UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS FÍSICAS E MATEMÁTICAS
PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA

Caio Boccato Dias de Góes

Aprendizado de máquina clássico e quântico: classificação de estados emaranhados e equações diferenciais parciais

Florianópolis
2023

Caio Boccato Dias de Góes

Aprendizado de máquina clássico e quântico: classificação de estados emaranhados e equações diferenciais parciais

Tese submetida ao Programa de Pós-Graduação em Física da Universidade Federal de Santa Catarina para a obtenção do título de doutor em Física.
Orientador: Prof. Dr. Eduardo Inacio Duzzioni

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Góes, Caio
Aprendizado de máquina clássico e quântico : classificação de estados emaranhados e equações diferenciais parciais / Caio Góes ; orientador, Eduardo Duzzioni, 2023.
132 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro de Ciências Físicas e Matemáticas, Programa de Pós-Graduação em Física, Florianópolis, 2023.

Inclui referências.

1. Física. 2. computação quântica. 3. aprendizado de máquina. 4. equação diferencial parcial. 5. emaranhamento. I. Duzzioni, Eduardo. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Física. III. Título.

Caio Boccato Dias de Góes

Aprendizado de máquina clássico e quântico: classificação de estados emaranhados e equações diferenciais parciais

O presente trabalho em nível de doutorado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Celso Jorge Villas Boas , Dr.
Universidade Federal de São Carlos

Prof. Fernando da Rocha Vaz Bandeira de Melo, Dr.
Centro Brasileiro de Pesquisas Físicas

Prof. Luis Guilherme de Carvalho Rego, Dr.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de doutor em Física.

Documento assinado digitalmente
Paulo Henrique Souto Ribeiro
Data: 04/04/2023 09:02:57-0300
CPF: ***.886.506-**
Verifique as assinaturas em https://v.ufsc.br

Coordenação do Programa de
Pós-Graduação em Física

Documento assinado digitalmente
Eduardo Inacio Duzzioni
Data: 04/04/2023 10:36:14-0300
CPF: ***.513.789-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Eduardo Inacio Duzzioni
Orientador

Florianópolis, 2023.

Este trabalho é dedicado aos meus colegas de classe,
amigos, aos meus queridos pais e à minha esposa.

AGRADECIMENTOS

Começo agradecendo ao meu orientador, Prof. Dr. Eduardo Inacio Duzzioni, pelo incentivo ao longo do trabalho, pela confiança e apoio constante no desenvolvimento deste trabalho e pela ajuda em consultas de toda índole.

Aos meus amigos de grupo de pesquisa e sala Thiago Maciel, Antônio Crispim Lourenço, Giovani G. Pollachini entre outros pelas discussões e ajuda ao longo destes anos de estudo.

Ao PPGFSC pela oportunidade de estudo e utilização de suas dependências.

Também gostaria de agradecer à Fundação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES, pela bolsa de estudos.

A minha família, pelo incentivo e apoio incondicional dedicados ao longo destes anos da minha jornada acadêmica.

A minha esposa Fransueli Bahr da Silva de Góes por todo incentivo, ajuda e paciência necessários ao compartilhamento de uma vida.

RESUMO

Neste trabalho foram desenvolvidos dois *frameworks* para a solução de problemas no campo do Machine Learning Quântico. O primeiro trabalho atuou no campo da computação clássica com um conjunto de dados quântico. Foi apresentada uma abordagem baseada em Machine Learning Automático para classificar estados aleatórios de dois qutrits como separável ou emaranhado mesmo quando o critério de Perez-Horodecki falha. O *framework* foi aplicado com sucesso utilizando dados suficientes para realizar uma tomografia de estados quânticos completa e sem nenhuma medida direta de emaranhamento. Adicionado a isso, foi estimado a Robustez Generalizada de emaranhamento através de técnicas de regressão e utilizada para validar nosso classificador. O segundo trabalho atuou no campo da computação quântica com conjunto de dados clássicos. Foi proposto um algoritmo híbrido baseado em Machine Learning para encontrar a uma solução aproximada para uma equação diferencial parcial com boa precisão e escala favoravelmente com o número de qubits requeridos. A componente clássica consiste no treinamento de diversos regressores (*weak-learners*), capazes de solucionar aproximadamente a equação diferencial utilizando Machine Learning. A componente quântica consiste em adaptar o algoritmo *QBoost* para solucionar problemas de regressão para criar um ensemble de *learners* clássicos. O *framework* foi aplicado com sucesso para solucionar a equação de Burgers 1D com viscosidade, mostrando que o método de ensemble quântico realmente melhora a solução produzida pelos *weak-learners* clássicos. O algoritmo foi implementado nos computadores quânticos da empresa D-Wave Systems Inc., confirmando a boa performance da solução quântica comparada aos métodos de *Simulated anneling* e *Exact Solver*.

Palavras-chave: AutoML. Emaranhamento. PPT. Ensemble. Simulated annealing. Quantum annealing. EDP.

ABSTRACT

In this work, two frameworks for solving problems in the field of Quantum Machine Learning were developed. The first work acted in the field of classical computing with a quantum dataset. An approach based on Automated Machine Learning was presented to classify random states of two qutrits as separable or entangled even when the Perez-Horodecki criterion fails. The framework was successfully applied using enough data to perform a complete quantum state tomography and without any direct measurement of entanglement. Added to this, the Generalized Robustness of entanglement was estimated through regression techniques and used to validate our classifier. The second work is based on the field of quantum computing with classical datasets. A hybrid algorithm based on Machine Learning was proposed to find an approximate solution to a partial differential equation with good accuracy and scales favorably with the number of qubits required. The classic component consists of training several regressors (weak-learners), capable of approximately solving the differential equation using Machine Learning. The quantum component consists of adapting the QBoost algorithm to solve regression problems to create an ensemble of classical learners. The framework was successfully applied to solve the 1D Burgers equation with viscosity, showing that the quantum ensemble method actually improves the solution produced by classical weak-learners. The algorithm was implemented in the quantum computers of the company D-Wave Systems Inc., confirming the good performance of the quantum solution compared to the Simulated anneling and Exact Solver methods.

