entre aprendizado de máquina e, mais especificamente, aprendizado profundo, e o diagnóstico de câncer vem do fato de que essas técnicas de inteligência artificial permitem que pesquisadores e profissionais de saúde entendam melhor a correlação entre perfis de expressão gênica e estágios de doenças como câncer ou ainda os diferentes estágios de desenvolvimento de uma célula (TAN; GILBERT, 2003). E, principalmente na atualidade, em que a alta disponibilidade e rápida taxa de geração de dados contribuem para uma vasta base de dados médicos, percebe-se, mais do que nunca, a importância e necessidade de técnicas de aprendizado de máquina (SAJDA, 2006): o uso de aprendizado de máquina viabiliza a análise de uma grande quantidade de dados gerados por novas técnicas e tecnologias de sequenciamento de genes, como aquelas baseadas em RNA-Seq, possibilitando melhores compreensão, diagnóstico e cura para câncer e trazendo grandes benefícios sociais e econômicos para a sociedade.

1.2 Objetivos

Este projeto tem como principal objetivo averiguar o potencial de algoritmos de aprendizado profundo na análise de dados biológicos. Mais especificamente, serão utilizados dados de expressão gênica para diagnóstico de câncer obtidos com tecnologias RNA-Seq, o que também contribui para a disseminação do conhecimento científico referente à aplicação de técnicas de aprendizado de máquina para análise de dados biológicos.

Com relação a objetivos específicos, pode-se citar:

1. Realizar uma revisão bibliográfica sobre aprendizado de máquina, redes neurais artificiais e aprendizado profundo no contexto de classificação de dados e sobre dados de expressão gênica;
2. Coletar, investigar e interpretar dados de expressão gênica de domínio público obtidos por tecnologia RNA-Seqv2 e miRNA-Seq;
3. Comparar diversos algoritmos de aprendizado de máquina (especialmente diferentes arquiteturas de aprendizado profundo) na análise de dados de expressão gênica;
4. Preparar um ambiente computacional para a utilização das diferentes arquiteturas e experimentos;

1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

- O **Capítulo 2** apresenta a revisão bibliográfica dos tópicos abordados neste trabalho, incluindo conceitos básicos de aprendizado de máquina, técnicas de aprendizado profundo e apresentando uma breve introdução à tecnologia RNA-Seq.
- O **Capítulo 3** apresenta o problema de classificação de dados de expressão gênica utilizando as técnicas de aprendizado de máquina investigadas neste trabalho, além dos passos adotados na elaboração dos experimentos e dos resultados obtidos.
- O **Capítulo 4** apresenta conclusões finais tanto em relação ao trabalho em si quanto sobre a relação entre o trabalho e o curso de Engenharia de Computação.

Capítulo 2

REVISÃO BIBLIOGRÁFICA

2.1 Considerações Iniciais

Neste capítulo serão apresentados os conceitos e tecnologias utilizados no desenvolvimento deste trabalho: serão abordadas definições e conceitos relacionados a aprendizado de máquina, redes neurais artificiais, aprendizado profundo e possíveis arquiteturas e também tecnologia RNA-Seq. No fim do capítulo será apresentada uma sucinta discussão a respeito de outros trabalhos relacionados a este projeto.

2.2 Conceitos e Definições

Nesta seção, serão apresentados os conceitos e definições relevantes no contexto deste trabalho.

2.2.1 Aprendizado de Máquina

O Aprendizado de Máquina (AM) pode ser definido como o "processo de indução de uma hipótese (ou aproximação de uma função) a partir da experiência passada"(GAMA *et al.*, 2011). Tendo como base as áreas de Inteligência Artificial e Probabilidade e Estatística e envolvendo elementos de Teoria da Computação, Neurocência e Teoria da Informação, o AM é um ramo da computação que tem tido notável crescimento recentemente e tem sido empregado em diversos tipos de aplicação, como reconhecimento de imagens e de fala, condução autônoma de automóveis e diagnóstico de câncer por meio da análise de dados de expressão gênica (GAMA *et al.*, 2011).

Em AM, o aprendizado por parte dos algoritmos ocorre após ser feito um treinamento com dados que representam uma coleção de exemplos de um determinado problema. Esse treinamento baseia-se no princípio de indução: são obtidas conclusões genéricas sobre o problema a partir do conjunto de dados de exemplos, generalizando os fatos que podem ser aprendidos a partir dos exemplos (GAMA *et al.*, 2011). Entretanto, o processo indutivo pode levar a conclusões erradas; não há garantias de que a indução leve a conclusões sempre corretas (MITCHELL, 1997). No processo indutivo, uma maior representatividade do conceito sendo analisado nos dados leva a uma modelagem mais confiável.(SOUZA, 2010).

Os algoritmos de AM podem ser classificados de acordo com o paradigma utilizado para lidar com a tarefa em questão. Essas tarefas podem ser divididas em duas classes (GAMA *et al.*, 2011):

- Tarefas Preditivas: têm como objetivo criar um modelo a partir do conjunto de dados de treinamento que será utilizado para predizer um rótulo ou valor que descreva um novo exemplo. No caso de valores discretos, trata-se de uma tarefa de classificação. Já no caso de valores contínuos, temos uma tarefa de regresão. O aprendizado nesse caso é chamado de supervisionado, já que os exemplos do conjunto de treinamento são rotulados em classes (FLACH, 2012).
- Tarefas Descritivas: têm como objetivo a exploração e descrição de um conjunto de dados. Por exemplo, em uma tarefa de agrupamento, a meta é agrupar exemplos do conjunto de dados que tenham características em comum. Associação e sumarização são outros tipos de tarefas descritivas. Esse tipo de tarefa faz uso do aprendizado classificado como não supervisionado, em que os dados de treinamento não são rotulados (FLACH, 2012).

Este trabalho terá como foco tarefas preditivas de classificação, utilizando o modelo conhecido como Redes Neurais Artificiais (que será brevemente explicado adiante) para criar arquiteturas de aprendizado profundo. Os desempenhos dessas arquiteturas na análise dos dados serão comparados entre si e também com o desempenho do modelo conhecido como Árvore de Decisão. Entretanto, também será feito o uso de aprendizado não supervisionado em uma das etapas do trabalho, em que será utilizada uma Rede Neural Artificial para reduzir a dimensionalidade dos dados de expressão gênica.

2.2.2 Árvores de Decisão

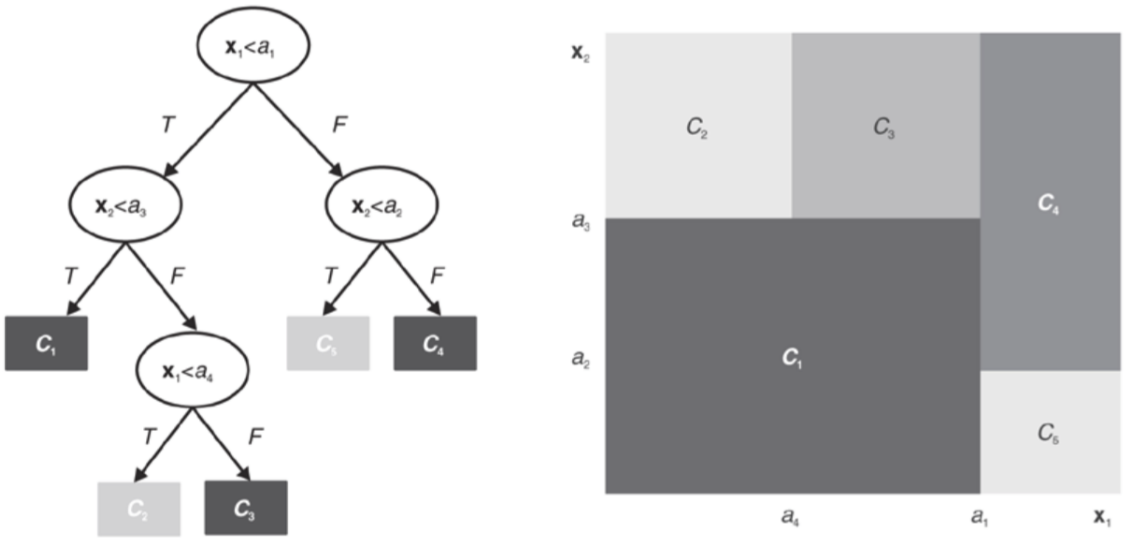
Segundo Gama *et al.* (2011), "uma árvore de decisão usa a estratégia dividir para conquistar para resolver um problema de decisão". Isso significa que a abordagem utilizada por Árvores de Decisão consiste em dividir um problema relativamente complexo em problemas menores, aplicando recursivamente essa estratégia até obter problemas triviais ou de fácil solução. CART (*Classification and Regression Trees*) (BREIMAN *et al.*, 1984) é um importante algoritmo baseado em árvores de decisão e será utilizado neste trabalho.

Uma árvore de decisão é constituída de um grafo acíclico e direcionado, possuindo nós intermediários, denominados nós de decisão, e nós folha. Os nós de decisão contém testes condicionais que avaliam os valores dos atributos dos dados. Esses testes podem envolver um único atributo (chamados de testes univariados) ou múltiplos atributos. Já os nós folha apresentam uma função, que geralmente é a constante que minimiza a função de custo do algoritmo e definem o valor da variável alvo do problema. Cada nó da árvore representa uma região no espaço definido pelos atributos dos dados problema. As regiões definidas pelos nós

folha são mutuamente excludentes, isto é, a intersecção dessas regiões é nula. (GAMA *et al.*, 2011).

A Figura 1 apresenta a representação gráfica de uma árvore de decisão e também o espaço definido pelos atributos dos dados do problema, x_1 e x_2. Os nós de decisão da árvore apresentam testes univariados, em que é realizado um teste com o valor de um atributo x_i (que pode assumir valores a_i) do exemplo sendo analisado. Cada teste é realizado levando-se em consideração o atributo em questão e, se o teste é verdadeiro, a sub-árvore à esquerda do nó atual é escolhida (indicado no diagrama pela letra T, de *true*). Se o teste é falso, a sub-árvore à direita é considerada. Os nós folha, por sua vez, representam regiões mutuamente excludentes c_i, que designam o valor que a variável alvo pode assumir em cada situação.

Figura 1 – Árvore de decisão e espaço definido pelos atributos dos dados do problema



Fonte: Gama *et al.* (2011).

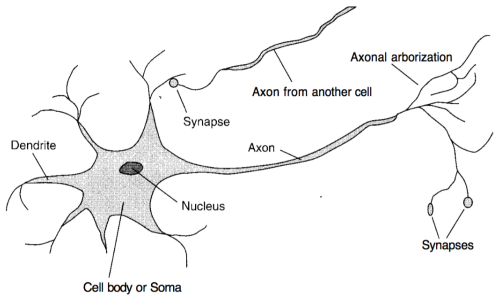
Os testes realizados em cada nó de decisão dividem o espaço de possibilidades. Essas divisões são feitas com base em regras de divisão, guiadas por métricas que indicam quão bem um atributo pode discriminar as classes. Para o algoritmo CART, as regras de divisão são baseadas no índice de Gini (GAMA *et al.*, 2011).

2.2.3 Redes Neurais Artificiais

Redes Neurais Artificiais representam uma importante técnica de aprendizado de máquina utilizada em tarefas preditivas de classificação. Segundo Gama *et al.* (2011), essa técnica tem como inspiração o modelo biológico do sistema nervoso, levando em consideração sua estrutura e funcionamento para a criação de um modelo matemático que possa reproduzir a capacidade de aprendizado do cérebro humano ao adquirir conhecimento. As Redes Neurais Artificiais são formadas por pequenas unidades de processamento denominadas neurônios

artificiais (GAMA *et al.*, 2011), que são responsáveis por computar funções matemáticas e estão densamente conectadas. O neurônio artificial é inspirado no neurônio biológico encontrado nos seres humanos, apresentado na Figura 2. O funcionamento desse tipo de célula pode ser explicado sucintamente da seguinte forma: o neurônio é formado por um corpo que tem pequenas ramificações denominadas dendritos. Há também uma ramificação maior e mais alongada denominada axônio. Os neurônios comunicam-se uns com os outros por meio de sinapses que ocorrem entre o axônio do neurônio transmissor e o dendrito do neurônio receptor. Isso permite que o neurônio responda a estímulos internos e externos ao transmitir impulsos nervosos a outros neurônios (GAMA *et al.*, 2011) (RUSSELL; NORVIG, 2003)

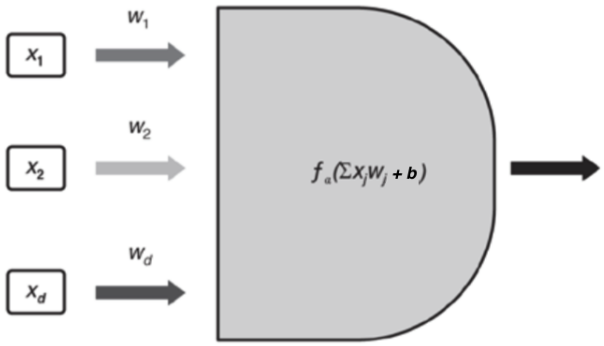
Figura 2 – Representação simplificada de um neurônio biológico



Fonte: Russell e Norvig (2003).

Os neurônios artificiais são modelados como uma função matemática $f_\alpha(\sum x_j w_j + b)$ que, ao receber valores x_j em seus "terminais"de entrada (simulando os dendritos), ponderados com pesos w_j e somados a um viés b, utiliza-os para computar uma saída (simbolizando o axônio) que é propagada a outros neurônios artificiais. A Figura 3 apresenta o modelo de neurônio artificial.

Figura 3 – Modelo de um neurônio artificial. Adaptado de Gama *et al.* (2011).



A função f_α é chamada de função de ativação e exemplos incluem a função degrau:

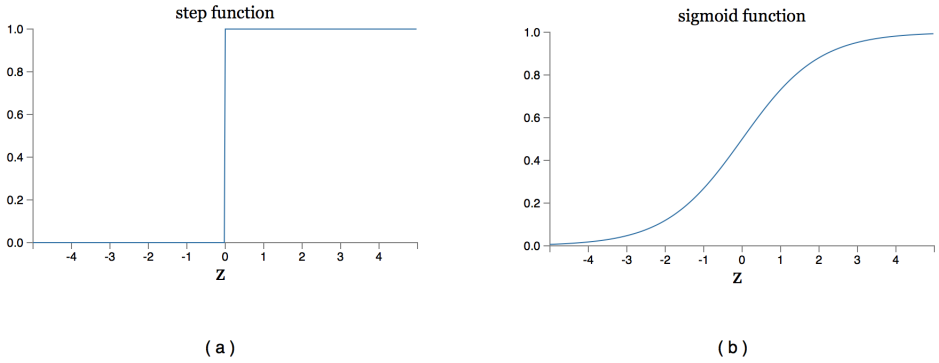
$$f_\alpha(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

e a função sigmoidal:

$$f_\alpha(z) = \frac{1}{1+e^{-z}}$$

onde $z = \sum x_j w_j + b$ (exemplificado na Figura 4) (NIELSEN, 2015) (GAMA *et al.*, 2011).

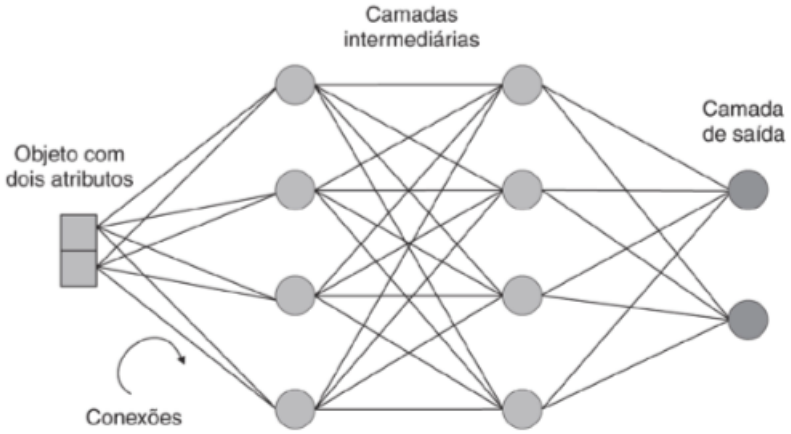
Figura 4 – Exemplos de função de ativação de neurônios artificiais. (a) Função degrau. (b) Função sigmoidal



Fonte: Nielsen (2015).

As Redes Neurais Artificiais podem ser construídas de forma que neurônios artificiais sejam dispostos em camadas. Em redes com mais de uma camada, os neurônios de uma das camadas podem receber como entrada os valores de saída da camada anterior, formando uma malha. A Figura 5 apresenta um exemplo de Rede Neural Artificial, em que há como entrada um conjunto de dois atributos (o que pode ser considerado uma camada de entrada), duas camadas intermediárias e uma camada de saída (GAMA *et al.*, 2011).

Figura 5 – Exemplo de uma Rede Neural Artificial com múltiplas camadas



Fonte: Gama *et al.* (2011).

As Redes Neurais Artificiais são consideradas métodos baseados em otimização. O conjunto de neurônios artificiais representa uma função matemática (combinando as funções

matemáticas de cada neurônio que compõe a rede) e o treinamento da rede consiste em minimizar a medida de erro entre o resultado produzido pela rede e os rótulos dos dados utilizados no treinamento (GAMA *et al.*, 2011). Para realizar o treinamento e, assim, minimizar o erro do modelo, é utilizado um algoritmo conhecido como *backpropagation*, que consiste em uma maneira eficiente de calcular o gradiente envolvido na minimização do erro (NIELSEN, 2015).

2.2.4 Aprendizado Profundo e Arquiteturas

O termo Aprendizado Profundo (AP) (*Deep Learning*, em inglês) refere-se a modelos computacionais formados por Redes Neurais Artificiais de múltiplas camadas, capazes de aprender diferentes representações de dados com vários níveis de abstração. Esses modelos são fundamentais na atualidade, representando o estado da arte em Redes Neurais Artificiais e alavancando o desenvolvimento de tecnologias de reconhecimento de fala, reconhecimento visual de objetos, detecção de objetos e também tendo importância significativa em outras áreas como genômica e processamento e análise de dados médicos (LECUN; BENGIO; HINTON, 2015).

A característica revolucionária do AP, um dos ramos do AM, que impulsionou o desenvolvimento da área de Inteligência Artificial, refere-se à capacidade automática de tais modelos em extrair padrões e representaçõs a partir de "dados crus"ou brutos, isto é, sem a interferência de engenheiros ou especialistas na definição de tais padrões a partir dos dados. Diversas técnicas convencionais de AM apresentam limitações em sua capacidade de processar dados em sua forma natural (sem a intervenção humana). Assim, ao utilizar-se tais técnicas torna-se necessária a presença de mão de obra especializada no domínio do problema, de forma a projetar um sistema capaz de extrair dos dados originais uma representação interna adequada (isto é, extrair atributos que permitam uma análise de dados de maior qualidade) (LECUN; BENGIO; HINTON, 2015).