Keywords:AutoML. Entanglement. PPT. Ensemble. Simulated annealing. Quantum annealing. PDE.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Nos últimos 30 anos, devido ao crescimento do poder computacional e a disponibilidade de uma vasta quantidade de dados, o *ML* (ML, aprendizado de máquina) experimentou um crescente sucesso na resolução de problemas, com aplicações indo da visão computacional (KRIZHEVSKY *et al.*, 2017) a jogar jogos complexos do console Atari (MNIH *et al.*, 2013) ou jogos tradicionais como Go (SILVER *et al.*, 2016). Entretanto, nos últimos anos surgiram desafios que podem botar um fim nesta revolução. Os dois principais desafios são o aumento significativo do tamanho dos conjuntos de dados disponíveis e o final da lei de Moore (MARKOV, 2014). Enquanto novos desenvolvimentos nas arquiteturas de hardware, como as unidades de processamento gráficas (GPU) ou as unidades de processamento tensorial (TPU), habilitam melhoras na performance em ordens de magnitude comparadas à unidade de processamento central (CPU), elas não conseguem suportar uma melhora ainda maior, já que elas encontraram suas limitações físicas. Elas não oferecem mais uma solução estrutural para o problema proposto e novas soluções são requisitadas.

Por outro lado, uma nova tecnologia vem alcançando a maturidade lentamente. A computação quântica, uma forma de computação que faz uso de fenômenos da mecânica quântica, como a superposição e o emaranhamento, está sendo pensada para superar essas limitações dos hardwares clássicos (NIELSEN; CHUANG, 2000). Algoritmos quânticos, ou seja, algoritmos que podem ser executados em um computador quântico, têm sido investigados desde a década de 80, e vem recebendo, recentemente, um aumento de interesse pelo mundo.

1.1 MACHINE LEARNING

Machine learning (aprendizado de máquina, ML) é O campo de estudo que envolve algoritmos que permitem aos computadores resolverem problemas utilizando dados, retirando a necessidade de costurar soluções específicas (SCHULD; PETRUCCIONE, 2018). Esta prática transformou cada aspecto da sociedade moderna, da medicina (GEORGE, 2001) a finanças (OZBAYOGLU *et al.*, 2020). Um dos braços do ML é o aprendizado supervisionado, que consiste em treinar modelos a aprender a relação entre as entradas e as saídas (HASTIE *et al.*, 2009). Tipicamente, começamos adiquirindo um conjunto de treinamento com dados rotulados, ou seja, uma coleção de pares de entrada e saída. Como exemplo podemos usar a relação entre as características de diferentes frutas como densidade, tamanho, largura, cor para classificar qual é a fruta. Ao treinar o modelo de ML com o conjunto de treinamento, o objetivo é que o modelo aprenda as relações entre as características físicas e as frutas. Se esse for o caso, o modelo pode ser usado para prever qual é a fruta através de suas características físicas, mesmo com exemplos que não estão no conjunto de dados.

Neste caso dizemos que o modelo é capaz de generalizar a partir de dados não vistos por ele.

De forma simplificada, o treinamento de um modelo de ML começa com a definição de uma função custo (conhecida também como risco ou perda), que é uma função escalar positiva que mede qual a acurácia da previsão do modelo a partir de um dado de entrada (HASTIE *et al.*, 2009). Quanto menor o valor da função custo, melhor o modelo reproduz a saída. Desta maneira, treinar um modelo de ML significa minimizar a função custo com respeito ao conjunto de treinamento.

1.2 COMPUTAÇÃO QUÂNTICA

Computação quântica consiste no processamento de informação utilizando sistemas que obedecem as leis da mecânica quântica (NIELSEN; CHUANG, 2000). Em 1982, Richard Feynman apontou que sistemas quânticos são difíceis de simular em um computador clássico. Ele sugeriu, então, que essa complexidade poderia ser explorada para construir um computador baseado em princípios da mecânica quântica (NIELSEN; CHUANG, 2000). Apenas três anos depois, David Deutsch formalizou a teoria descrevendo um dispositivo, um computador quântico universal (DEUTSCH, 1985). Mesmo que esse dispositivo ainda não tenha sido realizado fisicamente, as pessoas começaram a desenvolver algoritmos para computadores quânticos que teoricamente podem ser mais eficazes que os algoritmos clássicos. Em 1996, Seth Lloyd mostrou que sistemas da mecânica quântica poderiam ser simulados eficientemente em computadores quânticos (LLOYD, 1996). Peter Shor desenvolveu o algoritmo que leva o seu nome e que fatora números inteiros em números primos em tempo polinomial, potencialmente quebrando os protocolos de criptografia de hoje em dia (SHOR, P., 1994). A fatoração em primos é conhecido como sendo um problema sub-exponencialmente difícil para os computadores clássicos, e a eficiência do algoritmo de Shor mostrou que a utilização do computador quântico pode ir muito além da simulação de sistemas quânticos (SHOR, P. W., 1997). O algoritmo de Shor despertou um grande interesse da comunidade mundial e colocou um holofote na computação quântica.

Hoje, existe um grande foco em construir um computador quântico, com grandes empresas como Google e IBM na liderança, assim como um enorme financiamento governamental de países como EUA e China. Apesar do esforço, os computadores quânticos de hoje não são capazes de implementar algoritmos quânticos que são capazes de mudar o mundo. Isso porque os computadores quânticos ainda são pequenos e muito ruidosos. Enquanto o algoritmo é executado ele manipula um sistema quântico muito delicado, e este sistema é altamente suscetível à interferências do ambiente, causando a degradação do estado. Este fenômeno é chamado decoerência, e coloca um limite estreito no tamanho do algoritmo (SAKI *et al.*, 2019). O limite no hardware e a presença de ruído faz com que todos os algoritmos quânticos teorizados, como o

algoritmo de Shor, sejam impossíveis de implementar nos dias de hoje (BENEDETTI *et al.*, 2019).

1.3 MACHINE LEARNING QUÂNTICO

Pode o computador quântico ser últil para desenvolver algoritmos de ML, e esses algoritmos podem ser aplicados nos hardwares ruidosos em um futuro próximo? Combinando os dois campos, obtemos um novo campo de pesquisa emergente, o machine learning quântico. Na Figura 1 podemos ver as diferentes maneiras de combinar o campo da computação quântica com o campo do ML.

Uma das áreas que recebeu uma atenção particular é a do ML quântico (CILIBERTO *et al.*, 2020), uma combinação da mecânica quântica e do ML. Essa nova área pode ser dividida, geralmente, em três áreas de pesquisa distintas:

- QC - Algoritmos de ML são executados em computadores quânticos (QPU) e aplicados a dados clássicos
- CQ - Algoritmos de ML são executados em computadores clássicos e aplicados a dados quânticos.
- QQ - Algoritmos de ML são executados em computadores quânticos e aplicados a dados quânticos.

O principal objetivo destas abordagens é criar algoritmos quânticos capazes de alcançar a vantagem quântica sobre qualquer algoritmo clássico conhecido para a mesma tarefa.

1.4 ORGANIZAÇÃO DA TESE

No Capítulo 2 da tese, nós introduzimos fundamentos teóricos do campo de ML. No Capítulo 3 descrevemos um trabalho realizado dentro do campo CQ, onde utilizamos algoritmos clássicos de ML em um dataset quântico, contendo estados de dois qutrits, para classificar se o estado é emaranhado, emaranhado preso ou separável. No Capítulo 4 apresentamos outro trabalho, dentro do campo QC, onde utilizamos um algoritmo híbrido de ML para realizar a regressão da equação de Burgers em uma dimensão com viscosidade. No capítulo 5 apresentamos a conclusão referente aos resultados apresentados nesta tese.

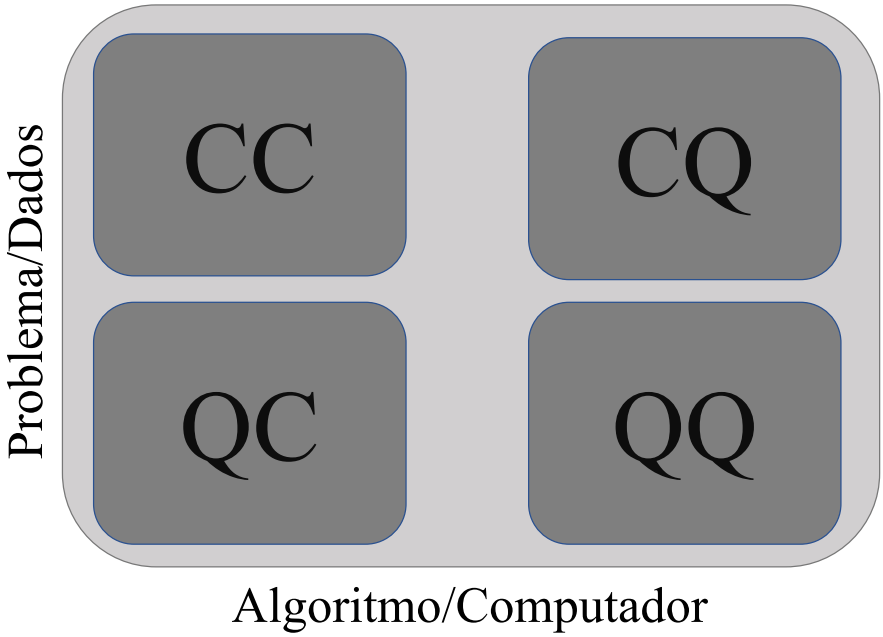


Figura 1 – Diferentes campos de estudo em ML quântico. As diferentes áreas estão relacionadas à escolha do tipo de algoritmo, isto é, se é executado em um computador clássico ou quântico, e a escolha do problema a ser solucionado, ou seja, se irá operar em dados clássicos ou quânticos. As palavras clássico e quântico estão representadas por C e Q, respectivamente.

2 APRENDIZADO DE MÁQUINA (*MACHINE LEARNING*, ML)

ML é um campo de estudos onde o paradigma de como uma computação é realizada foi alterado. Na computação tradicional, de forma simplificada, uma computação é realizada fornecendo algum dado de entrada e uma regra, e o produto da computação será um dado de saída que transformou o dado de entrada de acordo com a regra fornecida. Já no caso do ML, é fornecido um conjunto de dados de entrada e um conjunto de dados de saída e o resultado da computação será a regra que transforma o conjunto de entrada no conjunto de saída.

O campo do ML possui três grandes grupos de problemas: o aprendizado supervisionado, o não supervisionado e o semi-supervisionado. O aprendizado supervisionado consiste em fornecer ao computador uma sequência de dados junto com determinados valores ou rótulos. Neste procedimento, durante a fase de treinamento, o algoritmo irá tentar relacionar padrões nos dados com os valores ou rótulos fornecidos. No caso do aprendizado não supervisionado são fornecidos dados sem rótulos ao computador. O algoritmo, neste caso, irá tentar agrupar os dados fornecidos em classes distintas, criando assim rótulos para cada grupo de dados. No caso semi-supervisionado, o computador tem acesso a apenas alguns dados supervisionados e a partir destes tenta encontrar estruturas dentro dos dados que não possuem rótulos.

Os principais problemas ao qual o ML é capaz de resolver são:

- **Reconhecimento de padrões:** Atribuir um padrão de entrada representado por um vetor de variáveis à uma das classes pré-especificadas. Algumas aplicações bem conhecidas deste problema incluem reconhecimento de discursos (DIGALAKIS *et al.*, 1993; SELTZER *et al.*, 2013), classificação do formato de ondas de eletroencefalograma (WULSIN, D. *et al.*, 2010; WULSIN, D. F. *et al.*, 2011), classificação de células sanguíneas (PIURI; SCOTTI, s.d.; ONGUN *et al.*, s.d.) e inspeção de circuitos impressos (JARVIS, 1980; TIEN *et al.*, 2004).

- **Clusterização/Categorização:** Um problema clássico de aprendizado não supervisionado. O problema consiste em explorar a similaridade entre os vetores de variáveis e agrupar em classes os vetores que possuam grande similaridade. Podemos citar como aplicações mineração de dados (BOSE; MAHAPATRA, 2001; LOW *et al.*, 2012; BALL; BRUNNER, 2010) e compressão de dados (FANG *et al.*, 1992).

- **Aproximação de funções:** Dado um conjunto de dados contendo n exemplos $\{x_1^{(1)}, x_2^{(1)}, ..., x_1^{(n)}, x_2^{(n)}\}$ com os rótulos associados $\{y^{(1)}, ..., y^{(n)}\}$ gerados por uma função, não conhecida, $h(x_1, x_2)$. O objetivo é encontrar uma função aproximada para $h(x_1, x_2)$. Este problema está diretamente relacionado a sistemas não lineares (FUNAHASHI, 1989; CHEN *et al.*, 1990; CHEN; BILLINGS, 1992).

- **Previsão:** Dado um conjunto $\{y(t_1), ..., y(t_n)\}$, em sequência temporal, o problema é prever o valor da amostra $y(t_{n+1})$ em um dado tempo no futuro t_{n+1}. Este tipo de problema possui um impacto muito grande na tomada de decisões em ciências, engenharias e negócios. Podemos observar aplicações diretas do problema na previsão do tempo (SHARMA *et al.*, 2011; RASOULI *et al.*, 2012) e em previsão da bolsa de valores (PATEL *et al.*, 2015a, 2015b).