Dessa forma, o AP tem obtido grande destaque especialmente no processamento de dados de alta dimensionalidade e de estrutura complexa, como por exemplo dados de expressão gênica. No cenário atual, em que o volume de dados gerado cresce exponencialmente e também há maior capacidade computacional para processamento, o AP apresenta-se como uma grande e promissora solução para automatizar o processo de engenharia manual que outrora tornaria impossível o processamento de dados tão complexos (BENGIO, 2016) (LECUN; BENGIO; HINTON, 2015).

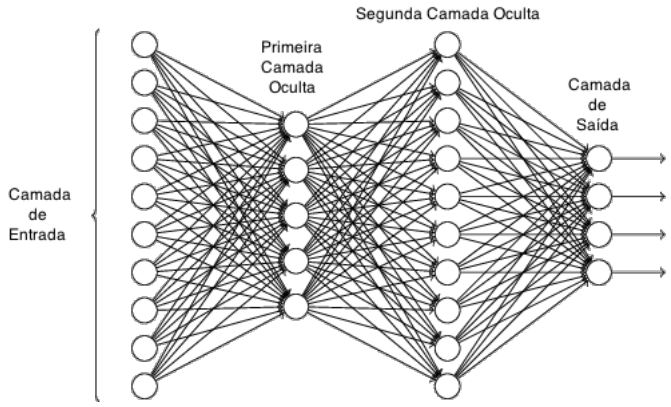
Uma característica fundamental de modelos de AP é a versatilidade: diferentes arquiteturas de Redes Neurais Artificiais podem ser construídas para lidar com problemas diversos. A seguir, são apresentadas algumas dessas arquiteturas, a saber *Multilayer Perceptrons*, Redes Neurais Convolucionais e *Autoencoders*.

2.2.4.1 Multilayer Perceptrons

O tipo mais tradicional de modelo para AP é conhecido como *Multilayer Perceptron* (MLP). No contexto de classificação, MLPs têm o objetivo de obter uma aproximação de uma dada função $y = f^*(x)$. Tal modelo é composto por redes profundas *feedforward*, ou seja, redes onde a informação flui a partir dos dados de entrada, x, propaga-se por camadas de neurônios artificiais que modelam uma aproximação f e finalmente chega a saída computada y. Dessa forma, a rede mapeia entrada e saída a partir da aproximação $y = f(x;\theta)$, onde θ são parâmetros resultantes do treinamento da rede que levam à melhor aproximação de f^* (GOODFELLOW; BENGIO; COURVILLE, 2016).

MLPs são grafos acíclicos compostos por múltiplas camadas artificiais, cada uma representando uma função, que atuam formando uma função composta $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$, onde $f^{(1)}$ é a primeira camada da rede, $f^{(2)}$ a segunda camada e assim por diante. A largura dessa sequência encadeada de funções representa a profundidade da rede, o que deu origem ao nome Aprendizado Profundo (GOODFELLOW; BENGIO; COURVILLE, 2016). A terminologia utilizada geralmente designa a primeira camada à esquerda como camada de entrada; as camadas intermediárias são também chamadas de camadas ocultas; a última camada à direita é denominada camada de saída (NIELSEN, 2015). A Figura 6 apresenta um exemplo de MLP com duas camadas ocultas.

Figura 6 – Exemplo de MLP com duas camadas ocultas. Adaptado de Nielsen (2015).



As redes MLP tem grande importância no cenário de AM, formando o alicerce de diversas aplicações comerciais e também compondo outros tipos de arquiteturas, como Redes Neurais Convolucionais (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.2.4.2 Redes Neurais Convolucionais

Redes Neurais Convolucionais (*Convolutional Neural Network* (CNN), em inglês), também conhecidas simplesmente como Redes Convolucionais *ConvNets*, têm sua importância no processamento de dados que tomam uma forma matricial (forma de *array*), isto é, apresentam uma topologia conhecida que se assemelha a uma grade (LECUN; BENGIO; HINTON, 2015)

(GOODFELLOW; BENGIO; COURVILLE, 2016). Esses dados formam grupos locais que são em geral altamente correlacionados. Por exemplo, imagens apresentam uma topologia matricial de duas dimensões, considerando os diferentes *pixels* que a compõem, com grupos de pixels vizinhos apresentando valores correlacionados. Exemplos de dados em uma dimensão incluem sinais, sequências e séries temporais (que podem ser entendidas como uma grade 1-D em que amostras são coletadas em intervalos de tempo regulares). Uma das principais aplicações de CNNs (e que têm sido uma das maiores contribuições do AP) é o reconhecimento visual de objetos em imagens, utilizado por exemplo para reconhecimento facial e também carros autônomos (GOODFELLOW; BENGIO; COURVILLE, 2016).

Esse tipo de arquitetura de AP é chamado de convolucional porque a rede emprega convolução no lugar de multiplicação de matrizes em pelo menos uma de suas camadas. A convolução é uma operação matemática linear utilizada em vários campos do conhecimento, como engenharia e matemática (GOODFELLOW; BENGIO; COURVILLE, 2016). Formalmente, a operação de convolução envolve a integral de duas funções e produz uma terceira, sendo definida como:

$$s(t) = \int x(a)w(t-a)da$$

A simbologia comumente utilizada para a operação de convolução é apresentada a seguir:

$$s(t) = (x * w)(t)$$

Em CNNs, a operação de convolução é realizada entre uma entrada (representada no exemplo acima por x, geralmente uma matriz de dados) e um *kernel* (w, simbolizando os parâmetros adotados pelo algoritmo de aprendizado). O resultado da operação (s) é conhecido como mapa de atributos (*feature map*). Essa operação reúne ideias que tornam o modelo de aprendizado versátil (GOODFELLOW; BENGIO; COURVILLE, 2016):

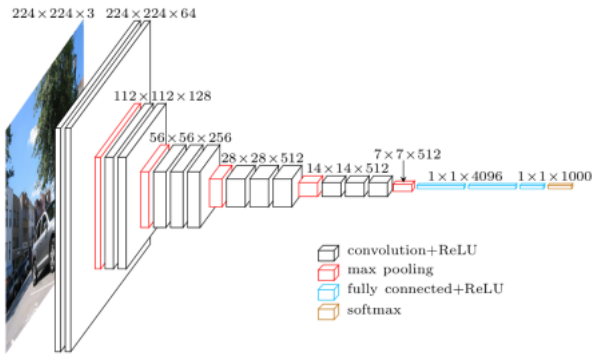
- Interações esparsas: redes neurais tradicionais utilizam multiplicação matricial de forma que cada unidade da entrada interage com cada unidade da saída. Por outro lado, em CNNs são adotadas interações esparsas: fazendo o *kernel* ser menor que a entrada, é possível detectar atributos importantes nos dados (como bordas, por exemplo, em dados de imagens) com menor número de operações matemáticas.

- Compartilhamento de parâmetros: o mesmo parâmetro é utilizado em mais de uma função do modelo, o que permite que o modelo aprenda um mesmo conjunto de parâmetros para todo conjunto de dados, reduzindo a quantidade de espaço de armazenamento necessário para o modelo.

- Representações equivariantes: a função modelada por uma Rede Convolucional é tal que se a entrada da função sofre uma mudança, a saída muda da mesma forma. Isso permite classificar um dado I e uma versão distorcida do mesmo, I', da mesma forma.

Também é utilizada a operação de *pooling* em certas camadas, com o objetivo de fundir características ou atributos semanticamente similares (LECUN; BENGIO; HINTON, 2015). Essa operação substitui a um grupo local da saída da etapa anterior por uma síntese estatística das saídas vizinhas. Por exemplo, na operação de *max pooling* é extraída a maior saída em uma vizinhança retangular. A operação de *pooling* permite realizar uma representação aproximadamente invariante a pequenas translações da entrada (GOODFELLOW; BENGIO; COURVILLE, 2016). A Figura 7 apresenta um exemplo de Rede Convolucional conhecido como VGG16 (SIMONYAN; ZISSERMAN, 2014):

Figura 7 – Exemplo de Rede Convolucional VGG16 para processamento de imagens, com diversas camadas de convolução e de *pooling*



Fonte: Blier (2016).

2.2.4.3 Autoencoders

Um *Autoencoder* é uma rede neural que utiliza principalmente aprendizado não supervisionado e cuja função é tentar reproduzir sua entrada na saída, sendo que a informação passa, internamente, por uma ou mais camadas ocultas que representam uma codificação da entrada. Esse tipo de rede neural pode ser modelado como uma função codificadora $h = f(x)$ associada a uma função decodificadora $r = g(h)$, que tenta reconstruir a entrada. Entretanto, a utilidade de um *Autoencoder* vem do fato de o mesmo ser construído de forma que a reconstrução da entrada não seja perfeita. Essas restrições têm a função de forçar a priorização de aspectos da entrada de maior relevância, permitindo que se tome conhecimento de importantes propriedades dos dados sendo manipulados. Assim, uma das aplicações de *Autoencoders* é a redução de dimensionalidade de dados: a partir do aprendizado das características mais importantes desses dados, é possível representá-los com menos atributos (GOODFELLOW; BENGIO; COURVILLE, 2016). A capacidade de extrair tanto propriedades lineares quanto não lineares a partir dos dados é uma

das características que faz *Autoencoders* serem considerados poderosos e versáteis (DANAEE; GHAEINI; HENDRIX, 2016).

Uma variante dos *Autoencoders* tradicionais é o *Denoising Autoencoder*, que tenta reconstruir uma entrada a partir de uma versão corrompida com ruído da mesma. *Autoencoders* tradicionais buscam reduzir uma função de perda L utilizada para penalizar a falta de similaridade entre a entrada original e sua reconstrução:

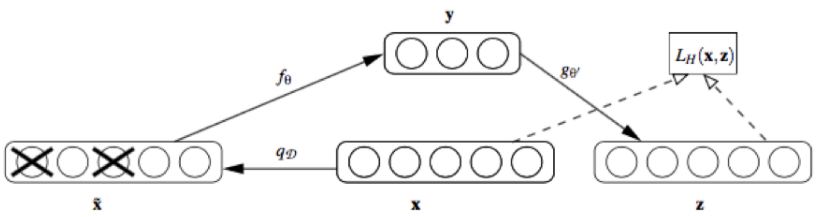
$$L(x, g(f(x)))$$

Já *Denoising Autoencoders* minimizam a função

$$L(x, g(f(\tilde{x})))$$

onde $\tilde{x}$ é uma versão corrompida de x com ruído. Assim, a tarefa de reconstrução da entrada realizada por *Denoising Autoencoders* envolvem também a remoção de ruído, e isso força as funções de codificação e decodificação do autoencoder a implicitamente aprenderem propriedades importantes dos dados (GOODFELLOW; BENGIO; COURVILLE, 2016). A Figura 8 apresenta em linhas gerais o funcionamento de um *Denoising Autoencoder*. Um exemplo de entrada x é corrompido com ruído q_D, formando a entrada corrompida $\tilde{x}$. Uma função f_θ mapeia $\tilde{x}$ em uma representação y de dimensionalidade reduzida, que é utilizada para a tentativa de reconstrução de x denominada z, a partir de $g_{\theta'}$. O erro da reconstrução é computado pela função $L_H(x, z)$.

Figura 8 – Funcionamento de um *Denoising Autoencoder*



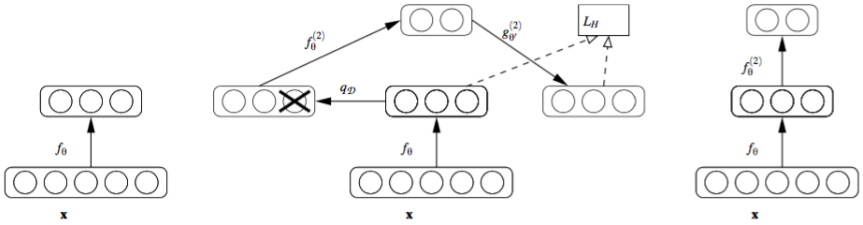
Fonte: Vincent *et al.* (2010).

Uma técnica interessante aplicada a *Autoencoders* consiste no empilhamento dos mesmos, formando uma estrutura denominada *Stacked Autoencoders*. Nesse tipo de estrutura, que é combinada com *Denoising Autoencoders*, o treinamento da rede neural acontece camada a camada. Primeiramente, considerando um *Denoising Autoencoder* simples formado por um codificador e um decodificador, em que o codificador formará a primeira camada do modelo final, utiliza-se a entrada corrompida $\tilde{x}$ para treinamento. Após realização do treinamento, a entrada original x é usada para ser extraída sua representação de dimensionalidade reduzida, y, a

partir da aplicação de f_θ. Depois disso, inicia-se o treinamento da segunda camada, separada da primeira por enquanto, formada também por um codificador e um decodificador. O treinamento acontece de forma similar: agora, y é a entrada da segunda camada, que também é corrompida com ruído, gerando $\tilde{y}$. O modelo é treinado e por fim a entrada original é utilizada para obter-se uma nova representação de dimensionalidade reduzida de y. Esse processo repete-se camada a camada. Por fim, os codificadores que foram treinados separadamente são combinados formando um *Stacked Denoising Autoencoder* (SDAE) (VINCENT *et al.*, 2010).

A Figura 9 ilustra o processo de treinamento camada a camada de um SDAE. Primeiro, à esquerda, é mostrada a aplicação de f_θ sobre os dados de entrada originais x, obtendo uma representaçao de dimensionalidade reduzida, após treinamento da primeira camada como mostrado na Figura 8. No centro, essa nova representação com menos dimensões é utilizada na próxima camada do processo, de forma independente, obtendo uma terceira representação de dimensionalidade mais reduzida ainda. À direita, por fim, temos a combinação das diferentes funções de codificação que formam o SDAE.

Figura 9 – Treinamento camada a camada de um SDAE



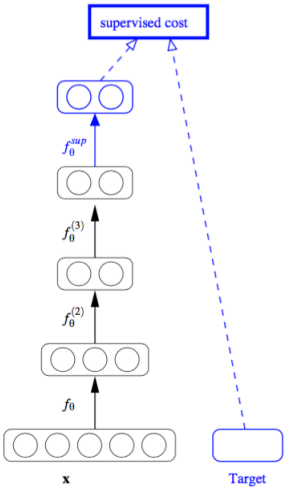
Fonte: Vincent *et al.* (2010).

Após o treinamento não supervisionado de um SDAE pode-se executar uma técnica denominada *fine-tuning*, onde aprendizado supervisionado é utilizado para ajustar os parâmetros aprendidos pela rede na etapa não supervisionada. Para isso, pode-se adicionar uma última camada de regressão logística, por exemplo, à rede neural formada pelos diferentes codificadores, formando uma rede neural profunda capaz de utilizar aprendizado supervisionado (exemplificado na Figura 10) (VINCENT *et al.*, 2010).

2.2.5 Balanceamento de Dados

Um problema recorrente em tarefas de AM é o problema do aprendizado desbalanceado. Isto é, num conjunto de dados de dois tipos de classe, por exemplo, o conjunto é dito desbalanceado quando o número de amostras de uma determinada classe (denominada classe majoritária) é notadamente superior ao número de amostras da outra classe (denominada classe minoritária). Esse desbalanceamento pode comprometer o desempenho de algoritmos de AM, já que em geral tais algoritmos assumem uma distribuição balanceada de amostras nas diferentes

Figura 10 – Operação de *fine-tuning*, utilizando aprendizado supervisionado para ajustar os parâmetros do SDAE



Fonte: Vincent *et al.* (2010).

classes disponíveis em um problema ou consideram penalizações de erros de classificação iguais para diferentes classes (HE; GARCIA, 2009). Com dados desbalanceados, os algoritmos de AM podem encontrar dificuldades em aprender conceitos relacionados à classe minoritária (BATISTA; PRATI; MONARD, 2004).

Para solucionar o problema de dados desbalanceados, técnicas de balanceamento podem ser aplicadas ao conjunto de dados sendo utilizado em uma tarefa de AM. Essas técnicas podem envolver a adição de amostras ao conjunto de dados (*oversampling*), remoção de amostras do conjunto de dados (*undersampling*) e também misturar adição e remoção. Uma importante técnica de adição de amostras sintéticas chama-se SMOTE (CHAWLA *et al.*, 2002); já *NearMiss* (MANI; ZHANG, 2003) é uma conhecida técnica de remoção de amostras; e SMOTE+Tomek (BATISTA; PRATI; MONARD, 2004) é uma técnica que mistura adição e remoção. Essas técnicas são apresentadas sucintamente a seguir:

- SMOTE (*Synthetic Minority Over-sampling Technique*) é um algoritmo utilizado para criar dados artificiais baseados na semelhança no espaço de atributos entre diferentes amostras da classe minoritária. Nessa técnica são considerados os K vizinhos mais próximos de uma amostra da classe minoritária. Um desses K vizinhos é escolhido aleatoriamente e então é gerado um exemplo sintético que pertença ao segmento de reta que conecta o exemplo ao vizinho.

- NearMiss é uma classe de algoritmos que utiliza um classificador KNN (*K-nearest neighbor*) para remover exemplos da classe majoritária. Por exemplo, as amostras removidas são aquelas com menor média de distância aos 3 exemplos mais próximos da classe minoritária (caso do NearMiss-1) ou aos 3 exemplos mais distantes da classe minoritária (caso do

NearMiss-2).

- SMOTE+Tomek é a aplicação da técnica SMOTE associada à limpeza de dados realizada pela técnica de elos Tomek. Assim, é possível lidar com a sobreposição de dados é introduzida pelo uso de técnicas de *oversampling* como SMOTE.

2.2.6 Redução da dimensionalidade de dados

Dados de diversos domínios podem apresentar alta dimensionalidade, como é o caso de dados de expressão gênica. Nesses casos, diversos atributos irrelevantes e redundantes podem representar uma espécie de ruído para o conhecimento que pode ser extraído a partir do conjunto de dados. Dessa forma, técnicas de redução de dimensionalidade podem ajudar a obter um espaço de atributos que torne mais efetiva a atuação de algoritmos de AM sobre determinado conjunto de dados (TANG; ALELYANI; LIU, 2014).

Técnicas de redução de dimensionalidade podem ser categorizadas basicamente de duas formas: seleção de atributos e extração de atributos. O funcionamento da seleção de atributos consiste na escolha de um subconjunto de atributos que minimize redundância e maximize relevância na análise em questão. Já a extração de atributos projeta o espaço de atributos original em um novo espaço com menor dimensionalidade e novos atributos que são construídos a partir dos atributos originais. Essa projeção, entretanto, torna difícil o mapeamento entre os atributos originais e os novos atributos construídos, o que dificulta a interpretação de cada atributo do novo espaço de atributos. Por outro lado, a seleção de atributos permite manter os atributos originais e assim conservar a interpretação dos mesmos. Em dados de expressão gênica, por exemplo, é importante manter o significado dos atributos originais para que genes responsáveis por doenças, por exemplo, sejam identificados (TANG; ALELYANI; LIU, 2014).

Assim como citado na seção 2.2.4.3, SDAEs podem ser utilizados para redução de dimensionalidade de dados. Como essa redução é feita a partir da codificação dos dados, essa técnica enquadra-se na categoria de extração de atributos. Por outro lado, ReliefF (KIRA; RENDELL, 1992) e FCBF (YU; LIU, 2003) são exemplos de técnicas de seleção de atributos. ReliefF é uma técnica baseada em análise estatística, isto é, seleciona atributos baseando-se na relevância estatística dos mesmos para o conceito sendo aprendido. FCBF é um ágil método baseado em informação mútua, que avalia a correlação entre atributos. Ambos ReliefF e FCBF são técnicas de seleção de atributos baseadas em filtros, onde são analisadas as características dos dados isoladamente de qualquer algoritmo de classificação (TANG; ALELYANI; LIU, 2014).

2.2.7 Avaliação de Desempenho de Classificadores

Para avaliar a qualidade de um classificador utilizado em tarefas preditivas costuma-se utilizar novos exemplos rotulados (não utilizados na etapa de treinamento), que são classificados

pelo modelo. Então, compara-se os rótulos esperados dos exemplos utilizados com as predições feitas pelo modelo (GAMA *et al.*, 2011).

As métricas de desempenho mais frequentemente utilizadas são a taxa de acertos (também conhecida como acurácia) e a taxa de erros, definidas como a seguir (HE; GARCIA, 2009):

$$\text{Taxa de Acertos} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{\text{Número de acertos no conjunto de teste}}{\text{Número de amostras no conjunto de teste}}$$

$$\text{Taxa de Erros} = 1 - \text{Taxa de Acertos}$$

onde TP é o número de amostras de verdadeiro positivo, TN de verdadeiro negativo, FP de falso positivo e FN de falso negativo.

No entanto, essas métricas fornecem apenas uma visão simples do desempenho do classificador e podem levar a conclusões precipitadas no caso de dados desbalanceados. Para solucionar esse problema, são utilizadas outras métricas que permitem análises mais compreensivas, como G-mean (HE; GARCIA, 2009) (NETO, 2016). A métrica G-mean é definida como:

$$\textit{G-mean} = \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}}$$

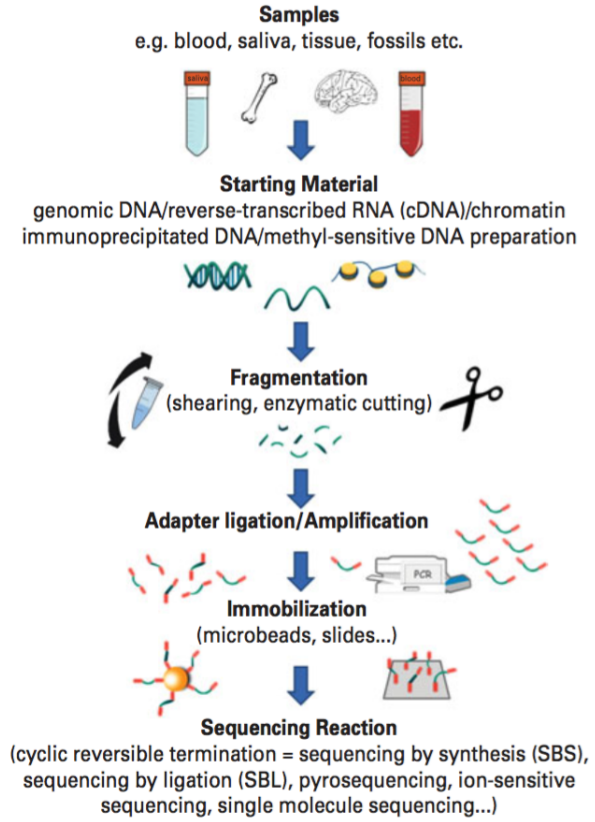
Geralmente, em uma tarefa preditiva dispõe-se de apenas um conjunto de dados que deve ser utilizado tanto para treinamento quanto para validação do modelo. Uma abordagem comumente utilizada para separação dos dados em conjunto de treinamento e conjunto de teste é a validação cruzada, também chamada de *r-fold cross validation* (GAMA *et al.*, 2011). Essa abordagem consiste na divisão do conjunto de exemplos em r grupos, usando $r-1$ grupos na etapa de treinamento e posteriormente usando o grupo restante na etapa de teste. O processo repete-se r vezes, sendo utilizado um grupo diferente para teste em cada vez. O desempenho do modelo, ao final dos r ciclos, é dado pela média dos desempenhos sobre cada conjunto de testes.

2.2.8 Tecnologia RNA-Seq

Para possibilitar a compreensão e entendimento das variações e funções do genoma humano, foi preciso realizar o sequenciamento do genoma de múltiplas espécies e, dada a espécie humana, de diversas pessoas, obtendo assim leituras de expressão gênica, metilação de Ácido Desoxirribonucleico (DNA) e estutura de cromatina. Assim, evidenciou-se a necessidade de uma revolução tecnológica que tornasse a tarefa de sequenciamento mais poderosa e reduzisse seus custos. Como resultado dessa revolução tecnológica tem-se o estado da arte em tecnologia de sequenciamento da atualidade: o *Deep Sequencing*, isto é, Sequenciamento em Grande Escala (GOLDMAN; DOMSCHKE, 2014).

Esse tipo de sequenciamento, também conhecido como *Next Generation Sequencing*, *High-throughput Sequencing* e *Massively Parallel Sequencing*, é uma tecnologia que permite que o sequenciamento de grandes porções de genomas (e também de genomas inteiros) seja feito com baixas taxas de erros e ruídos, atla precisão e baixo custo. A Figura 11 ilustra o processo de sequenciamento realizado pela tecnologia *Deep Sequencing*: primeiramente, amostras de DNA ou Ácido Ribonucleico (RNA) são extraídas da fonte (por exemplo, sangue, saliva, tecido). Então, as amostras são transcritas reversamente, formando amostras de DNA complementar (cDNA), que são fragmentadas em pequenos pedaços (de forma física ou por reações enzimáticas). Além disso, em função das necessidades de cada tecnologia, tais fragmentos podem ser ligados a adaptadores feitos de DNA sintético e separados ou imobilizados, sendo finalmente realizado o sequenciamento de maneira massivamente paralela (GOLDMAN; DOMSCHKE, 2014).

Figura 11 – O processo de *Deep Sequencing*



Fonte: Goldman e Domschke (2014).

Uma abordagem recente para análise de expressão gênica e que utiliza *Deep Sequencing* é a tecnologia *RNA Sequencing* (RNA-Seq). Essa abordagem é utilizada para mensurar níveis do transcritoma, que é o conjunto completo de transcritos (RNA mensageiro, RNA ribossômico, RNA transportador e micro-RNA) de uma célula ou conjuntos de células em um determinado momento. A partir do entendimento do transcritoma é possível compreender expressão gênica, desenvolvimento celular e desenvolvimento de doenças (WANG; GERSTEIN, 2009).

Embora seja recente, a técnica de RNA-Seq já modificou o entendimento que se tem sobre a complexidade e o tamanho de transcritomas de seres eucariontes (WANG; GERSTEIN, 2009) e tende a substituir técnicas analógicas como *Microarray* (ROY *et al.*, 2011).

A tecnologia *Microarray* torna possível a análise da expressão de uma grande quantidade de genes de maneira simultânea, a partir da reação de hibridização entre sondas de DNA fixadas em pastilhas e amostras de teste de RNA ou DNA (SOUZA, 2010). Essa tecnologia é muito difundida no meio científico. Os tipos de plataformas *Microarray* podem ser divididos em dois grupos: (i) *Microarray* de DNA complementar e (ii) *Microarray* de oligonucleotídeos (COLOMBO; RAHAL, 2009).

A tendência da tecnologia RNA-Seq substituir *Microarray* deve-se às suas vantagens, como por exemplo:

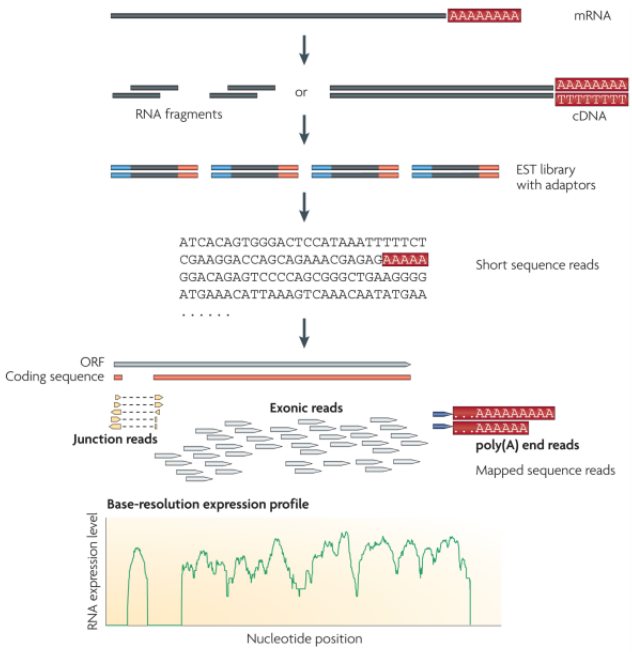
- alta precisão nos níveis de transcritos e na quantificação de níveis de expressão gênica, enquanto a tecnologia de *Microarray* apresenta altos níveis de ruído (WANG; GERSTEIN, 2009).
- não limitação na detecção de transcritos, isto é, a possibilidade de detectar transcritos que não correspondem a nenhuma sequência genômica existente, o que torna possível a determinação de novas sequências genômicas. Em outras palavras, não é preciso saber dos genes de interesse *a priori*. Essa última característica é uma grande vantagem com relação à tecnologia de *Microarray*, baseada em conhecimento prévio sobre o sequenciamento do genoma (WANG; GERSTEIN, 2009) (ROY *et al.*, 2011).

Por outro lado, ainda existem vantagens da tecnologia *Microarray* em relação à tecnologia RNA-Seq. A maior delas deve-se à grande difusão e utilização da mesma no meio científico: por exemplo, as ferramentas disponíveis para análise de dados de expressão gênica de *Microarray* têm mais maturidade que aquelas dedicadas a RNA-Seq. Além disso, o volume de dados gerado pela tecnologia *Microarray* é significativamente menor que no caso de RNA-Seq, o que pode representar maior facilidade de manipulação de dados por parte de cientistas e pesquisadores (ROY *et al.*, 2011).

O funcionamento da tecnologia RNA-Seq, por ser uma abordagem que utiliza *Deep Sequencing*, é muito semelhante ao processo representado na Figura 11 e está sucintamente representado na Figura 12. Primeiramente, as amostras de RNA são fragmentadas e convertidas em cDNA. Então, a cada fragmento de cDNA são adicionados adaptadores. Depois, são feitas leituras de pequenas sequências de cada fragmento de cDNA, utilizando-se alguma tecnologia de sequenciamento de *high-throughput*. Essas sequências são então comparadas com o genoma ou transcritoma de referência, classificadas e assim são gerados os perfis de expressão gênica para cada gene (WANG; GERSTEIN, 2009).

Entretanto, RNA-Seq apresenta algumas dificuldades e desafios, destacando-se:

Figura 12 – Processo de um experimento RNA-Seq



Fonte: Wang e Gerstein (2009).

- longas moléculas de RNA precisam ser fragmentadas para que sejam compatíveis com tecnologias de *Deep Sequencing*. O método de fragmentação utilizado confere um viés ao resultado (WANG; GERSTEIN, 2009).
- uma maior cobertura do sequenciamento implica um custo maior, isto é, mais cobertura exige mais profundidade de sequenciamento (WANG; GERSTEIN, 2009).
- os grandes e crescentes volumes de dados gerados por essa tecnologia representam um desafio (ROY *et al.*, 2011). Por isso, a aplicação de algoritmos de aprendizado de máquina mostra-se útil na análise desse tipo de dados.

2.3 Trabalhos Relacionados

Múltiplos esforços têm sido despendidos na pesquisa e na elaboração de métodos de análise de dados de expressão gênica. Um dos precursores na área foi o trabalho realizado por Golub *et al.* (1999): dados de *Microarray* foram analisados para descoberta e predição de classes de câncer. Souza (2010), por sua vez, investigou meta-aprendizado na recomendação de algoritmos de classificação para análise de dados *Microarray*. E no trabalho realizado por (TAN; GILBERT, 2003), três técnicas de aprendizado supervisionado foram comparadas na classificação de câncer a partir de dados *Microarray*: árvores de decisão C4.5, *bagged* e *boosting*.

Embora seja mais recente, a tecnologia RNA-Seq também já foi utilizada em trabalhos da área: Neto (2016) avaliou diferentes técnicas de seleção de atributos e algoritmos de aprendizado de máquina na análise de dados RNA-Seq para identificação de tumores. Já Danaee, Ghaeini

e Hendrix (2016) apresentaram abordagens de aprendizado profundo para detecção de câncer, utilizando SDAEs para extração de *features* de relevância e usando *Support Vector Machine* (SVM) e Redes Neurais Artificiais como classificadores. Uma outra metodologia utilizando aprendizado profundo foi proposta por Chen *et al.* (2016), que apresentam um método baseado em redes neurais artificiais multicamadas para inferência de expressão gênica tanto em dados *Microarray* como RNA-Seq.

2.4 Considerações Finais

Neste capítulo, foi apresentada uma revisão bibliográfica das tecnologias e técnicas de relevância para este trabalho. Começando por Aprendizado de Máquina, foi feita uma breve descrição dos tipos de tarefas que podem ser resolvidas com esse ramo da Inteligência Artificial, além de apresentadas métricas para avaliação do desempenho de modelos preditivos. Em especial, um desses modelos foi abordado em mais detalhes: Redes Neurais Artificiais. O modelo de Árvores de Decisão também foi exposto. Além disso, foi abordado o conceito de Aprendizado Profundo e foram apresentadas arquiteturas relevantes. Tratou-se também dos temas relacionados a balanceamento e redução de dimensionalidade dos dados. Após isso, a técnica de RNA-Seq foi apresentada de maneira breve, sendo destacados, em linhas gerais, seus mecanismos de funcionamento. O capítulo foi concluído com uma discussão de trabalhos relacionados. Na próxima seção, serão abordadas a metodologia e as técnicas utilizadas neste trabalho, apresentando em detalhes o problema de classificação de dados de expressão gênica, descrevendo a etapa de pré-processamento de dados e discutindo os experimentos realizados, resultados obtidos e dificuldades encontradas.

Capítulo 3

DESENVOLVIMENTO

3.1 Considerações Iniciais

Neste capítulo serão apresentados os detalhes deste projeto. Serão discutidos os objetivos do trabalho, bem como a metodologia empregada e também serão descritas as atividades realizadas. Por fim, serão apresentados os resultados obtidos a partir dos experimentos realizados e as dificuldades enfrentadas.

3.2 Projeto

O câncer é uma doença responsável por aproximadamente 13% das mortes ao redor do mundo, sendo que em 2008 cerca de 7,6 milhões de mortes foram ocasionadas por tal doença (VIDYASAGAR, 2014). Dessa forma, é de extrema relevância que sejam feitos estudos acerca dos motivos de causa do câncer, por exemplo analisando a influência dos níveis de expressão gênica nos diferentes tipos dessa doença. Diferentes tecnologias, como *Microarray* e RNA-Seq, têm sido empregadas para deduzir e quantificar o transcriptoma, viabilizando a análise da expressão gênica de indivíduos e também gerando uma considerável quantidade de dados (WANG; GERSTEIN, 2009). Assim, o aprendizado de máquina apresenta-se como ferramenta importante no processamento desses tipos de dados, tornando viável a análise de grandes quantidades de informação (SAJDA, 2006).

Nesse contexto, diversos trabalhos têm sido realizados na área, utilizando algoritmos e técnicas de AM como SVM, Redes Neurais Artificiais, Árvores de Decisão, Meta-Aprendizado e Aprendizado Profundo (DANAEE; GHAEINI; HENDRIX, 2016; TAN; GILBERT, 2003; SOUZA, 2010; CHEN *et al.*, 2016; NETO, 2016). Especialmente, algoritmos de AP têm a capacidade de extrair automaticamente conceitos e padrões relevantes a partir de dados brutos, facilitando o trabalho envolvendo dados de domínios complexos como dados biológicos (LECUN; BENGIO; HINTON, 2015). Portanto, o objetivo deste projeto é comparar o potencial de algoritmos de aprendizado profundo na análise de dados de expressão gênica obtidos com técnica RNA-Seq.

Para realizar essa comparação, a proposta do trabalho foi coletar e preparar dados de expressão gênica de domínio público e utilizar arquiteturas distintas de algoritmos de AP para processá-los, preparando-se assim um ambiente computacional para realização de múltiplos

experimentos. Esses experimentos consistiram em tarefas de classificação, em que foram utilizados os dados de expressão gênica nas fases de treinamento e teste dos modelos de AM. A técnica de validação cruzada foi empregada nessas tarefas de classificação para dividir os dados entre conjunto de treinamento e conjunto de teste. Cada experimento foi composto das seguintes variáveis:

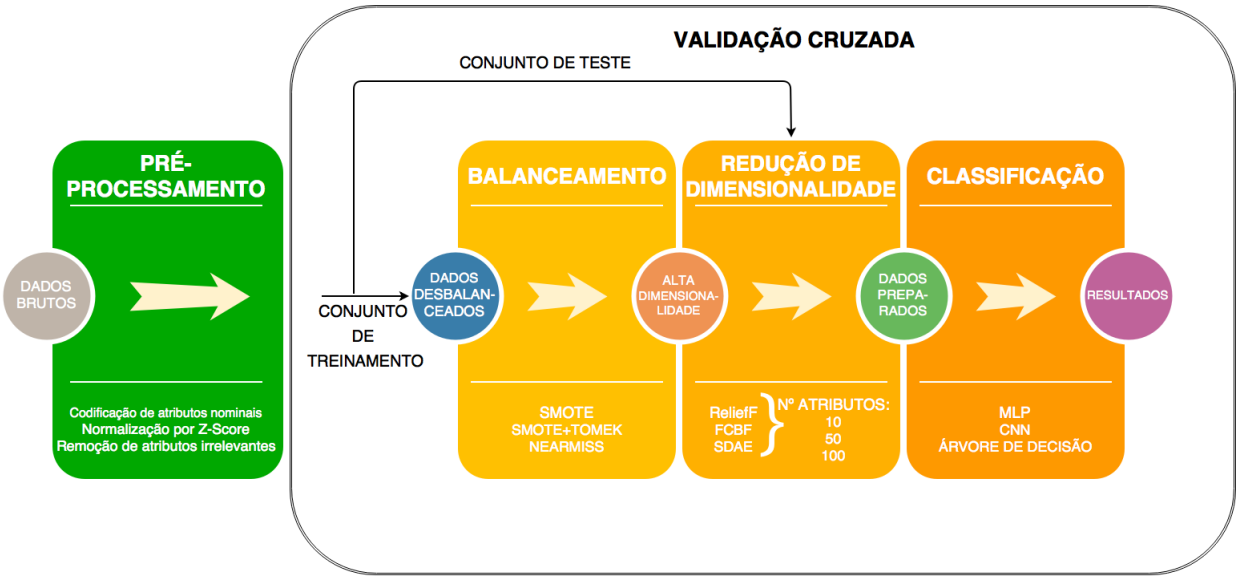
- Algoritmo de balanceamento de dados: primeiramente, foi escolhido um algoritmo de balanceamento para cuidar dos dados desbalanceados. Os dados de expressão gênica apresentam mais amostras de determinada classe que de outra, como será mostrado adiante. Assim, o uso de técnicas de balanceamento lidou com esse problema. Neste trabalho, 3 técnicas de balanceamento distintas foram analisadas.
- Técnica de redução de dimensionalidade: dada a alta dimensionalidade dos dados de expressão gênica, torna-se interessante utilizar alguma técnica que possibilite a análise dos dados em um espaço de atributos com menos dimensões. Assim, foram comparadas 3 técnicas de seleção e de extração de atributos.
- Número de atributos selecionados/extraídos: ainda no contexto de redução de dimensionalidade, foi variado o número de atributos final do conjunto de dados, após aplicação da técnica de redução de dimensionalidade. Foram utilizadas 3 opções de número de atributos.
- Algoritmo de classificação: por fim, a última variável de cada experimento diz respeito ao algoritmo de classificação utilizado. Três algoritmos de classificação foram considerados.