- **Otimização:** Tem como objetivo encontrar uma solução que satisfaça um conjunto de restrições de maneira que uma função seja maximizada ou minimizada em relação a uma ou mais variáveis do problema. Existe uma variedade de problemas em diversas áreas, como estatística, matemática, ciências, engenharia, medicina, economia, etc. Podemos citar como exemplo o problema do caixeiro viajante (MASUTTI; CASTRO, 2009; GELENBE *et al.*, s.d.; MODARES *et al.*, 1999), muito presente em qualquer problema de logística.

- **Controle:** O problema foca no controle de sistemas dinâmicos. Considere um sistema definido pela sequência de elementos $\{u(t), y(t)\}$, onde $u(t)$ é uma entrada de controle e $y(t)$ é a saída resultante do sistema no tempo t. O objetivo é através de uma entrada de controle $u(t)$ o modelo de ML seja capaz de produzir no sistema uma trajetória determinada pela saída $y(t)$. Podemos citar como exemplo de controle de potência máxima em sistemas PV (VEERACHARY *et al.*, 2003; HIYAMA *et al.*, 1995) e na utilização de redes neurais para aumentar a fidelidade de estados na medição de múltiplos qubits em armadilhas de íons (SEIF *et al.*, 2018).

2.1 MODELOS DE ML

Nesta seção iremos abordar diversos modelos de ML que de alguma maneira foram utilizados nos trabalhos que compõem esta tese. Para facilitar a compreensão de tais modelos irei utilizar um problema simples como exemplo que possa ser resolvido pelos algoritmos propostos. O problema tem como objetivo classificar os dados entre quatro frutas distintas maçãs, laranjas, limões e mexericas (rótulos, "*labels*"), a partir das características comprimento e altura ("*features*"). Definimos então o conjunto de exemplos de diversas características associadas a um rótulo como um conjunto de dados $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$, onde $\mathcal{X} = \{(x_1^1, x_2^1), ..., (x_1^n, x_2^n)\}$ são nossas *n features* e $\mathcal{Y} = \{y^1, ..., y^n\}$ são os *n* rótulos associados às *features* contidas em $\mathcal{X}$. O conjunto de dados utilizados nos exemplos foram utilizados para realizar o treinamento dos modelos de ML, ou seja, o conjunto de dados é o conjunto de treinamento. Para o conjunto de teste utilizamos uma grade no espaço das *features*, ou seja, o conjunto de treinamento possui pontos distribuídos em forma de grade por todo o espaço.

2.1.1 k Primeiros Vizinhos (*"k-nearest neighbors"*, KNN)

Este algoritmo (ALTMAN, 1992), que pode ser utilizado tanto para classificação ou regressão, consiste basicamente em armazenar todo o conjunto de dados na memoria e realizar o cálculo da distância entre os pontos através de uma dada métrica (como por exemplo, distância euclidiana). Para realizar uma previsão, em relação a um novo dado, o algoritmo encontra os dados mais próximos deste no conjunto de treinamento. A letra k corresponde ao número de pontos vizinhos ao qual o novo dado será comparado. Para o caso de $k = 1$ apenas o primeiro vizinho é considerado. Para $k > 3$ o algoritmo irá realizar uma votação. Ganha o rótulo que possuir o maior número de vizinhos.

Mais especificamente, o algoritmo segue os seguintes passos:

1. Carrega os dados.
2. Insira o número de vizinhos k.
3. Para cada exemplo nos dados.
    a) Calcule a distância entre o exemplo consultado e os exemplos dos dados carregados.
    b) Adicione a distância e o rótulo do exemplo em uma matriz.
4. Ordene a matriz de distâncias e rótulos do menor para para o maior em relação a distâncias.
5. Pegue as primeiras k entradas da matriz ordenada.
6. Pegue os rótulos das k entradas selecionadas.
7. Se for um problema de regressão, retorne a média dos k rótulos.
8. Se for um problema de classificação, retorne o rótulo majoritário entre as k entradas.

Considerando nosso problema das frutas para $k = 7$, digamos que ao realizar a consulta sobre um novo dado o algoritmo encontrou que três dos primeiros vizinhos possuem o rótulo laranja e outros quatro primeiros vizinhos possuem o rótulo maçã. Desta forma por voto majoritário, o algoritmo irá classificar o dado da consulta como sendo uma maçã. Podemos observar o comportamento do algoritmo KNN com $k = 7$ para as variáveis altura e comprimento, como representado na Figura 2. Podemos perceber que conforme aumentamos o valor de k as áreas de cada classe se torna mais bem definidas.