Primeiramente, foi realizado um pré-processamento dos dados, preparando-os para cada experimento. A partir daí foram executados os diferentes experimentos, cujas etapas aconteceram na ordem apresentada acima: primeiro foi realizado o balanceamento dos dados; após isso, a redução de dimensionalidade; e por fim, a classificação. Essa combinação de etapas compreendeu cada experimento individual de um total de 81 experimentos por base de dados, sendo que a validação cruzada (*5-fold cross validation*) foi realizada para cada um desses experimentos. Isto é, cada experimento individual foi realizado múltiplas vezes (mais especificamente, 5 vezes), sendo que os conjuntos de dados de treinamento e de teste foram variados em cada uma delas. Cada um desses experimentos foi aplicado a cada um dos conjuntos de dados considerados neste trabalho (ou seja, cada base de dados foi processada um total de 405 vezes).

A Figura 13 apresenta a arquitetura dos experimentos realizados, delineando as etapas de cada experimento. A validação cruzada envolve todas as diferentes etapas dos experimentos. O pré-processamento dos dados acontece apenas uma vez para cada conjunto de dados.

Na próxima seção serão detalhados os diferentes módulos que compõem a arquitetura dos experimentos, além de apresentados os conjuntos de dados sendo utilizados.

Figura 13 – Etapas na realização de cada experimento



3.3 Atividades Realizadas

Nesta seção são discutidos o ambiente computacional empregado, os processos de obtenção e de pré-processamento de dados e as técnicas utilizadas em cada etapa dos experimentos.

3.3.1 Ambiente Computacional

Para realização dos experimentos, as linguagens de programação *R*[1] (versão 3.4.1) e *Python*[2] (versão Python 3.5.2 - Anaconda Custom 64-bit) foram utilizadas. Essas linguagens foram escolhidas por serem *open source* e por serem muito difundidas e utilizadas pela comunidade de AM, oferecendo diversas bibliotecas, *frameworks* e ferramentas relacionadas à área. A linguagem *R* foi utilizada para o pré-processamento dos dados; já *Python* foi a linguagem escolhida para realizar as demais etapas dos experimentos. As bibliotecas utilizadas neste trabalho serão apresentadas nas próximas seções, correspondentes às etapas em que foram empregadas.

Os experimentos realizados utilizaram computação em nuvem, fazendo uso dos serviços *Azure*[3]. A plataforma *Azure* oferece a infraestrutura computacional que se mostrou necessária para a realização dos experimentos: devido ao grande volume de dados, dados esses de alta dimensionalidade, e à quantidade de experimentos executados para cada base de dados, tornou-se impraticável realizar tais experimentos em um computador pessoal padrão. Uma infraestrutura computacional com grande poder de processamento e notável capacidade de memória foi necessária. Assim, foi utilizada uma máquina virtual do tipo *Standard NC6* na plataforma *Azure*. Essa máquina ofereceu os seguintes recursos:

1 https://www.r-project.org/
2 https://www.python.org
3 https://azure.microsoft.com/en-us/

- 6 vCPUs Intel Xeon CPU E5-2690 v3 @ 2.60GHz
- 1 GPU NVIDIA Corporation GK210GL - Tesla K80, com 16 GB de memória
- 56 GB de RAM

O poder computacional das CPUs reduziu significativamente o tempo de processamento necessário para realizar os experimentos com as bases de dados. Além disso, a grande quantidade de RAM permitiu que fossem mantidos em memória modelos de grandes redes neurais artificiais. Entretanto, a presença de *Graphics Processing Unit* (GPU) foi o que otimizou o processamento das redes neurais artificiais que, devido ao seu tamanho e complexidade, representaram o gargalo computacional dos experimentos. A grande vantagem da utilização de GPUs é a capacidade de realização de computação paralela de forma poderosa e eficiente (NVIDIA Corporation, 2017), o que auxiliou em muito os cálculos e operações matriciais envolvidos no treinamento de redes neurais artificiais. As bibliotecas em *Python* utilizadas para modelagem de redes neurais artificiais, que serão apresentadas adiante, possuem implementações otimizadas para GPUs, o que permitiu a utilização desse recurso. Mesmo assim, o tempo de processamento dos dados foi em média de 10 horas por base de dados RNA-Seqv2 e 1 hora para os dados miRNA-Seq.

O sistema operacional utilizado para realização dos experimentos foi *Linux*, mais especificamente a distribuíção *Ubuntu*[4] (versão 16.04.3 LTS). Além de ser um sistema operacional *open source* e possuir uma vasta comunidade de usuários, a plataforma *Azure* oferece uma imagem para máquina virtual da distribuição *Ubuntu* que já possui um pacote de ferramentas de AM instalado[5]. Esse pacote inclui as linguagens de programação e muitas das bibliotecas utilizadas no trabalho, além de ter pré-configuradas as ferramentas *CUDA Toolkit*[6] (utilizada para habilitar um ambiente de desenvolvimento que faça uso da GPU NVIDIA) e *NVIDIA cuDNN*[7] (uma biblioteca de primitivas para Redes Neurais Profundas utilizada por aplicações que empregam GPUs NVIDIA).

A ferramenta *Git*[8] foi utilizada para controle de versão do código desenvolvido no trabalho, em conjunto à plataforma de desenvolvimento *GitHub*[9]. O código desenvolvido e os resultados dos experimentos podem ser acessados em <https://github.com/henrisilver/DeepLearningCancer>.

3.3.2 Conjuntos de Dados e Pré-Processamento

Os dados de expressão gênica utilizados neste trabalho dizem respeito à tecnologia RNA-Seq. Mais especificamente, foram utilizados dados de dois conjuntos: Illumina HiSeq

[4] https://www.ubuntu.com
[5] https://azuremarketplace.microsoft.com/en-us/marketplace/apps/microsoft-ads.linux-data-science-vm-ubuntu
[6] https://developer.nvidia.com/cuda-toolkit
[7] https://developer.nvidia.com/cudnn
[8] https://git-scm.com
[9] https://github.com

2000 RNA Sequencing Version 2 (RNA-Seqv2) e Illumina HiSeq 2000 miRNA Sequencing (miRNA-Seq). Esses conjuntos de dados foram obtidos a partir do trabalho anterior realizado por Neto (2016), que coletou os dados a partir de repositórios do *The Cancer Genome Atlas* (TCGA)[10], uma colaboração entre o *National Cancer Institute*[11] e do *National Human Genome Research Institute* [12]. Neto (2016) utilizou a biblioteca *TCGABiolinks*[13] para coletar os dados RNA-Seqv2 e a ferramenta *FireBrowse* [14] para coletar os dados miRNA-Seq.

Os dados RNA-Seqv2 utilizam normalização *RNASeq by Expectation Maximization* (RSEM) e têm seus valores de expressão gênica medidos através de RNA mensageiro. Já os dados miRNA-Seq utilizam os valores de *read per million miRNA mapped* como medida de expressão gênica (NETO, 2016). Além disso, genes com baixa expressão (atributos com valor 0 para todos as amostras) foram removidos (NETO, 2016). Foram escolhidas 6 bases de dados de cada conjunto. Cada uma dessas bases, dentro de cada conjunto, apresenta dados relacionados a um tipo de câncer particular. Todas essas bases de dados apresentam amostras que são rotuladas com TP (*Primary Solid Tumor*), correspondendo a amostras doentes (com câncer), ou NT (*Solid Tissue Normal*), correspondendo a amostras saudáveis (National Cancer Institute, 2017).

As bases de dados utilizadas estavam dispostas em arquivos de texto individuais, em formato matricial $A \times T$, onde cada uma das A linhas corresponde a uma amostra e cada uma das T colunas a um atributo. Cada atributo corresponde a um gene ou, no caso da última coluna, à classe (TP ou NT) da amostra. A Tabela 1 apresenta informações sobre as bases de dados RNA-Seqv2 e a Tabela 2 apresenta informações sobre as bases de dados miRNA-Seq.

O pré-processamento desses dados foi realizado utilizando a linguagem R. Esse pré-processamento consistiu em:

- Converter o formato dos arquivos: inicialmente, os arquivos de dados possuíam a extensão *.txt*, sendo que o delimitador entre os dados era um espaço em branco. Os arquivos foram convertidos para o formato CSV (*Comma Separated Values*) (REPICI, 2002), com vírgulas usadas como delimitadores, já que tal formato é o mais comum para representar planilhas e bases de dados (Python Software Foundation, 2017).

- Codificação da variável dependente: a variável dependente, "classe", que designa se uma amostra é classificada como "saudável"ou "com câncer", estava codificada originalmente como "NT"ou "TP", respectivamente. Esses valores nominais foram substituídos por 1 e 0 (respectivamente), já que redes neurais artificiais lidam apenas com dados numéricos (GAMA *et al.*, 2011).

[10] https://cancergenome.nih.gov/
[11] https://www.cancer.gov/
[12] https://www.genome.gov/
[13] https://www.bioconductor.org/packages/release/bioc/html/TCGAbiolinks.html
[14] http://firebrowse.org/

- Normalização utilizando *z-score*: as demais variáveis (variáveis independentes representando cada gene) foram normalizadas utilizando-se para isso o processo *z-score* (padronização) , sendo que cada coluna do conjunto passou a possuir média 0 e desvio padrão 1. Essa padronização foi realizada para evitar que algum atributo tivesse predominância sobre os demais, minimizando o viés de um atributo sobre outro nas redes neurais artificiais utilizadas. Além disso, a padronização tende a acelerar o processo de convergência no treinamento da rede, já que os diferentes atributos estão na mesma escala (PRIDDY; KELLER, 2005).

Os dados utilizados neste trabalho podem ser acessados via <https://github.com/henrisilver/DeepLearningCancer>.

Tabela 1 – Bases de dados RNA-Seqv2, indicando número de atributos de cada base, contagem de amostras TP e NT e total de amostras

Tipo de Câncer	**Nº Atributos**	**Nº TP**	**Nº NT**	**Total**
[BLCA] *Bladder Urothelial Carcinoma*	20242	408	19	427
[BRCA] *Breast Invasive Carcinoma*	20252	1097	114	1211
[HNSC] *Head and Neck Squamous Cell Carcinoma*	20264	520	44	564
[LUAD] *Lung Adenocarcinoma*	20199	515	59	574
[LUSC] *Lung Squamous Cell Carcinoma*	20245	503	51	554
[THCA] *Thyroid Carcinoma*	20531	505	59	564

Informações obtidas a partir dos dados coletados por Neto (2016)

Tabela 2 – Bases de dados miRNA-Seq, indicando número de atributos de cada base, contagem de amostras TP e NT e total de amostras

Tipo de Câncer	**Nº Atributos**	**Nº TP**	**Nº NT**	**Total**
[BLCA] *Bladder Urothelial Carcinoma*	878	409	19	428
[BRCA] *Breast Invasive Carcinoma*	898	755	87	842
[HNSC] *Head and Neck Squamous Cell Carcinoma*	904	486	44	530
[LUAD] *Lung Adenocarcinoma*	895	450	46	496
[LUSC] *Lung Squamous Cell Carcinoma*	887	342	45	387
[THCA] *Thyroid Carcinoma*	897	502	59	561

Informações obtidas a partir dos dados coletados por Neto (2016)

3.3.3 Módulos do Sistema

Essa seção descreve cada uma das etapas que constituem os experimentos realizados neste trabalho.

3.3.3.1 Informações Gerais

Para facilitar a execução dos diferentes experimentos, foi construído um *pipeline* utilizando a linguagem de programação *Python* que automatizou o processo de análise dos conjuntos de dados. Tal *pipeline* consistiu na leitura dos arquivos de dados e variação dos parâmetros de cada etapa dos experimentos, realizando assim as diversas combinações de experimentos de forma automatizada.

De forma geral, diversas bibliotecas foram empregadas nas várias etapas dos experimentos:

- *pandas*[15]: um conjunto *open source* de estruturas de dados e ferramentas de análise de dados de alto desempenho para linguagem *Python*. *pandas* foi utilizada para leitura e manipulação dos conjuntos de dados.
- *scikit-learn*[16]: uma das principais bibliotecas para AM em *Python*, oferecendo diversas ferramentas simples e eficientes para mineração e análise de dados. Utilizou-se *scikit-learn* para realização de validação cruzada, normalização de dados, classificação e construção de matrizes de confusão.
- *NumPy*[17]: é o pacote fundamental para computação científica em *Python*, dando suporte a vetores multidimensionais e realizando operações matemáticas, como operações de álgebra linear, de forma otimizada. Além de ser usado por praticamente todas as outras bibliotecas, o pacote também foi usado explicitamente para diversas operações com vetores.
- *Matplotlib*[18]: uma poderosa biblioteca em *Python* para geração de gráficos em duas dimensões. Foi utilizada para geração das figuras de matrizes de confusão e outros gráficos.

Para realização da validação cruzada, especificamente, foi utilizada a classe *StratifiedKFold*[19] da biblioteca *scikit-learn*. Essa classe foi utilizada para realizar validação cruzada de forma estratificada, mantendo a proporção de classes do conjunto completo em cada subconjunto gerado. Embora Kohavi *et al.* (1995) recomende validação cruzada estratificada com $K = 10$, foi escolhido $K = 5$ (*5-fold cross validation*) devido ao alto custo computacional que seria decorrente da utilização de mais *folds*, ainda que um valor de K maior pudesse trazer melhores resultados.

[15] http://pandas.pydata.org
[16] http://scikit-learn.org/stable/
[17] http://www.numpy.org
[18] https://matplotlib.org
[19] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html

3.3.3.2 Balanceamento de Dados

A primeira etapa de cada experimento consistiu em realizar o balancemanto dos dados analisados. Como pode-se observar na Tabela 1 e na Tabela 2, há uma grande diferença entre o número de amostras do tipo TP e o número de amostras do tipo NT das bases de dados analisadas, caracterizando-as assim como bases de dados desbalanceados. A base com maior grau de desbalanceamento foi BLCA do tipo miRNA-Seq, com razão entre amostras NT e TP igual a aproximadamente 0,0465. Já o menor grau de desbalanceamento foi observado na base LUSC do tipo miRNA-Seq, com razão entre amostras NT e TP próxima a 0,1316. Dessa forma, observa-se que o grau de desbalanceamento das bases é significativo.

Três técnicas de balanceamento de dados foram escolhidas e comparadas: SMOTE (CHAWLA *et al.*, 2002) (técnica de *oversampling*), NearMiss (MANI; ZHANG, 2003) (técnica de *undersampling*) e SMOTE+Tomek (BATISTA; PRATI; MONARD, 2004) (combinando *oversampling* e *undersampling*). Dessa forma, escolheu-se uma técnica de cada tipo para que fossem comparadas na análise dos dados de expressão gênica. Com relação à técnica NearMiss, foi escolhida a variação NearMiss-2, que no estudo realizado por Mani e Zhang (2003) obteve o melhor desempenho em comparação a outras variações.

A implementação das técnicas de balanceamento foi feita utilizando-se a biblioteca *imbalanced-learn*[20] (LEMAîTRE; NOGUEIRA; ARIDAS, 2017) para *Python*, que tem como um de seus atrativos a compatibilidade com a biblioteca *scikit-learn*.

3.3.3.3 Redução de Dimensionalidade

A segunda etapa dos experimentos teve como objetivo a redução da dimensionalidade dos dados. Dados de expressão gênica apresentam grande número de atributos e muitos dos mesmos podem ser considerados irrelevantes para o problema sendo analisado, já que apenas uma parcela dos genes influencia um fenômeno biológico específico (CALZA *et al.*, 2007 apud SOUZA, 2010). Além disso, é possível haver redundância de atributos, ocorrendo alta correlação entre genes; a eliminação de atributos redundantes pode facilitar a análise de dados (JÄGER; SENGUPTA; RUZZO, 2002 apud SOUZA, 2010).

Dessa forma, para realizar a redução de dimensionalidade dos dados, foram utilizadas duas técnicas de seleção de atributos baseadas em filtros: ReliefF (KIRA; RENDELL, 1992) (que foca na relevância estatística dos atributos) e FCBF (YU; LIU, 2003) (que foca na informação mútua entre atributos). Essas técnicas foram comparadas entre si e também com um *Stacked Denoising Autoencoder*, que pode ser utilizado para redução de dimensionalidade de dados como apresentado na seção 2.2.4.3. O número de atributos final resultante da redução de dimensionalidade também foi variado. Três opções de número final de atributos foram utilizadas: 10, 50 ou 100 atributos. Esses valores foram escolhidos tomando-se como ponto de partida o

[20] https://github.com/scikit-learn-contrib/imbalanced-learn

trabalho de Neto (2016).

No caso das técnicas de seleção de atributos, foi empregada a biblioteca *scikit-feature*[21] (LI *et al.*, 2016), compatível com *scikit-learn* e oferecendo diversas opções de técnicas de seleção de atributos.

Já com relação ao SDAE, um modelo de AP, foi utilizada a biblioteca *Keras*[22] (CHOLLET *et al.*, 2015), uma API em *Python* de alto nível para redes neurais. Altamente poderosa, essa API pode ser utilizada com diversos *backends* de redes neurais, oferecendo suporte a utilização de GPUs. O *backend* escolhido para este trabalho foi *Theano*[23] (Theano Development Team, 2016), uma biblioteca em *Python* para execução eficiente de operações matemáticas envolvendo vetores multidimensionais. Um motivo que justifica a escolha da associação entre *Keras* e *Theano* é o fato de tais bibliotecas já estarem instaladas na máquina virtual utilizada na plataforma *Azure*, sendo necessário apenas configurar a utilização da GPU por parte da biblioteca *Theano*. Além disso, a implementação do SDAE teve como inspiração o trabalho desenvolvido por Danaee, Ghaeini e Hendrix (2016), onde também foi utilizada a mesma associação entre *Keras* e *Theano*.

O SDAE construído neste trabalho é composto de quatro camadas: a primeira, que é a camada de entrada, possui a mesma dimensão que os dados de entrada (ou seja, o número de neurônios artificiais nessa camada é igual ao número original de atributos dos dados). As demais camadas possuem progressivamente menor dimensão, sendo que a última camada possui dimensão igual ao número final de atributos desejado. A razão entre a dimensão de duas camadas adjacentes é constante. A utilização de 4 camadas já representou uma rede profunda razoavelmente complexa; um maior número de camadas poderia tornar a rede ainda mais profunda, porém adicionando um alto custo computacional associado. Já um número menor de camadas tornaria os resultados menos satisfatórios.

No treinamento não supervisionado camada a camada realizado no SDAE, foi adotado o *batch size* padrão da biblioteca *Keras*, igual a 32 amostras. Já o número de épocas selecionado foi 3, embora o padrão da biblioteca seja 10. Esse valor foi escolhido para diminuir o tempo necessário para treinamento, considerando que o treinamento camada a camada com 3 épocas por camada resulta num total de 9 épocas, número próximo ao valor padrão e que seria utilizado caso o treinamento fosse realizado de uma só vez (combinando todas as camadas). Assim como indicado por Chollet (2016), o treinamento camada a camada envolveu a utilização da função de ativação *Rectified Linear Unit* para camadas codificadoras e função sigmoidal para camadas decodificadoras. A função de otimização escolhida foi ADADELTA (ZEILER, 2012), como indicado por Chollet (2016), e a função de perda foi Erro Quadrático Médio *Mean Squared Error*, utilizada em tarefas de regressão (SOUZA, 2010). O treinamento não supervisionado também contou com a introdução de ruído camada a camada para garantir a robustez do modelo

[21] https://github.com/jundongl/scikit-feature
[22] https://github.com/fchollet/keras
[23] http://deeplearning.net/software/theano/

(DANAEE; GHAEINI; HENDRIX, 2016).

Após essa etapa de treinamento não supervisionado, foi realizada a operação de *fine tuning*: uma última camada foi adicionada à rede para classificação, realizaando aprendizado supervisionado. Nessa última etapa, foi utilizado o valor padrão da biblioteca *Keras* de treinamento com 10 épocas e *batch size* de 32 amostras. Como se trata de uma etapa de classificação, a função de perda utilizada foi *binary cross entropy* (CHOLLET, 2016). Após o treinamento, o SDAE foi criado ao combinar-se a camada de entrada e as demais camadas codificadoras.

3.3.3.4 Classificação

A etapa final de cada experimento correspondeu ao treinamento de um modelo de classificação e à validação desse modelo ao classificar o conjunto de teste, verificando se os rótulos previstos correspondiam aos rótulos reais das amostras do conjunto de teste. Os dados utilizados nesta etapa estavam balanceados e com dimensionalidade reduzida, fruto da utilização das etapas anteriores.

Três classificadores diferentes foram considerados: uma rede MLP, uma rede neural convolucional e por fim uma árvore de decisão. A implementação das redes neurais artificiais (MLP e CNN) foi feita utilizando a biblioteca *Keras* e o *backend Theano*. Com relação à árvore de decisão, foi utilizada a classe *DecisionTreeClassifier*[24] oferecida pela bilioteca *scikit-learn*. Essa classe modela árvores de decisão implementadas com uma versão otimizada do algoritmo CART (Scikit-Learn Developers, 2017). Os detalhes de dos classificadores que empregam redes neurais serão apresentados a seguir. Já o classificador baseado em árvores de decisão não teve nenhum parâmetro indicado; foram utilizadas todas as configurações padrão da biblioteca *scikit-learn*.

A rede MLP foi construída tendo como base o trabalho realizado por Chen *et al.* (2016), que utilizaram redes neurais artificiais multicamadas para inferência de expressão gênica tanto em dados *Microarray* quanto RNA-Seq. Assim como em tal trabalho, a rede MLP utilizada aqui possui uma camada de entrada de dimensionalidade compatível com os dados de entrada (10, 50 ou 100 atributos), 3 camadas ocultas e uma camada de saída com apenas um neurônio artifical (usado para classificação). Foram escolhidas 3 camadas ocultas devido aos melhores resultados encontrados por Chen *et al.* (2016) com essa mesma escolha. Todos as camadas da rede foram completamente conectadas umas às outras, e todas as camadas ocultas utilizadas tiveram o mesmo número de nós que a camada de entrada. A função de ativação utilizada nas camadas ocultas foi a função tangente hiperbólica, que captura os padrões não lineares contidos nos dados (CHEN *et al.*, 2016). Na camada de saída foi utilizada a função de ativação sigmoidal, para classificação, diferentemente da função linear utilizada por Chen *et al.* (2016), que foi empregada para regressão. A inicialização dos parâmetros da rede neural (peso e viés das conexões) foi feita utilizando o método Uniforme Glorot (GLOROT; BENGIO, 2010 apud CHEN *et al.*, 2016). Também foi utilizada a técnica de *dropout* com taxa de 10%, que garantiu os melhores resultados

[24] http://scikit-learn.org/stable/modules/tree.html

para Chen *et al.* (2016). Já a função de otimização utilizada foi RMSProp, uma técnica que tem se mostrado efetiva quando utilizada em redes profundas e utilizada frequentemente por praticantes de redes neurais (GOODFELLOW; BENGIO; COURVILLE, 2016). A função de perda utilizada foi *binary cross-entropy*, recomendada para classificação binária (LECUN; RANZATO, 2013). Finalmente, o número de épocas de treinamento utilizado foi 200, assim como feito por Chen *et al.* (2016). O valor de *batch size* utilizado foi o padrão da biblioteca *Keras*: 32.

A rede CNN construída, por sua vez, não teve como base nenhum trabalho anterior, já que não foram encontrados trabalhos que utilizassem redes convolucionais para classificação de dados de expressão gênica. De fato, a escolha de uma arquitetura como CNN para classificação do tipo de dados envolvido nesse trabalho foi feita para avaliar a aptidão de tal arquitetura e, possivelmente, examinar se dados de expressão gênica apresentam alguma topologia que favoreça o uso da operação de convolução, com grupos locais de dados altamente correlacionados. Assim, muitos dos parâmetros da rede foram escolhidos iguais à rede MLP. A rede CNN foi construída com uma camada de entrada de dimensão equivalente aos dados de entrada; uma camada convolucional com função de ativação *Rectified Linear Unit*; uma camada de *pooling*; uma camada completamente conectada com dimensão correspondente aos dados de entrada e com funçao de ativação tangente hiperbólica; e uma camada de saída com um neurônio, com função de ativação sigmoidal, utilizada para classificação. Assim como para a rede MLP, foi utilizada a técnica de *dropout* com taxa de 10% após a camada completamente conectada. Além disso, os parâmetros das camadas completamente conectada e de saída também foram inicialidados com o método Uniforme Glorot; foram utilizadas 200 épocas de treinamento; foi utilizado *batch size* igual a 32 amostras; e a rede também utilizou função de perda *binary cross-entropy* e função de otimização RMSProp. Finalmente, foram selecionados, de forma arbitrária, os seguintes parâmetros para a camada de convolução: 5 filtros distintos e tamanho de *kernel* igual à metade da dimensão dos dados de entrada.

3.3.4 Métricas de Avaliação

Para analisar os resultados obtidos a partir da realização dos experimentos, foram utilizadas as seguintes ferramentas e métricas de avaliação:

- G-mean: essa métrica foi utilizada para avaliar os resultados de cada execução dos experimentos. Foi utilizada para comparar os rótulos previstos (após utilização do modelo de classificação) e os rótulos verdadeiros dos dados do conjunto de teste. Como não foi realizado balanceamento dos dados desse conjunto de teste, G-mean foi a métrica adequada. A validação cruzada envolveu o cálculo de 5 valores G-mean diferentes (um para cada *fold*). A média desses valores foi tomada como valor final.

- Matriz de confusão: para auxiliar na interpretação dos valores G-mean, foi gerada a matriz de confusão para cada experimento. Mais especificamente, foi escolhido o *fold* cujo valor

G-mean correspondeu à mediana para cada experimento; a partir desse *fold* foi computada a matriz de confusão. O modelo referente à mediana foi escolhido na tentativa de capturar as características que melhor representam os resultados do experimento.

- Também foi gerada a visualização gráfica das árvores de decisão, utilizando mais uma vez o modelo correspondente ao *fold* cujo valor G-mean foi a mediana. As árvores só foram geradas quando utilizou-se os métodos de seleção de atributos, já que o SDAE não conserva os atributos originais.

- Finalmente, foram comparados os atributos selecionados com as técnicas ReliefF e FCBF. Isso foi feito para cada combinação de técnica de balanceamento e modelo de classificação, para cada um dos *folds* da validação cruzada, considerando apenas o caso de 10 atributos selecionados (facilitando assim a visualização dos resultados).

3.4 Resultados Obtidos

Nesta seção serão discutidos os resultados dos 81 experimentos realizados para cada uma das bases RNA-Seqv2 e miRNA-Seq. Para auxiliar na análise e interpretação desses resultados, foi utilizado um *script* escrito na linguagem *Python* para, por exemplo, extrair os melhores e piores resultados de experimentos de cada base e também gerar gráficos, facilitando a visualização dos resultados. O *script* em questão e todos os resultados dos experimentos podem ser acessados em <https://github.com/henrisilver/DeepLearningCancer>. Na comparação dos resultados, a precisão máxima considerada foi de $0,0001$, assim como feito por Neto (2016), já que é possível haver grande semelhança entre os resultados produzidos por diferentes experimentos.

A Tabela 3 e a Tabela 4 apresentam os dados referentes ao melhor desempenho e pior desempenho encontrados para os conjuntos de dados RNA-Seqv2 e miRNA-Seq, respectivamente. A combinação de parâmetros de cada experimento (Método de Seleção de Atributos, Número de Atributos Selecionados, Método de Balanceamento e Classificador) é indicada. Aqui, por simplicidade, utiliza-se o termo "Seleção de Atributos"para indicar também o método SDAE. O melhor desempenho obtido para cada conjunto de dados também é comparado percentualmente com o pior resultado e com o primeiro quartil, mediana, média e terceiro quartil. Essa comparação foi feita utilizando a diferença percentual. Dados dois valores x e y, a diferença percentual é definida por:

$$\text{Diferença Percentual} = \frac{|x-y|}{|x+y|/2} \times 100\%$$

A Figura 14 sintetiza as informações apresentadas na Tabela 3 e na Tabela 4, exibindo de forma gráfica o melhor e o pior resultado para cada base de dados. No eixo das abscissas, temos dispostos os melhores e piores resultados para as diferentes bases. No eixo das ordenadas, temos o valor G-Mean associado.

Tabela 3 – Melhores e piores resultados obtidos para cada conjunto de dados RNA-Seqv2, incluindo comparações dos melhores resultados

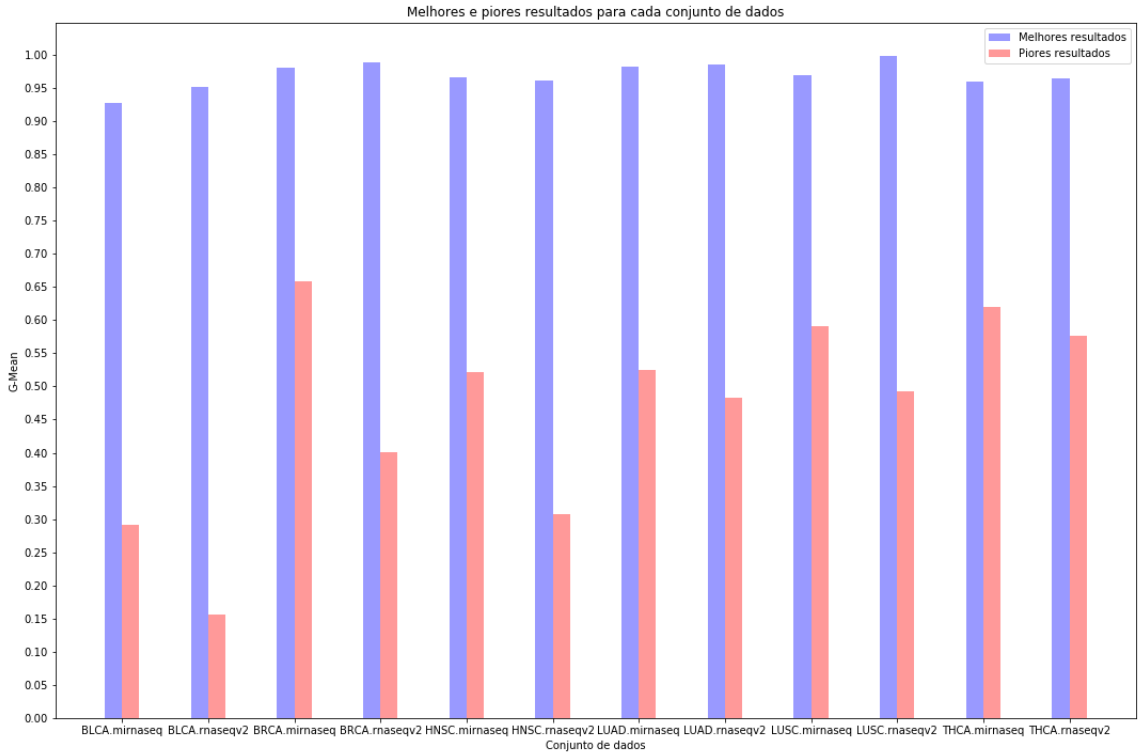
							Dif. % Melhor				
Dataset	Tipo	G-Mean	Selec. Atr.	# Atr.	Balanc.	Classif.	Pior	1º Qt	Mediana	Média	3º Qt
BLCA - RNA-Seqv2	Melhor	0,9510	ReliefF	10	SMOTE	CNN	143,3213	35,4689	13,8520	23,2592	7,8234
	Pior	0,1570	FCBF	10	NearMiss	CNN	-	-	-	-	-
BRCA - RNA-Seqv2	Melhor	0,9885	ReliefF	50	NearMiss	CNN	84,6242	27,6470	16,5982	18,2434	4,9767
	Pior	0,4007	SDAE	10	NearMiss	DecisionTree	-	-	-	-	-
HNSC - RNA-Seqv2	Melhor	0,9601	ReliefF	100	SMOTETomek	MLP	102,8229	20,8015	14,8596	17,2721	5,9637
	Pior	0,3081	FCBF	50	NearMiss	CNN	-	-	-	-	-
LUAD - RNA-Seqv2	Melhor	0,9856	ReliefF	100	SMOTE	MLP	68,3913	24,6894	5,6869	13,3738	2,0190
	Pior	0,4833	FCBF	50	NearMiss	CNN	-	-	-	-	-
LUSC - RNA-Seqv2	Melhor	0,9980	ReliefF	100	NearMiss	CNN	67,8835	16,2631	7,3659	11,2592	2,4751
	Pior	0,4922	SDAE	100	NearMiss	DecisionTree	-	-	-	-	-
THCA - RNA-Seqv2	Melhor	0,9642	ReliefF	100	SMOTE	MLP	50,4904	17,9352	11,5866	12,7923	4,0851
	Pior	0,5755	FCBF	100	NearMiss	CNN	-	-	-	-	-

Tabela 4 – Melhores e piores resultados obtidos para cada conjunto de dados miRNA-Seq, incluindo comparações dos melhores resultados

							Dif. % Melhor				
Dataset	Tipo	G-Mean	Selec. Atr.	# Atr.	Balanc.	Classif.	Pior	1º Qt	Mediana	Média	3º Qt
BLCA - miRNA-Seq	Melhor	0,9277	SDAE	100	SMOTE	CNN	104,4634	32,5500	17,4000	23,2141	10,4941
	Pior	0,2911	FCBF	50	NearMiss	CNN	-	-	-	-	-
BRCA - miRNA-Seq	Melhor	0,9806	ReliefF	50	SMOTE	MLP	39,4189	20,2111	13,2550	13,3135	4,4194
	Pior	0,6577	SDAE	50	SMOTE	DecisionTree	-	-	-	-	-
HNSC - miRNA-Seq	Melhor	0,9660	ReliefF	100	SMOTETomek	MLP	59,6774	25,3061	13,9061	16,1125	6,5647
	Pior	0,5220	FCBF	50	NearMiss	CNN	-	-	-	-	-
LUAD - miRNA-Seq	Melhor	0,9820	ReliefF	100	SMOTETomek	MLP	60,7195	36,5268	11,7749	18,4587	6,5955
	Pior	0,5246	FCBF	100	NearMiss	CNN	-	-	-	-	-
LUSC - miRNA-Seq	Melhor	0,9684	ReliefF	100	SMOTE	MLP	48,4351	25,3403	9,9171	14,2591	4,2392
	Pior	0,5908	FCBF	100	NearMiss	CNN	-	-	-	-	-
THCA - miRNA-Seq	Melhor	0,9594	ReliefF	100	SMOTE	MLP	42,9630	19,3290	13,4253	13,3604	6,2006
	Pior	0,6201	FCBF	10	NearMiss	MLP	-	-	-	-	-

É notável a diferença de desempenho entre a melhor e a pior configuração de experimento em todos os casos. Em especial, a maior diferença entre a melhor e a pior configuração aconteceu para a base BLCA do conjunto RNA-Seqv2 e, como indicado na Tabela 3, essa diferença foi de mais de 143%. Curiosamente, nessa base, tanto o melhor quanto o pior experimento utilizaram o classificador CNN e tiveram o mesmo número de atributos selecionados (10). A diferença ficou por conta da técnica de seleção de atributos e pelo método de balanceamento dos dados. O melhor experimento utilizou ReliefF e SMOTE, respectivamente. Já o pior utilizou FCBF e NearMiss. Se considerarmos que de todas as bases RNA-Seqv2 a base BLCA é a que possui menor número de amostras (427, como indica a Tabela 1), faz sentido que uma técnica de *undersampling* como NearMiss traga resultados ruins, removendo amostras de um conjunto que já apresenta escassez de exemplos. De fato, todas as bases utilizadas apresentam relativa escassez de exemplos, já que uma das limitações do AP é a necessidade de grande quantidade de exemplos, como será apresentado na Seção 3.5, o que justifica o fato de a técnica NearMiss em 11 dos 12 piores resultados. O único pior resultado que não utiliza NearMiss é para a base BRCA do conjunto

Figura 14 – Síntese dos melhores e piores resultados dos experimentos com bases RNA-Seqv2 e miRNA-Seq



miRNA-Seq, que apresenta um número mais elevado de atributos e representa a menor diferença entre o melhor e pior resultado (pouco mais de 39%).