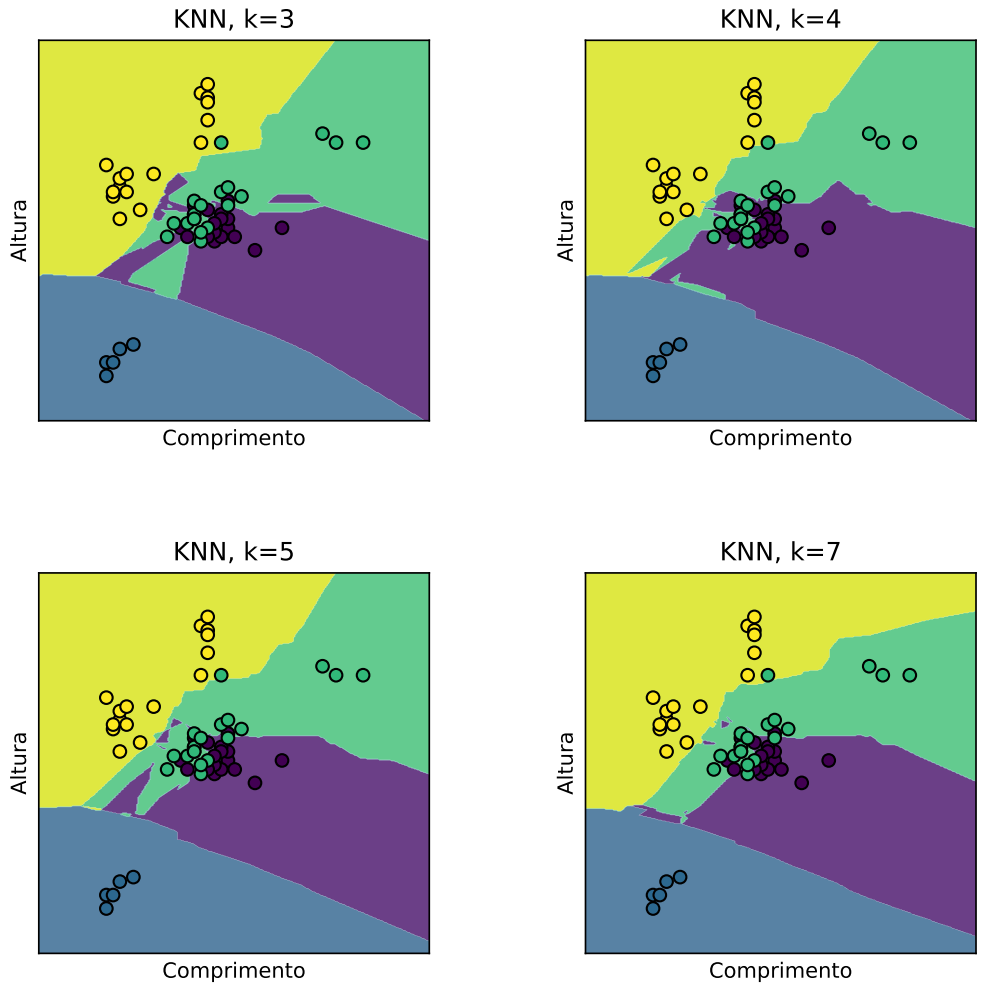


Figura 2 – Algoritmo KNN para diferentes números de vizinhos. Os pontos são os exemplos contidos no conjunto de treinamento. Para o conjunto de teste foi utilizada uma grade de dados do espaço de *features* e são representados pelas áreas hachuradas. Cada cor corresponde a uma fruta diferente (amarelo = mexerica, verde = maçã, roxo = laranja, azul = limão).

2.1.2 Árvore de Decisão

Em sua essência, o algoritmo de árvore de decisão aprende uma hierarquia de questões de SIM ou NÃO levando a uma decisão. A árvore de decisão utiliza conjuntos de dados que consistem em atributos distribuídos em um vetor. Cada atributo do vetor é referente a uma classe. O algoritmo consiste simplesmente em separar os dados, o melhor possível, nas diferentes classes existentes até que um critério de parada seja alcançado. O algoritmo é facilmente visualizado em uma estrutura de árvore.

Os primeiros algoritmos de árvore de decisão, ID3 (QUINLAN, 1986) e C4.5 (SALZBERG, 1994), foram desenvolvidos por Ross Quinlan. Nestes algoritmos existem nós que podem ser divididos em três tipos distintos, raiz, interno e final (também chamado de folha). O nó raiz representa o começo do processo de decisão e não possui bordas de entrada. Os nós internos possuem uma borda de entrada e pelo menos duas bordas de saída. Tanto o nó raiz como os nós internos possuem um teste baseado em um atributo do conjunto de dados que irá diferenciar entre as possíveis classes. Os nós folha consistem na resposta do problema de decisão e possuem apenas uma borda de entrada e não possuem bordas de saída.

Para treinar uma árvore de decisão para criar um classificador é necessário um conjunto de treinamento contendo um atributo rotulado, atributos de entrada, um critério de separação e um critério de parada. Para um dado nó, o critério de separação calcula um valor para todos os atributos. Este valor representa uma medida da quantidade de informação que é ganha separando o nó utilizando esse atributo. O nó será dividido tomando como base o atributo que possuir um maior ganho em relação a quantidade de informação de acordo com o valor do atributo. Este procedimento é realizado para todas as sub-árvores geradas recursivamente até que um critério de parada seja alcançado.

Dado os vetores de treinamento $\mathcal{X} \in \mathbb{R}^n$ e um vetor contendo os rótulos $\mathcal{Y} \in \mathbb{R}^l$, o algoritmo irá particionar o espaço representado pelo vetor $\mathbf{x} \in \mathcal{X}$ de maneira que as amostras com os rótulos iguais ou similares são agrupados.

Sejam os dados no nó m representados por Q_m com N_m amostras. Para cada candidato temos uma função que realiza a divisão $\theta(j, t_m)$, onde j é uma *feature* do vetor x e t_m é um limiar que irá dividir os dados nos subconjuntos $Q_m^{esquerda}(\theta)$ e $Q_m^{direita}(\theta)$, definidos por

$$\begin{aligned} Q_m^{esquerda}(\theta) &= \{(x, y) | x_j \leq t_m\}, \\ Q_m^{direita}(\theta) &= \{(x, y) | x_j > t_m\}. \end{aligned} \tag{1}$$

A qualidade do separador Q_m é computada utilizando uma função custo $L(.)$, essa escolha é feita dependendo da tarefa (classificação ou regressão) da máquina.

$$G(Q_m, \theta) = \frac{N_m^{esquerda}}{N_m} L(Q_m^{esquerda}(\theta)) + \frac{N_m^{direita}}{N_m} L(Q_m^{direita}(\theta)). \tag{2}$$

Encontramos o melhor separador ao encontrar θ^* que minimiza $G(Q_m, \theta)$.

$$\theta^* = argmin_\theta G(Q_m, \theta). \tag{3}$$

Para o problema de classificação teremos que os valores de saída serão $0, 1, ..., K-1$. Para o nó m, definimos

$$p_{mk} = \frac{1}{N_m} \sum_{y \in Q_m} I(y = k), \tag{4}$$

como a proporção da classe k no nó m. Se m for o nó terminal, a probabilidade predita para essa região será p_{mk}, onde $I(y = k)$ fornece o número de casos em que uma determinada classe k ocorre. As medidas mais comuns para a função custo são as seguintes:

- Gini:

$$L(Q_m) = \sum_k p_{mk}(1 - p_{mk}). \tag{5}$$

- Entropia:

$$L(Q_m) = -\sum_k p_{mk} \log(p_{mk}). \tag{6}$$

- Misclassificação:

$$L(Q_m) = 1 - \max(p_{mk}). \tag{7}$$

A diferença entre a utilização das medidas como função custo, vai depender do problema abordado. Sendo que a escolha deve ser feita por experimentação, entendendo qual apresenta o melhor resultado para o problema trabalhado.

Aplicando em nosso problema exemplo de classificação de frutas com a medida gini como função custo, podemos observar a dinâmica do algoritmo através da Figura 3 e a classificação para diferentes profundidades de árvore na Figura 4.