Também percebe-se que em 9 dos 12 conjuntos de dados o pior resultado utilizou a técnica de seleção de atributos FCBF. Inclusive o pior de todos experimentos, realizado com a base BLCA do conjunto RNA-Seqv2, que obteve apenas 0,1570 de G-Mean, utilizou a técnica FCBF. A matriz de confusão para esse experimento é apresentada na Figura 15 e, como pode-se perceber, todas as amostras da classe minoritária (NT) foram classificadas de forma errada (classificadas como TP). Isso contribuiu muito para o péssimo resultado do experimento e pode ser um indício de que a técnica FCBF não conseguiu selecionar atributos que conferissem representatividade para a classe minoritária. Verificou-se também que à medida que o número selecionado de atributos aumentava, mantendo-se os demais parâmetros, os resultados tornavam-se ligeiramente melhores, como indicado na Figura 16. Embora o número de falsos negativos tenha aumentado muito, o resultado G-Mean para tal experimento já foi maior (0,3472) pelo fato de o classificador não errar a previsão de todos os rótulos da classe minoritária. Isso pode indicar que a técnica FCBF consegue escolher um subconjunto de atributos mais relevantes quando considera-se um número maior de atributos.

Analisando em mais detalhes os resultados com utilização da técnica FCBF, percebeu-se que as árvores de decisão geradas tornam-se demasidamente complexas e grandes em comparação ao uso de ReliefF, o que pode indicar *overfitting* da árvore. Por exemplo, a Figura 17 apresenta a árvore de decisão gerada para a base de dados HNSC do conjunto miRNA-Seq utilizando-se

Figura 15 – Matriz de Confusão para o experimento com pior resultado, que utilizou FCBF selecionando 10 atributos, NearMiss e CNN

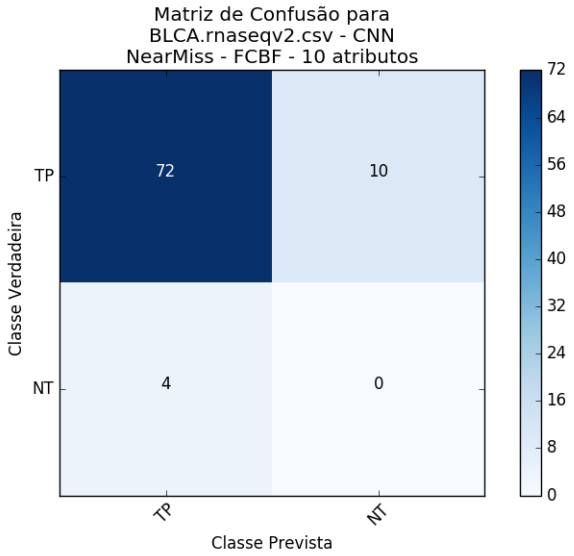
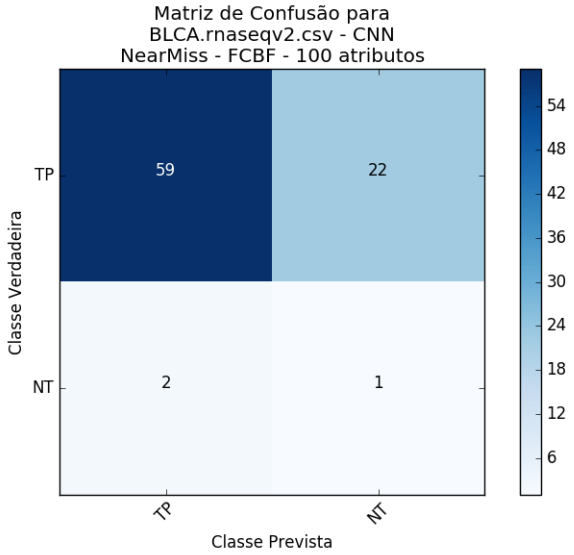


Figura 16 – Matriz de Confusão indicando melhor desempenho com a técnica FCBF ao aumentar o número de atributos selecionados, mantendo-se constantes os demais parâmetros



FCBF selecionando 100 atributos e SMOTE+Tomek. Em contraste, a Figura 18 apresenta a árvore de decisão gerada para o mesmo conjunto de dados utilizando ReliefF e mantendo-se todos os demais parâmetros constantes. Essa notável diferença reforça a ideia de que o método ReliefF consegue selecionar um melhor conjunto de atributos que a técnica FCBF.

Tendo isso em vista, decidiu-se comparar os atributos sendo selecionados por ambas as técnicas ReliefF e FCBF. Constatou-se que na grande maioria dos casos não há atributos em comum na seleção feita por cada um dos métodos; os conjuntos de atributos são mutuamente exclusivos, como ilustrado pela Figura 19, comparando atributos selecionados pelas técnicas FCBF e ReliefF para a base LUSC do conjunto RNA-Seqv2, utilizando MLP e SMOTE. Mesmo

Figura 17 – Árvore de decisão extremamente complexa, indicando *overfitting*, gerada utilizando a técnica FCBF para a base de dados HNSC do conjunto miRNA-Seq

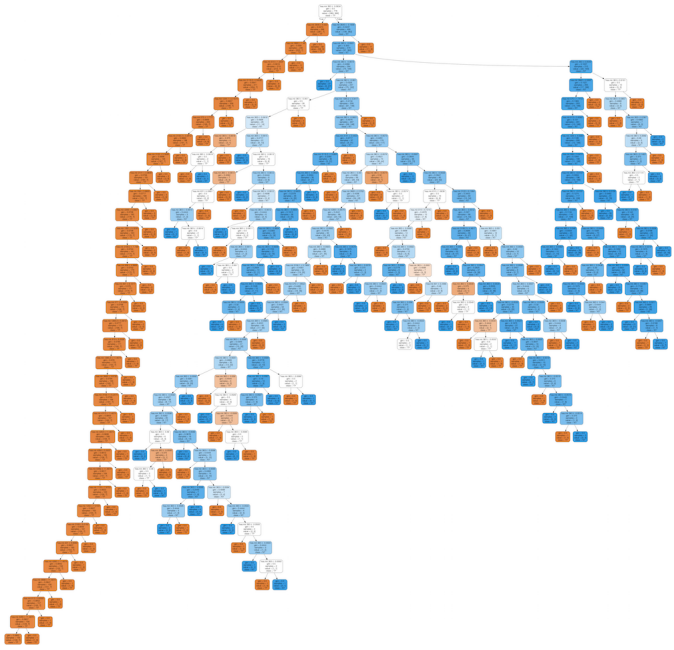
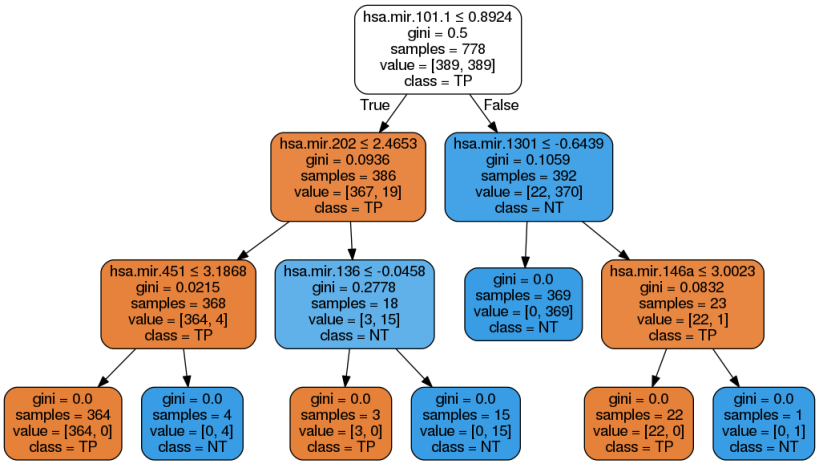


Figura 18 – Árvore de decisão gerada utilizando a técnica ReliefF para a base de dados HNSC do conjunto miRNA-Seq



quando há sobreposição de conjuntos, são poucos os atributos compartilhados pelos métodos de seleção, em geral apenas um. E curiosamente, em alguns casos, o algoritmo FCBF seleciona um número menor de atributos do que o desejado, como indica a Figura 20. Embora improvável, isso levou a ser considerada possibilidade de algum erro de implementação da biblioteca utilizada.

Vale ressaltar também que alguns experimentos alcançaram resultados excelentes, com mais de 0,98 de G-Mean, como por exemplo as bases BRCA, LUAD e LUSC do conjunto RNA-Seqv2 e BRCA e LUAD do conjunto miRNA-Seq. Além de essas bases apresentarem um relativo maior número de amostras, nota-se que todas utilizaram o método ReliefF para seleção de atributos. Também percebeus-se que em nenhum desses casos se utilizou o classificador DecisionTree. O motivo disso pode ser uma falta de otimização do mesmo, já que não foi

Figura 19 – Comparação entre atributos selecionados pelas técnicas FCBF e ReliefF para a base LUSC do conjunto RNA-Seqv2, utilizando MLP e SMOTE. Não houve sobreposição nos conjuntos de atributos selecionados

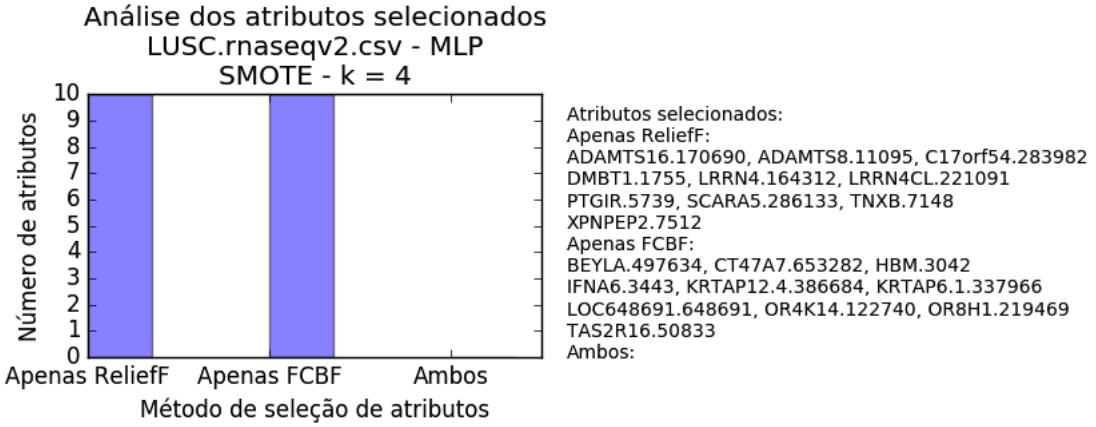
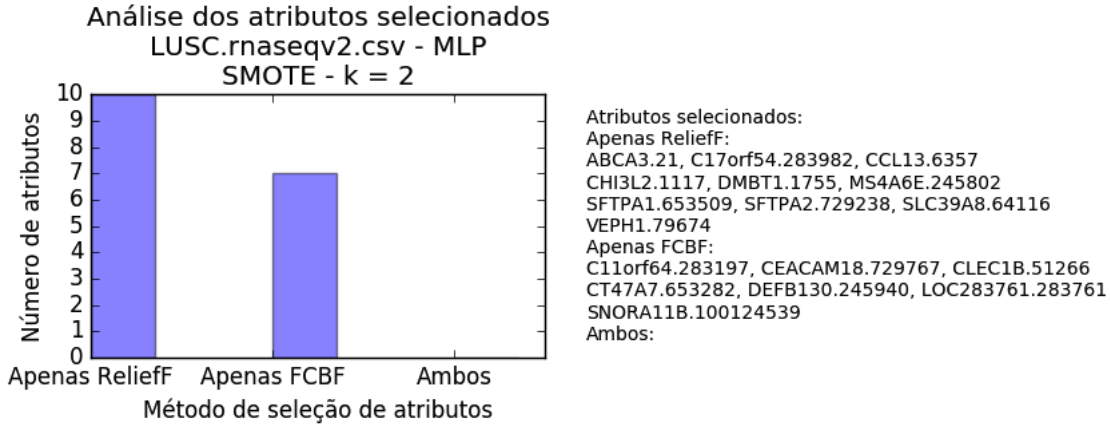
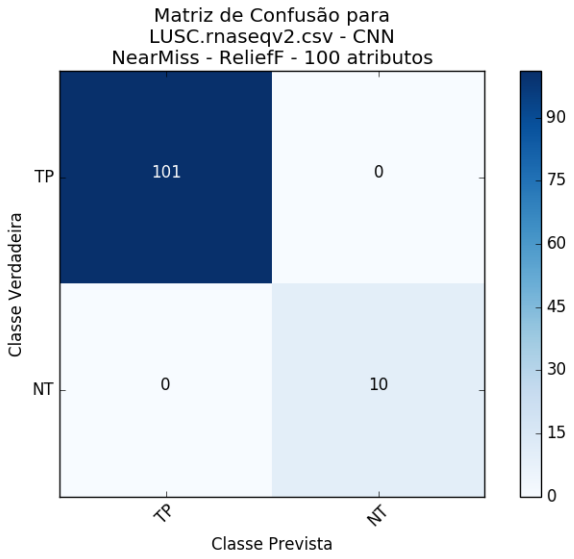


Figura 20 – Comparação entre atributos selecionados pelas técnicas FCBF e ReliefF para a base LUSC do conjunto RNA-Seqv2, utilizando MLP e SMOTE. No fold k = 2, a técnica FCBF selecionou apenas 7 atributos



feita nenhuma configuração para sua utilização. Além disso, a implementação de DecisionTree utilizada não apresenta suporte a poda de árvores, uma limitação que será discutida na Seção 3.5. Isso pode ter prejudicado o desempenho desse classificador, enquanto os classificadores MLP e CNN foram construídos com diversas otimizações e camadas de neurônios, permitindo maior número de graus de abstração. Além disso, em nenhum dos melhores resultados se utilizou apenas 10 atributos selecionados. Isso pode indicar que são necessários mais de 10 atributos para obter uma representação adequada da situação. O melhor de todos os resultados, 0,9980, ocorreu para a base LUSC do conjunto RNA-Seqv2, que curiosamente utilizou classificador CNN e NearMiss como método de balanceamento, além de ReliefF selecionando 100 atributos. A Figura 21 mostra a matriz de confusão de tal experimento. Nota-se que todos os exemplos do conjunto de teste foram classificados corretamente, mas o resultado foi abaixo de 1,0 já que a métrica G-Mean utilizada na Tabela 3 e na Tabela 3 diz respeito à média decorrente dos *folds* da validação cruzada e a matriz de confusão representada diz respeito ao *fold* da mediana apenas.

Figura 21 – Matriz de confusão do melhor resultado dentre todos os experimentos realizados



A fim de analisar individualmente a influência de cada parâmetro nos resultados, foi realizada a média de todas as ocorrências das três opções de cada parâmetro para cada conjunto de dados. A Tabela 5 apresenta a comparação dos métodos de balanceamento; a Tabela 6 apresenta a comparação das técnicas de redução de dimensionalidade; a Tabela 7 compara as opções de número de atributos selecionados; e a Tabela 8 compara os classificadores. A Figura 22, a Figura 23, a Figura 24 e a Figura 25 sintetizam, respectivamente, as informações da Tabela 5, da Tabela 6, da Tabela 7 e da Tabela 8.

Tabela 5 – Comparação entre métodos de balanceamento para todos os conjuntos de dados

Dataset	Melhor	Mediano	Pior	Dif.% Mel-Med	Dif.% Mel-Pio
BLCA - miRNA-Seq	SMOTETomek	SMOTE	NearMiss	1,6818	33,9892
BLCA - RNA-Seqv2	SMOTE	SMOTETomek	NearMiss	2,6834	23,1771
BRCA - miRNA-Seq	SMOTETomek	SMOTE	NearMiss	0,9221	2,2161
BRCA - RNA-Seqv2	SMOTETomek	SMOTE	NearMiss	0,0859	6,6185
HNSC - miRNA-Seq	SMOTE	SMOTETomek	NearMiss	1,4123	7,941
HNSC - RNA-Seqv2	SMOTE	SMOTETomek	NearMiss	2,2348	16,3132
LUAD - miRNA-Seq	SMOTE	SMOTETomek	NearMiss	0,6061	6,7734
LUAD - RNA-Seqv2	SMOTE	SMOTETomek	NearMiss	0,0797	9,0182
LUSC - miRNA-Seq	SMOTETomek	SMOTE	NearMiss	1,0772	8,2833
LUSC - RNA-Seqv2	SMOTE	SMOTETomek	NearMiss	1,1185	10,6759
THCA - miRNA-Seq	SMOTETomek	SMOTE	NearMiss	0,8683	5,1301
THCA - RNA-Seqv2	SMOTE	SMOTETomek	NearMiss	0,3401	8,3869

Figura 22 – Comparação entre os diferentes métodos de balancemaneto para todos os conjuntos de dados

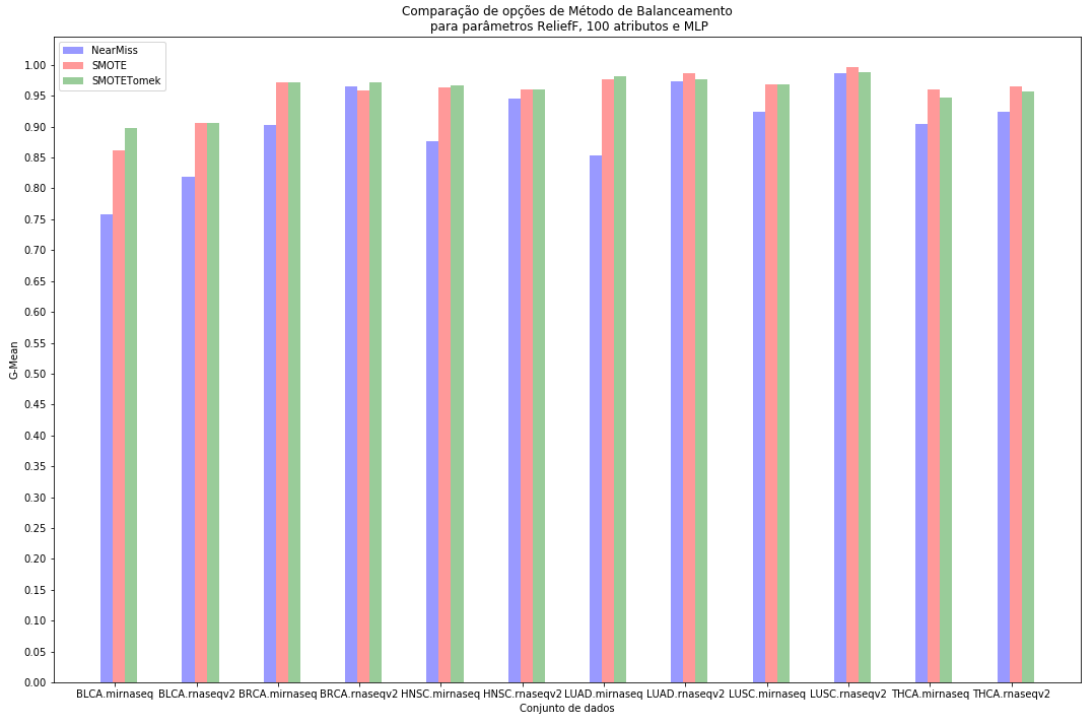


Tabela 6 – Comparação entre técnicas de redução de dimensionalidade para todos os conjuntos de dados

Dataset	Melhor	Mediano	Pior	Dif.% Mel-Med	Dif.% Mel-Pio
BLCA - miRNA-Seq	ReliefF	SDAE	FCBF	4,2737	18,1497
BLCA - RNA-Seqv2	ReliefF	FCBF	SDAE	15,4684	21,3476
BRCA - miRNA-Seq	ReliefF	SDAE	FCBF	9,3957	19,1961
BRCA - RNA-Seqv2	ReliefF	SDAE	FCBF	17,9873	27,8227
HNSC - miRNA-Seq	ReliefF	SDAE	FCBF	8,293	22,6124
HNSC - RNA-Seqv2	ReliefF	SDAE	FCBF	8,7669	27,1352
LUAD - miRNA-Seq	ReliefF	SDAE	FCBF	8,3974	32,5103
LUAD - RNA-Seqv2	ReliefF	SDAE	FCBF	8,0569	25,8531
LUSC - miRNA-Seq	ReliefF	SDAE	FCBF	9,0976	23,3208
LUSC - RNA-Seqv2	ReliefF	SDAE	FCBF	9,9133	13,7892
THCA - miRNA-Seq	ReliefF	SDAE	FCBF	6,2967	13,3278
THCA - RNA-Seqv2	ReliefF	SDAE	FCBF	7,8718	19,2751

Figura 23 – Comparação entre as diferentes técnicas de redução de dimensionalidade para todos os conjuntos de dados

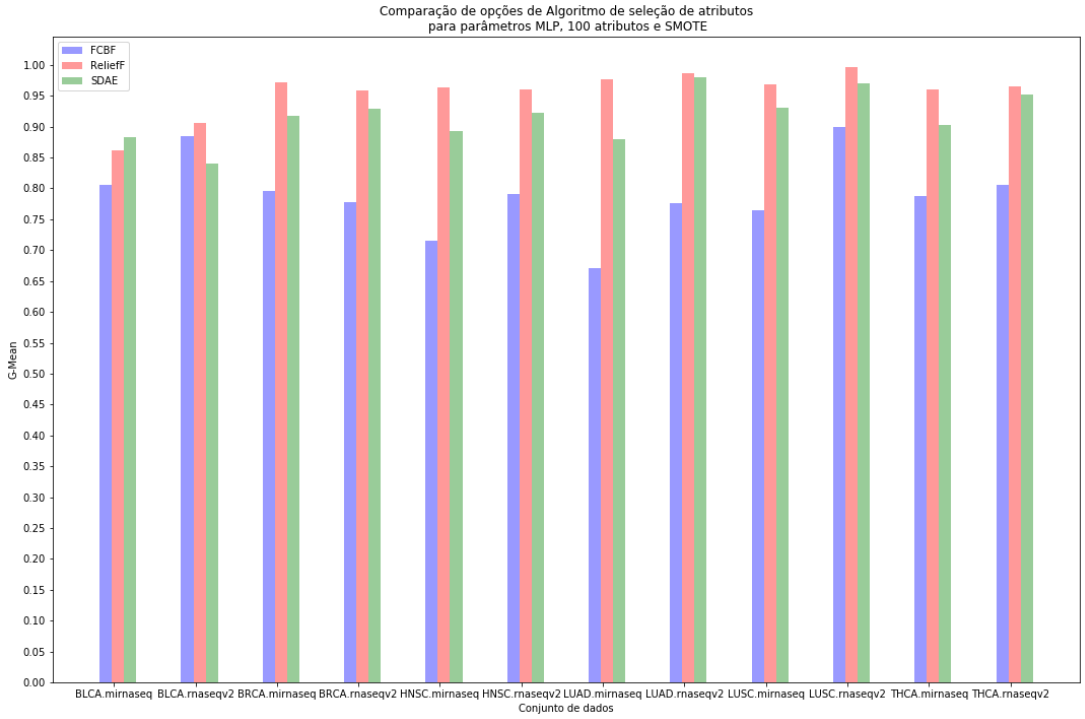


Tabela 7 – Comparação entre opções de número de atributos selecionados para todos os conjuntos de dados

Dataset	Melhor	Mediano	Pior	Dif.% Mel-Med	Dif.% Mel-Pio
BLCA - miRNA-Seq	10	50	100	0,1819	1,029
BLCA - RNA-Seqv2	100	50	10	1,4092	4,8788
BRCA - miRNA-Seq	10	50	100	1,5495	1,6554
BRCA - RNA-Seqv2	50	100	10	0,4069	6,1235
HNSC - miRNA-Seq	10	50	100	1,1251	2,0201
HNSC - RNA-Seqv2	100	50	10	0,1713	2,8687
LUAD - miRNA-Seq	10	100	50	0,4738	0,8268
LUAD - RNA-Seqv2	100	50	10	0,029	1,7722
LUSC - miRNA-Seq	100	50	10	0,8745	2,529
LUSC - RNA-Seqv2	100	10	50	0,0361	0,6524
THCA - miRNA-Seq	100	50	10	0,8295	2,7308
THCA - RNA-Seqv2	10	100	50	0,1288	2,1242

Figura 24 – Comparação entre as diferentes opções de número de atributos selecionados para todos os conjuntos de dados

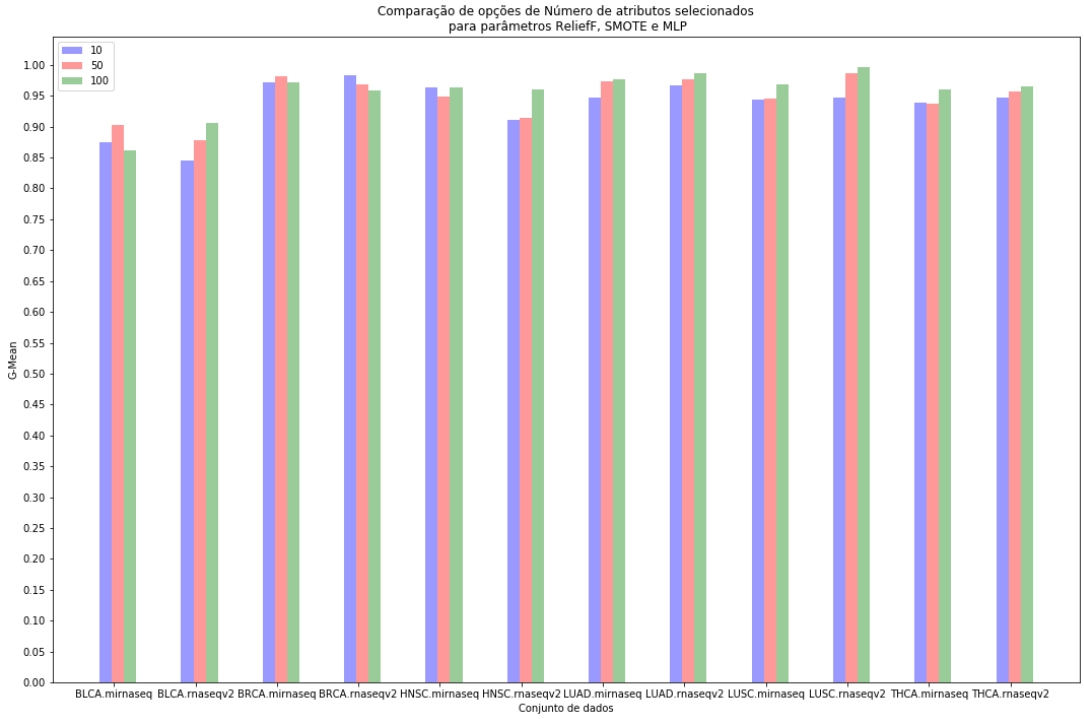
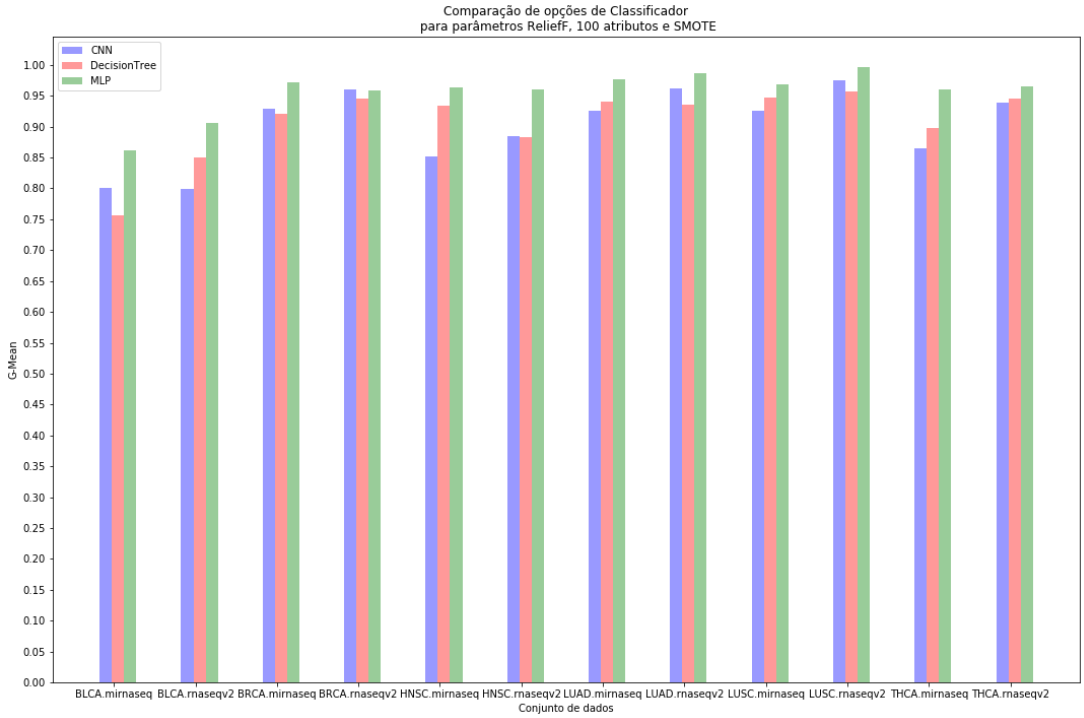


Tabela 8 – Comparação entre opções de número de atributos selecionados para todos os conjuntos de dados

Dataset	Melhor	Mediano	Pior	Dif.% Mel-Med	Dif.% Mel-Pio
BLCA - miRNA-Seq	MLP	CNN	DecisionTree	3,5949	8,003
BLCA - RNA-Seqv2	MLP	CNN	DecisionTree	4,6045	12,056
BRCA - miRNA-Seq	MLP	CNN	DecisionTree	1,0199	7,6792
BRCA - RNA-Seqv2	MLP	CNN	DecisionTree	3,5701	7,7104
HNS - miRNA-Seq	MLP	DecisionTree	CNN	0,1248	6,6009
HNSC - RNA-Seqv2	MLP	DecisionTree	CNN	1,4289	5,4501
LUAD - miRNA-Seq	MLP	DecisionTree	CNN	1,3936	2,6895
LUAD - RNA-Seqv2	MLP	CNN	DecisionTree	3,026	4,108
LUSC, - miRNA-Seq	MLP	DecisionTree	CNN	2,1317	4,7594
LUSC - RNA-Seqv2	MLP	CNN	DecisionTree	2,2261	5,7868
THCA - miRNA-Seq	MLP	CNN	DecisionTree	1,1681	7,4699
THCA - RNA-Seqv22	MLP	DecisionTree	CNN	3,0317	3,5587

Figura 25 – Comparação entre os diferentes classificadores para todos os conjuntos de dados



A partir desses dados, pode-se concluir que:

- O classificador MLP fornece os melhores resultados. Já o classificador CNN também apresentou, surpreendentemente, bom desempenho, indicando a possibilidade de os dados de expressão gênica favorecerem o uso da operação de convolução.
- O método ReliefF foi em média o melhor para redução de dimensionalidade, enquanto FCBF foi a pior técnica. O desempenho do SDAE também esteve muito próximo do desempenho de ReliefF na maioria dos casos, indicando grande potencial.
- Um maior número de atributos após redução de dimensionalidade garante melhor representação dos dados. Mesmo assim, os resultados para 10, 50 e 100 atributos foram todos próximos, o que sugere que esse não foi um fator de tanta importância nos resultados.
- As técnicas SMOTE e SMOTE+Tomek tiverem desempenho similar, sendo no geral as melhores opções para balanceamento de dados, já que a técnica NearMiss reduz o número já pequeno de amostras dos conjuntos de dados
- Conjuntos de dados: em geral, quanto mais amostras, melhores resultados são possíveis.

3.5 Dificuldades e Limitações

Essa seção descreve os desafios e limitações encontrados na realização do trabalho.

Uma grande dificuldade diz respeito ao tempo que foi necessário para realizar experimentos e obter resultados, especialmente no caso das bases de dados RNA-Seqv2. Mesmo com a poderosa infraestrutura computacional utilizada, o processamento de cada base levou horas para ser concluído. Dessa forma, ter de repetir experimentos mostrou-se algo muito custoso. O número de *folds* utilizado na validação cruzada também precisou ser restringido para não elevar ainda mais o tempo necessário para análise dos dados, embora um maior número de *folds* pudesse fornecer melhores resultados.

Com relação à infraestrutura computacional utilizada, houve diversas tentativas frustradas de se preparar um ambiente computacional favorável à realização dos experimentos antes de se utilizar a plataforma *Azure*. Primeiramente, antes de se detectar a necessidade de uma infraestrutura mais poderosa, tentou-se utilizar o próprio computador pessoal do autor. Logo verificou-se que a memória RAM disponível era insuficiente e, além disso, o tempo de execução de cada experimento seria enorme. Após isso, tentou-se utilizar um outro computador pessoal que, dessa vez, possuía uma GPU NVIDIA GeForce. Para utilizá-la, configurou-se o sistema com as bibliotecas CUDA necessárias e instalou-se as linguagens de programação e bibliotecas empregadas no experimento. Entretanto, os recursos computacionais ainda mostraram-se insuficientes, notadamente a quantidade de memória da GPU (apenas 1GB). Decidiu-se então utilizar o *cluster Euler*[25] do Centro de Ciências Matemáticas Aplicadas à Indústria da Universidade de São Paulo. Infelizmente, no período em que elaborou-se este trabalho, a criação de novas contas de usuário em tal *cluster* foi suspensa para instalação de novos nós de processamento. Assim, a ideia de se utilizar computação em nuvem ganhou força. Num primeiro momento, pensou-se em utilizar os serviços *Amazon Web Services*[26]. Porém, após a criação de uma nova conta para utilização do serviço, não foi liberado o acesso a máquinas que possuíssem GPUs. Por fim, recorreu-se à plataforma *Azure*, que felizmente pôde ser utilizada com sucesso.

Entretanto, devido às características da máquina virtual utilizada na plataforma *Azure*, algumas configurações de bibliotecas tiveram de ser ajustadas, como no caso da biblioteca *Matplotlib*. Por padrão, essa biblioteca utiliza um *backend* relacionado ao *display* gráfico da unidade. No caso da máquina virtual esse *display* estava indisponível, sendo necessário assim configurar a biblioteca para utilizar um *backend* diferente.

Outra dificuldade refere-se à utilização da distribuição Intel da linguagem *Python*[27]. Essa distribuição tem como foco o desempenho computacional. Entretanto, essa distribuição *Python* não é compatível com importantes bibliotecas utilizadas nos experimentos do trabalho. Sua utilização poderia ter reduzido o tempo necessário para processamento das bases de dados.

Além disso, não foram feitos ajustes nos parâmetros dos algoritmos utilizados nas diversas etapas dos experimentos. Por exemplo, os classificadores não tiveram seus parâmetros

[25] https://sites.google.com/site/clustercemeai/
[26] https://aws.amazon.com
[27] https://software.intel.com/en-us/distribution-for-python

ajustados após as fases de treinamento. Além disso, muitas vezes foram utilizados valores padrão para esses parâmetros. Se ajustes de parâmetros fossem realizados, é possível que o desempenho obtido nos diversos experimentos tivesse sido significativamente melhor.

Uma limitação foi observada no uso das árvores de decisão da biblioteca *scikit-learn*: a biblioteca, no momento, não oferece suporte para a realização de podas nas árvores. Assim, não é possível lidar com o problema de *overfitting* ao qual estão sujeitas as árvores de decisão (Scikit-Learn Developers, 2017).

As bases de dados também representaram desafios: a alta dimensionalidade e o desbalanceamento dos dados foram combatidos, respectivamente, com técnicas de redução de dimensionalidade e técnicas de balanceamento. Entretanto, o pequeno número de amostras em cada base de dados é um fator que possivelmente limitou o desempenho dos experimentos, dado que técnicas de AP geralmente necessitam de bases de dados com grande volume de amostras (DANAEE; GHAEINI; HENDRIX, 2016).

3.6 Considerações Finais

Neste capítulo, foram apresentados os detalhes deste projeto. incluindo a explicação dos objetivos do trabalho, a discussão da metodologia utilizada para realização dos diversos experimentos e a análise dos resultados obtidos.

No próximo capítulo, será realizada a conclusão do trabalho, apresentando uma síntese de tudo que foi feito, além de uma análise sobre experiências e conhecimento adquiridos no processo de elaboração do projeto. Também serão feitas considerações acerca do curso de graduação em Engenharia de Computação e de sua relação com este trabalho. Por fim, serão discutidas possíveis atividades a serem realizadas em trabalhos futuros.

Capítulo 4

CONCLUSÃO

Neste capítulo serão apresentados um sumário sobre as atividades desempenhadas neste trabalho e uma discussão das experiências e conhecimentos adquiridos durante sua elaboração. Serão também feitas considerações sobre curso de graduação em Engenharia de Computação e de sua relação com este trabalho, além de discutidas possíveis atividades futuras relacionadas ao tema deste projeto.