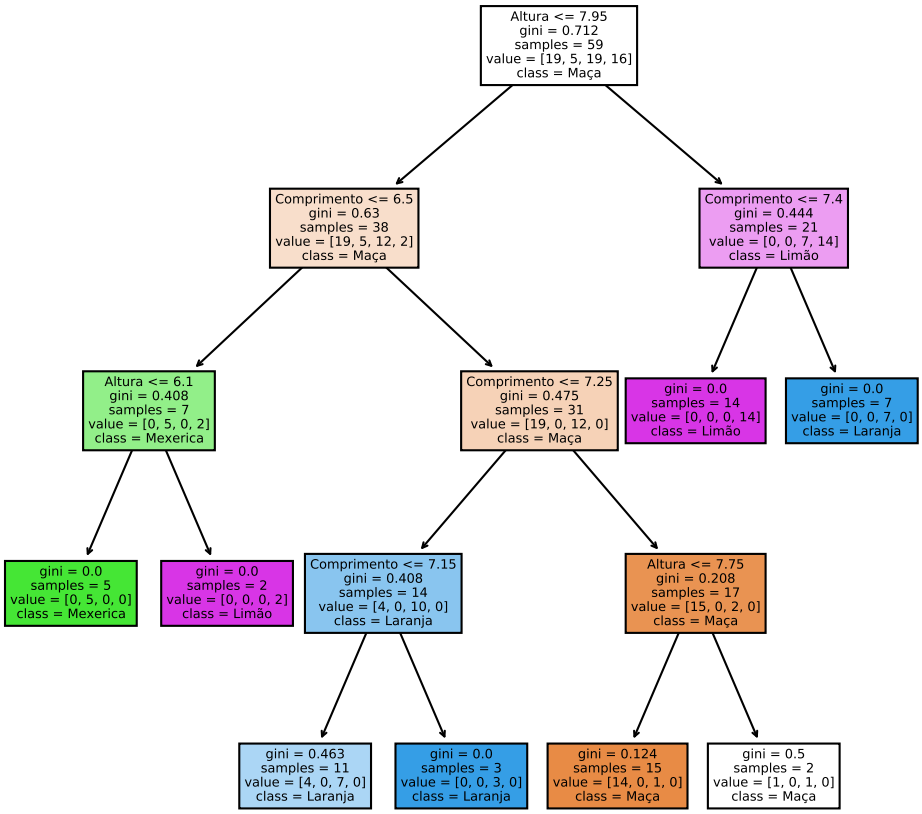


Figura 3 – Estrutura do algoritmo árvore de decisão com profundidade quatro para o problema de classificação de frutas (maçã, mexerica, laranja e limão) com as *features* altura e comprimento. Dentro de cada retângulo podemos ter até cinco atributos distintos, que correspondem ao critério de separação, valor da função custo escolhida, número total de amostras naquele nó, número de amostras em cada classe e, por último, a classe que o nó fornece como saída. O atributo values é o atributo que nos fornece o número de amostras em cada classe, seguindo o seguinte padrão value = [Maçã, Mexerica, Laranja, Limão].

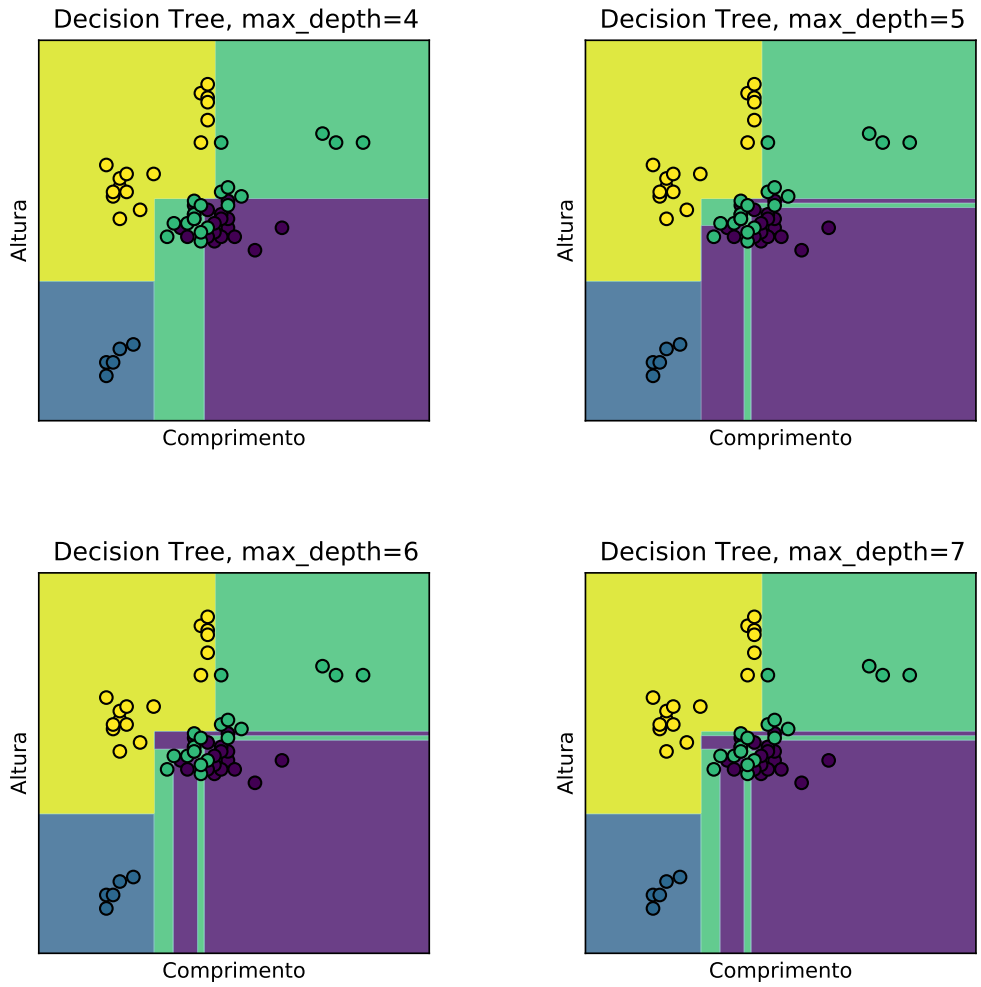


Figura 4 – Classificação para diferentes profundidades de árvore para o algoritmo *Decision Tree*. Os pontos são os exemplos contidos no conjunto de treinamento. Para o conjunto de teste foi utilizado uma grade de dados do espaço de *features* e são representados pelas áreas hachuradas. Cada cor corresponde a uma fruta diferente (amarelo = mexerica, verde = maçã, roxo = laranja, azul = limão).

2.1.3 Suport Vector Machine (SVM)

O SVM é um algoritmo de ML supervisionado (CORTES; VAPNIK, 1995) que divide os dados entre classes pré-definidas. O algoritmo tenta dividir os dados de maneira que produza o menor erro possível e a maior margem possível. Isto se dá devido ao fato de que quanto maiores as áreas vazias próximas a função divisora resulta em erros de classificação menores, pois os rótulos são melhores distinguidos entre eles. A divisão é feita através de um ou múltiplos hiperplanos em um espaço *n*-dimensional. Uma boa separação é alcançada pelo hiperplano que possui a maior distância até a amostra do conjunto de treinamento mais próxima. O modelo de SVM também pode ser estendido para problemas de regressão.