4.1 Contribuições

A grande contribuição deste trabalho foi investigar empiricamente o potencial de técnicas de Aprendizado Profundo na análise de dados de expressão gênica, em tarefas de classificação como por exemplo o diagnóstico de câncer. Além de comparar as técnicas de Aprendizado Profundo em tarefas de classificação ao utilizar de redes MLP e CNN, este trabalho também avaliou o desempenho de um SDAE na redução de dimensionalidade de dados com alto número de atributos, comparando tal desempenho com métodos de seleção de atributos.

Conforme visto no Capítulo 3, algoritmos de AP mostraram-se adequados para análise dos dados utilizados. O classificador MLP apresentou a melhor média de resultados, e o classificador CNN também apresentou resultados razoáveis, o que foi surpreendente, inclusive fornecendo os melhores resultados para algumas bases de dados. Isso pode sugerir que os dados de expressão gênica utilizados podem possuir alguma topologia que favoreça a utilização de redes convolucionais. A redução de dimensionalidade utilizando SDAE também teve bom desempenho, muitas vezes próximo do desempenho da técnica ReliefF.

- O classificador MLP fornece os melhores resultados. Já o classificador CNN também apresentou, surpreendentemente, bom desempenho, indicando a possibilidade de os dados de expressão gênica favorecerem o uso da operação de convolução.
- O método ReliefF foi em média o melhor para redução de dimensionalidade, enquanto FCBF foi a pior técnica. O desempenho do SDAE também esteve muito próximo do desempenho de ReliefF na maioria dos casos, indicando grande potencial.
- Um maior número de atributos após redução de dimensionalidade garante melhor representação dos dados. Mesmo assim, os resultados para 10, 50 e 100 atributos foram todos próximos, o que sugere que esse não foi um fator de tanta importância nos resultados.

- As técnicas SMOTE e SMOTE+Tomek tiverem desempenho similar, sendo no geral as melhores opções para balanceamento de dados, já que a técnica NearMiss reduz o número já pequeno de amostras dos conjuntos de dados
- Conjuntos de dados: em geral, quanto mais amostras, melhores resultados são possíveis.

O trabalho também representou uma rica experiência de aprendizado ao autor. Primeiramente, foi possível aprimorar as habilidades de escrita científica e de elaboração de textos técnicos, inclusive no que tange à ferramenta LaTeX, utilizada na escrita deste documento. Também foi possível praticar a pesquisa científica ao serem pesquisados e utilizados livros técnicos, artigos científicos e periódicos, além da utilização do Sistema Integrado de Bibliotecas da Universidade de São Paulo. Com relação ao tema do trabalho, o autor pôde adquirir valiosos conhecimentos da área de Aprendizado de Máquina, tema muito relevante no cenário tecnológico atual, tanto no mercado de trabalho quanto na academia. E por fim foi gratificante poder concluir o curso de graduação em Engenharia de Computação contribuindo para um tema tão relevante para a sociedade como diagnóstico de câncer.

4.2 Relação entre o curso de Engenharia de Computação e o Projeto

O curso de graduação em Engenharia de Computação oferecido em parceria pela Escola de Engenharia de São Carlos e pelo Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo ofereceu grande parte das ferramentas e do conhecimento necessários para elaboração deste trabalho.

As disciplinas *Introdução à Ciência da Computação I e II* e também *Algoritmos e Estruturas de Dados I e II* forneceram o alicerce central necessário para realização do projeto. Esse alicerce envolve, por exemplo, lógica de programação e também o conhecimento sobre estruturas de dados de diferentes complexidades, conhecimentos que foram fundamentais na elaboração do código desenvolvido para realização dos experimentos do trabalho. As disciplinas de *Sistemas Computacionais Distribuídos*, *Computação Distribuída* e *Programação Concorrente* também foram importantes ao fornecerem o conhecimento técnico necessário para lidar com uma plataforma de computação em nuvem como *Azure* e também com a utilização de GPUs na paralelização do treinamento de redes neurais artificiais. A disciplina de *Engenharia de Software* teve grande importãcia ao apresentar ao autor ferramentas de controle de versão como *git*, utilizada ao longo do projeto. A disciplina *Estatística* forneceu as ferramentas matemáticas utilizadas para análise de resultados. Finalmente, a disciplina de *Inteligência Artificial* foi essencial ao ser o primeiro contato do autor com Aprendizado de Máquina, instigando assim a vontade de ir além e elaborar um Trabalho de Conclusão de Curso na área.

4.3 Considerações sobre o Curso de Graduação em Engenharia de Computação

A experiência oferecida pelo curso de graduação em Engenharia de Computação é inegavelmente muito rica. Combinando aspectos de Engenharia Elétrica e Ciência da Computação, o curso oferece ao aluno diversas possibilidades e opções de especialização. A grade curricular do curso é muito completa, fornecendo forte base matemática aos estudantes. Além disso, a Universidade de São Paulo oferece uma ótima estrutura e um excelente ambiente para realização das atividades universitárias, considerando a qualidade de instalações como salas de aula, laboratórios e bibliotecas. Finalmente, um dos maiores atrativos do curso (e da Universidade em si) é o oferecimento de múltiplas oportunidades aos alunos, desde a participação em grupos extracurriculares à realização de projetos de iniciação científica, oportunidades de estágio e pesquisa e também de estudos no exterior.

Por outro lado, o fato de a grade curricular do curso ser tão completa também torna-se, de certa forma, uma desvantagem. A carga de trabalho envolvida no curso é muito grande, considerando o número de créditos realizados por semestre e a carga de trabalho associada a cada um desses créditos. Durante a elaboração deste trabalho, realizado no último semestre da graduação do autor, foi possível focar em um tema específico como AM e usufruir de maior disponibilidade de tempo para elaborar um trabalho de qualidade e sedimentar os conhecimentos obtidos. Entretanto, ao longo dos anos anteriores do curso de graduação, o número demasiadamente elevado de disciplinas por semestre impedia que se tivesse tempo para aprofundamento em cada um dos tópicos sendo estudados, sacrificando assim qualidade de aprendizado em função da quantidade de disciplinas sendo cursadas. Outro ponto de destaque é a falta de oferecimento de uma ênfase em Inteligência Artificial. Dada a relevância da área na atualidade, tanto no cenário acadêmico quanto no mercado de trabalho, o oferecimento de mais disciplinas relacionadas a Inteligência Artificial e AM e, possivelmente, uma ênfase na área, poderiam contribuir muito positivamente para a formação dos estudantes.

4.4 Trabalhos Futuros

Os resultados obtidos sugerem que utilizar um SDAE para redução de dimensionalidade pode ser uma solução de grande potencial, principalmente considerando a possibilidade de paralelizar o treinamento de uma rede neural profunda SDAE utilizando-se GPUs. Como trabalho futuro, propõe-se realizar mais experimentos envolvendo o SDAE na análise de dados de expressão gênica, possivelmente ajustando parâmetros, o que não foi feito neste trabalho, e aumentando o número de épocas de treinamento.

Além disso, as redes convolucionais utilizadas apresentaram resultados interessantes. Isso torna atrativa uma investigação aprofundada de variações na arquitetura de CNNs, de forma

a melhorar o desempenho na análise dos dados utilizados neste trabalho.

Finalmente, uma outra possibilidade é variar a ordem das etapas de cada experimento, por exemplo realizando redução de dimensionalidade antes do balanceamento dos dados. Isso poderia ser feito a fim de analisar a influência dos métodos de balanceamento no desempenho insatisfatório da técnica FCBF.

REFERÊNCIAS

BATISTA, G. E.; PRATI, R. C.; MONARD, M. C. A study of the behavior of several methods for balancing machine learning training data. **ACM Sigkdd Explorations Newsletter**, ACM, v. 6, n. 1, p. 20–29, 2004. Citado 2 vezes nas páginas 27 e 41.

BENGIO, Y. Machines who learn. **Scientific American**, v. 314, n. 6, p. 46–51, 2016. Citado 2 vezes nas páginas 14 e 21.

BLIER, L. **A BRIEF REPORT OF THE HEURITECH DEEP LEARNING MEETUP #5**. 2016. Online; accessado em 21 de Outubro de 2017. Disponível em: <https://blog.heuritech.com/2016/02/29/a-brief-report-of-the-heuritech-deep-learning-meetup-5/>. Citado na página 24.

BREIMAN, L.; FRIEDMAN, J.; OLSHEN, R.; STONE, C. **Classification and Regression Trees**. Monterey, CA: Wadsworth and Brooks, 1984. Citado na página 17.

CALZA, S.; RAFFELSBERGER, W.; PLONER, A.; SAHEL, J.; LEVEILLARD, T.; PAWITAN, Y. Filtering genes to improve sensitivity in oligonucleotide microarray data analysis. **Nucleic acids research**, Oxford University Press, v. 35, n. 16, p. e102, 2007. Citado na página 41.

CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. Smote: Synthetic minority over-sampling technique. **J. Artif. Int. Res.**, AI Access Foundation, USA, v. 16, n. 1, p. 321–357, jun. 2002. ISSN 1076-9757. Disponível em: <http://dl.acm.org/citation.cfm?id=1622407.1622416>. Citado 2 vezes nas páginas 27 e 41.

CHEN, Y.; LI, Y.; NARAYAN, R.; SUBRAMANIAN, A.; XIE, X. Gene expression inference with deep learning. **Bioinformatics**, v. 32, n. 12, p. 1832–1839, 2016. Disponível em: <+http://dx.doi.org/10.1093/bioinformatics/btw074>. Citado 4 vezes nas páginas 33, 34, 43 e 44.

CHOLLET, F. **Building Autoencoders in Keras**. 2016. Citado 2 vezes nas páginas 42 e 43.

CHOLLET, F. *et al.* **Keras**. [S.l.]: GitHub, 2015. <https://github.com/fchollet/keras>. Citado na página 42.

COLOMBO, J.; RAHAL, P. A tecnologia de microarray no estudo do câncer de cabeça e pescoço. **Brazilian Journal of Biosciences**, Instituto de Biociências da Universidade Federal do Rio Grande do Sul, v. 8, n. 1, p. 64–72, 2009. Citado na página 31.

DANAEE, P.; GHAEINI, R.; HENDRIX, D. A. A deep learning approach for cancer detection and relevant gene identification. In: NIH PUBLIC ACCESS. **Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing**. [S.l.], 2016. v. 22, p. 219. Citado 6 vezes nas páginas 25, 33, 34, 42, 43 e 57.

FLACH, P. **Machine Learning: The Art and Science of Algorithms That Make Sense of Data**. New York, NY, USA: Cambridge University Press, 2012. ISBN 1107422221, 9781107422223. Citado na página 17.

GAMA, J.; CARVALHO, A. C. P. L. F.; FACELI, K.; LORENA, A. **Inteligência Artificial - Uma Abordagem de Aprendizado de Máquina**. [S.l.]: LTC, 2011. Citado 9 vezes nas páginas 8, 16, 17, 18, 19, 20, 21, 29 e 38.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: **Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics**. [S.l.: s.n.], 2010. p. 249–256. Citado na página 43.

GOLDMAN, D.; DOMSCHKE, K. Making sense of deep sequencing. **International Journal of Neuropsychopharmacology**, v. 17, n. 10, p. 1717–1725, 2014. Disponível em: <+http://dx.doi.org/10.1017/S1461145714000789>. Citado 2 vezes nas páginas 29 e 30.

GOLUB, T. R.; SLONIM, D. K.; TAMAYO, P.; HUARD, C.; GAASENBEEK, M.; MESIROV, J. P.; COLLER, H.; LOH, M. L.; DOWNING, J. R.; CALIGIURI, M. A.; BLOOMFIELD, C. D.; LANDER, E. S. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. **Science**, American Association for the Advancement of Science, v. 286, n. 5439, p. 531–537, 1999. ISSN 0036-8075. Disponível em: <http://science.sciencemag.org/content/286/5439/531>. Citado na página 32.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citado 5 vezes nas páginas 22, 23, 24, 25 e 44.

HE, H.; GARCIA, E. A. Learning from imbalanced data. **IEEE Trans. on Knowl. and Data Eng.**, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 21, n. 9, p. 1263–1284, set. 2009. ISSN 1041-4347. Disponível em: <http://dx.doi.org/10.1109/TKDE.2008.239>. Citado 2 vezes nas páginas 27 e 29.

JÄGER, J.; SENGUPTA, R.; RUZZO, W. L. Improved gene selection for classification of microarrays. In: **Pacific Symposium on Biocomputing**. [S.l.: s.n.], 2002. v. 8, p. 53–64. Citado na página 41.

KIRA, K.; RENDELL, L. A. A practical approach to feature selection. In: **Proceedings of the ninth international workshop on Machine learning**. [S.l.: s.n.], 1992. p. 249–256. Citado 2 vezes nas páginas 28 e 41.

KOHAVI, R. *et al.* A study of cross-validation and bootstrap for accuracy estimation and model selection. In: STANFORD, CA. **Ijcai**. [S.l.], 1995. v. 14, n. 2, p. 1137–1145. Citado na página 40.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 5 2015. ISSN 0028-0836. Citado 4 vezes nas páginas 21, 22, 24 e 34.

LECUN, Y.; RANZATO, M. Deep learning tutorial. In: CITESEER. **Tutorials in International Conference on Machine Learning (ICML'13)**. [S.l.], 2013. Citado na página 44.

LEMAîTRE, G.; NOGUEIRA, F.; ARIDAS, C. K. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. **Journal of Machine Learning Research**, v. 18, n. 17, p. 1–5, 2017. Disponível em: <http://jmlr.org/papers/v18/16-365>. Citado na página 41.

LI, J.; CHENG, K.; WANG, S.; MORSTATTER, F.; ROBERT, T.; TANG, J.; LIU, H. Feature selection: A data perspective. **arXiv:1601.07996**, 2016. Citado na página 42.

MANI, I.; ZHANG, I. knn approach to unbalanced data distributions: a case study involving information extraction. In: **Proceedings of workshop on learning from imbalanced datasets**. [S.l.: s.n.], 2003. v. 126. Citado 2 vezes nas páginas 27 e 41.

MITCHELL, T. M. **Machine Learning**. [S.l.]: McGraw-Hill Science/Engineering/Math, 1997. Citado na página 16.

National Cancer Institute. **Sample Type Codes**. 2017. Online; accessado em 27 de Outubro de 2017. Disponível em: <https://gdc.cancer.gov/resources-tcga-users/tcga-code-tables/sample-type-codes>. Citado na página 38.

NETO, E. P. d. S. A. **Classificação de dados de expressão gênica utilizando Aprendizado de Máquina**. Monografia (Tese) — Universidade de São Paulo. Instituto de Ciências Matemáticas e de Computação, São Carlos, SP, 2016. Citado 7 vezes nas páginas 29, 32, 34, 38, 39, 42 e 45.

NIELSEN, M. A. **Neural Networks and Deep Learning**. [S.l.]: Determination Press, 2015. <http://neuralnetworksanddeeplearning.com>. Citado 4 vezes nas páginas 8, 20, 21 e 22.

NVIDIA Corporation. **GPU Applications: Transforming computational research and engineering - Machine Learning**. 2017. Online; accessado em 27 de Outubro de 2017. Disponível em: <http://www.nvidia.com/object/machine-learning.html>. Citado na página 37.

PRIDDY, K. L.; KELLER, P. E. **Artificial neural networks: an introduction**. [S.l.]: SPIE press, 2005. v. 68. Citado na página 39.

Python Software Foundation. **How To: The Comma Separated Value (CSV) File Format**. 2017. Citado na página 38.

REPICI, D. J. **How To: The Comma Separated Value (CSV) File Format**. 2002. Online; accessado em 27 de Outubro de 2017. Disponível em: <http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm>. Citado na página 38.

ROY, N. C.; ALTERMANN, E.; PARK, Z. A.; MCNABB, W. C. A comparison of analog and next-generation transcriptomic tools for mammalian studies. **Briefings in Functional Genomics**, v. 10, n. 3, p. 135–150, 2011. Disponível em: <http://bfg.oxfordjournals.org/content/10/3/135.abstract>. Citado 2 vezes nas páginas 31 e 32.

RUSSELL, S. J.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 2. ed. [S.l.]: Pearson Education, 2003. ISBN 0137903952. Citado na página 19.

SAJDA, P. Machine learning for detection and diagnosis of disease. **Annual review of biomedical engineering**, v. 8, p. 537–65, 2006. Citado 2 vezes nas páginas 14 e 34.

Scikit-Learn Developers. **Decision Trees**. 2017. Online; accessado em 27 de Outubro de 2017. Disponível em: <http://scikit-learn.org/stable/modules/tree.html>. Citado 2 vezes nas páginas 43 e 57.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **CoRR**, abs/1409.1556, 2014. Disponível em: <http://arxiv.org/abs/1409.1556>. Citado na página 24.

SOUZA, B. F. d. **Meta-aprendizagem aplicada à classificação de dados de expressão gênica [tese]**. [S.l.]: Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, 2010. Citado 6 vezes nas páginas 16, 31, 32, 34, 41 e 42.

TAN, A. C.; GILBERT, D. Ensemble machine learning on gene expression data for cancer classification. University of Glasgow, 2003. Citado 3 vezes nas páginas 14, 32 e 34.

TANG, J.; ALELYANI, S.; LIU, H. Feature selection for classification: A review. **Data Classification: Algorithms and Applications**, CRC Press, p. 37, 2014. Citado na página 28.

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. **arXiv e-prints**, abs/1605.02688, maio 2016. Disponível em: <http://arxiv.org/abs/1605.02688>. Citado na página 42.

VIDYASAGAR, M. Machine learning methods in the computational biology of cancer. In: THE ROYAL SOCIETY. **Proc. R. Soc. A**. [S.l.], 2014. v. 470, n. 2167, p. 20140081. Citado na página 34.

VINCENT, P.; LAROCHELLE, H.; LAJOIE, I.; BENGIO, Y.; MANZAGOL, P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. **J. Mach. Learn. Res.**, JMLR.org, v. 11, p. 3371–3408, dez. 2010. ISSN 1532-4435. Disponível em: <http://dl.acm.org/citation.cfm?id=1756006.1953039>. Citado 3 vezes nas páginas 25, 26 e 27.

WANG, Z.; GERSTEIN, M. Rna-seq: a revolutionary tool for transcriptomics. **Nat Rev Genet**, v. 10, n. 1, p. 57–63, 2009. Citado 4 vezes nas páginas 30, 31, 32 e 34.

YU, L.; LIU, H. Feature selection for high-dimensional data: A fast correlation-based filter solution. In: **Proceedings of the 20th international conference on machine learning (ICML-03)**. [S.l.: s.n.], 2003. p. 856–863. Citado 2 vezes nas páginas 28 e 41.

ZEILER, M. D. Adadelta: an adaptive learning rate method. **arXiv preprint arXiv:1212.5701**, 2012. Citado na página 42.