Dado os vetores de treinamento $\mathcal{X} \in \mathbb{R}^n$ e um vetor contendo os rótulos $\mathcal{Y} \in \{1,-1\}^n$, nosso objetivo é encontrar o hiperplano $w \in \mathbb{R}^n$ e $b \in \mathbb{R}$ de maneira que a previsão, $sign(w_T\phi(x) + b)$, onde $\phi(x)$ é uma função do vetor de entrada x, é correta para a maior parte das amostras. SVM resolve o seguinte problema primário:

$$\begin{aligned} \min_{w,b,\eta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \eta_i, \\ sujeito\ a: y_i(w^T \phi(x_i) + b) \geq 1 - \eta_i, \\ \eta_i \geq 0, i = 1, ..., n. \end{aligned} \tag{8}$$

Intuitivamente, estamos tentando maximizar a margem, que pode ser definida como um hiperplano paralelo ao hiperplano w que está localizado no primeiro ponto de determinada classe (minimizando $w^T w$), enquanto adicionamos uma penalidade quando uma amostra é classificada erroneamente ou se encontra no limite da margem. Idealmente, o valor $y_i(w^T \phi(x_i) + b)$ será ≥ 1 para todas as amostras, o que nos indica uma perfeita previsão. Porém, geralmente os problemas não permitem uma completa separação através de um hiperplano, permitimos então que algumas amostras tenham uma distância η_i da borda da margem. A constante C controla a força da penalidade, e como resultado, atua como o inverso de um parametro de regularização. Uma representação do funcionamento do algoritmo SVM pode ser vista na Figura 5.

O problema dual do primário será

$$\begin{aligned} \min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha, \\ sujeito\ a: y^T \alpha = 0, \\ 0 \leq \alpha_i \leq C, i = 1, ..., n, \end{aligned} \tag{9}$$

onde e é um vetor com todos elementos iguais a um, e Q é uma matrix *nxn* positiva semidefinida, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, onde $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ é o núcleo ou *kernel*.

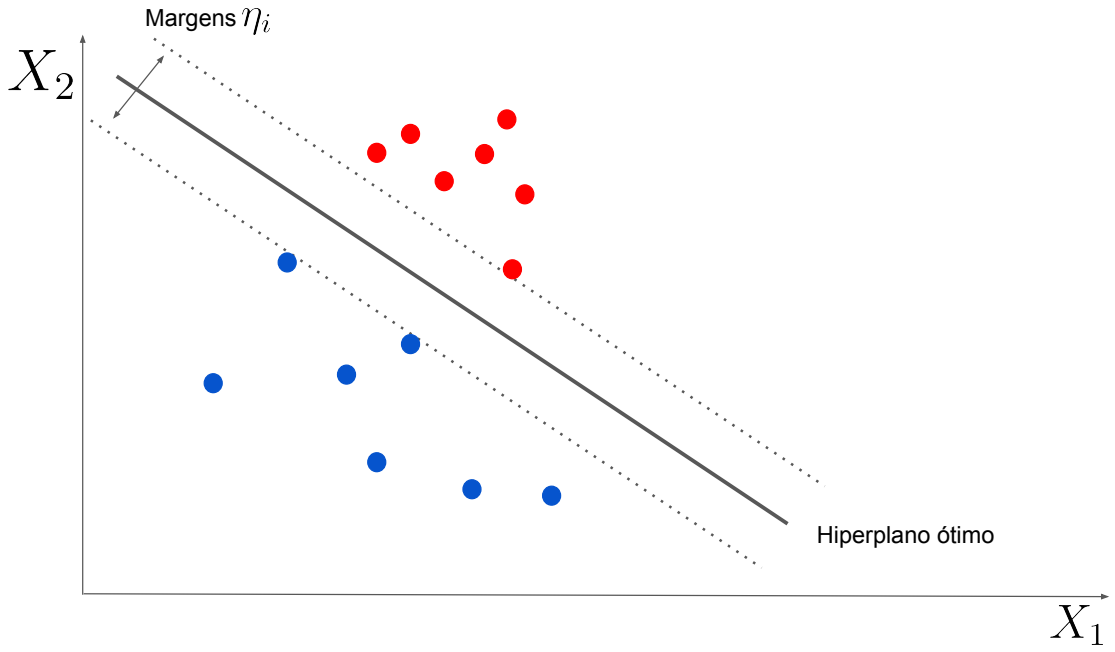


Figura 5 – Representação gráfica do algoritmo de SVM.

O termo α_i é chamado de coeficiente dual e é limitado superiormente por C. Esta formulação evidencia o fato de que os vetores de treinamento são implicitamente mapeados em um espaço dimensional maior pela função ϕ.

Após o problema de optimização ser resolvido, a saída da função decisão para uma determinada amostra será

$$\sum_{i \in SV} y_i \alpha_i K(x_i, x) + b, \tag{10}$$

e a previsão da classe corresponde ao sinal (*sign*) dela. Precisamos somente somar sobre os vetores de suporte (SV), pois os coeficientes duais, α_i, serão zero fora deles.

Podemos ter diversas funções núcleo, como por exemplo:

- linear: $\langle x, x' \rangle$.
- polinomial: $(\gamma \langle x, x' \rangle = r)^d$.
- função de base radial (rbf): $\exp\left(-\gamma \|x - x'\|^2\right)$, com $\gamma > 0$.
- sigmoidal: $tanh(\gamma)\langle x, x' \rangle + r$.

Podemos ver o comportamento de algumas funções núcleo para o problema exemplo do conjunto de treinamento na Figura 6, onde as amostras do conjunto de treinamento são representados pelos pontos nos gráficos e as linhas de classificação, dadas pela função núcleo estão representadas pelas áreas pintadas.

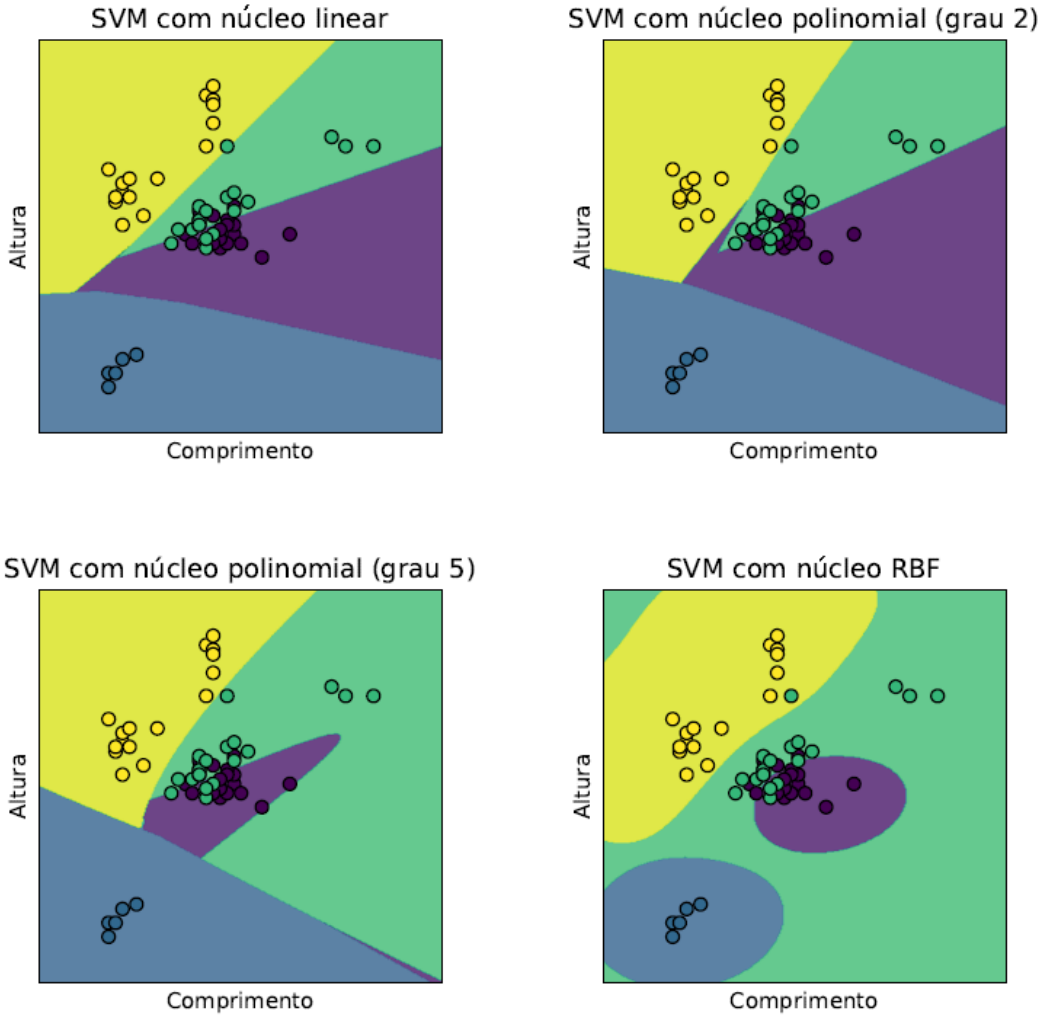


Figura 6 – Algoritmo SVM com diferentes funções núcleo para o problema de classificação de frutas. Os pontos são os exemplos contidos no conjunto de treinamento. Para o conjunto de teste foi utilizado uma grade de dados do espaço de *features* e são representados pelas áreas hachuradas. Cada cor corresponde a uma fruta diferente (amarelo = mexerica, verde = maçã, roxo = laranja, azul = limão).

2.1.4 Redes Neurais Artificiais

Devido a grande popularização das Redes Neurais Artificiais (*artificial neural networks* ANN) profundas e seu impacto no processamento de imagens e vídeos, assim como no processamento de linguagem natural, este modelo de ML hoje é o mais difundido. Aqui iremos apresentar a estrutura básica deste modelo, já que uma explanação completa sobre o tema foge do objetivo desta tese.

As ANNs são um modelo de ML inspirado nos neurônios biológicos. Uma rede neural biológica é um sistema complexo capaz de processar enormes quantidades de informação simultaneamente. Em muitas tarefas o cérebro humano é muito mais eficiente do que qualquer computador.

A estrutura cerebral tem capacidade de adquirir, armazenar e utilizar conhecimentos experimentados pelo ser vivo. Baseado em exemplos e '*feedbacks*' nosso cérebro nos ensina como agrupar diferentes padrões e classificar os mesmos, como por exemplo diferenciar uma laranja de uma maçã, ou reconhecer a face de uma pessoa. Este poder de processamento e reconhecimento estimulou muitos grupos de pesquisa a tentar emular o funcionamento do cérebro em máquinas, pesquisando como usar as máquinas para tarefas que são comum aos humanos. Deste conceito surgiu as ANN.

Em 1943 surgiu o primeiro modelo teórico para um neurônio artificial, criado pelo neuro-psicólogo Warren McColloch e pelo matemático Walter Pitts (MCCULLOCH; PITTS, 1943). Em 1949, um psicólogo Donald Hebb criou a primeira regra de aprendizado para as ANN (HEBB. . . , 1950), chamada de *regra de Hebb*. Esta regra nos diz que se dois neurônios são ativados simultaneamente, então a ligação entre eles tem que ser aumentada. Nos anos seguintes o campo observou um grande progresso. Como marcos deste progresso podemos citar a criação de novas arquiteturas de neurônios, como o *perceptron* (ROSENBLATT, 1958) e o *adaline* (WIDROW; HOFF, 1960) na década de 60. Nos tempos atuais, junto a evolução tecnológica foram criados diversos modelos de rede neural artificial, como os mapas auto-organizáveis (Redes de Kohonen) (KOHONEN, 1982, 1989), redes associativas (TETKO, 2002), redes recorrentes (RUMELHART, David E. *et al.*, 1986), etc. As redes neurais são utilizadas nos mais variados campos da acadêmia, indústria e comércio, desde aplicações na física (DUNJKO; BRIEGEL, 2018) à medicina (BAKATOR; RADOSAV, 2018).

2.1.4.1 Neurônio Biológico

O neurônio biológico é uma célula biológica especial que processa informação. Ela é formada por um corpo celular, *soma*, e dois tipos de ramos em formato de árvore, *axônio* e *dendritos*. A estrutura celular do neurônio pode ser vista na Figura 7. O corpo celular possui um núcleo, que contém informações sobre os traços hereditários, e um

plasma, que possui o equipamento molecular para produzir o material utilizado pelo neurônio.

O funcionamento do neurônio pode ser resumido na recepção de um sinal na forma de pulso elétrico, processamento deste sinal e emissão do sinal processado. O neurônio recebe um sinal de outros neurônios através de seus *dendritos* e transmite o sinal, processado pelo *soma*, através do *axônio*, que eventualmente se ramifica.

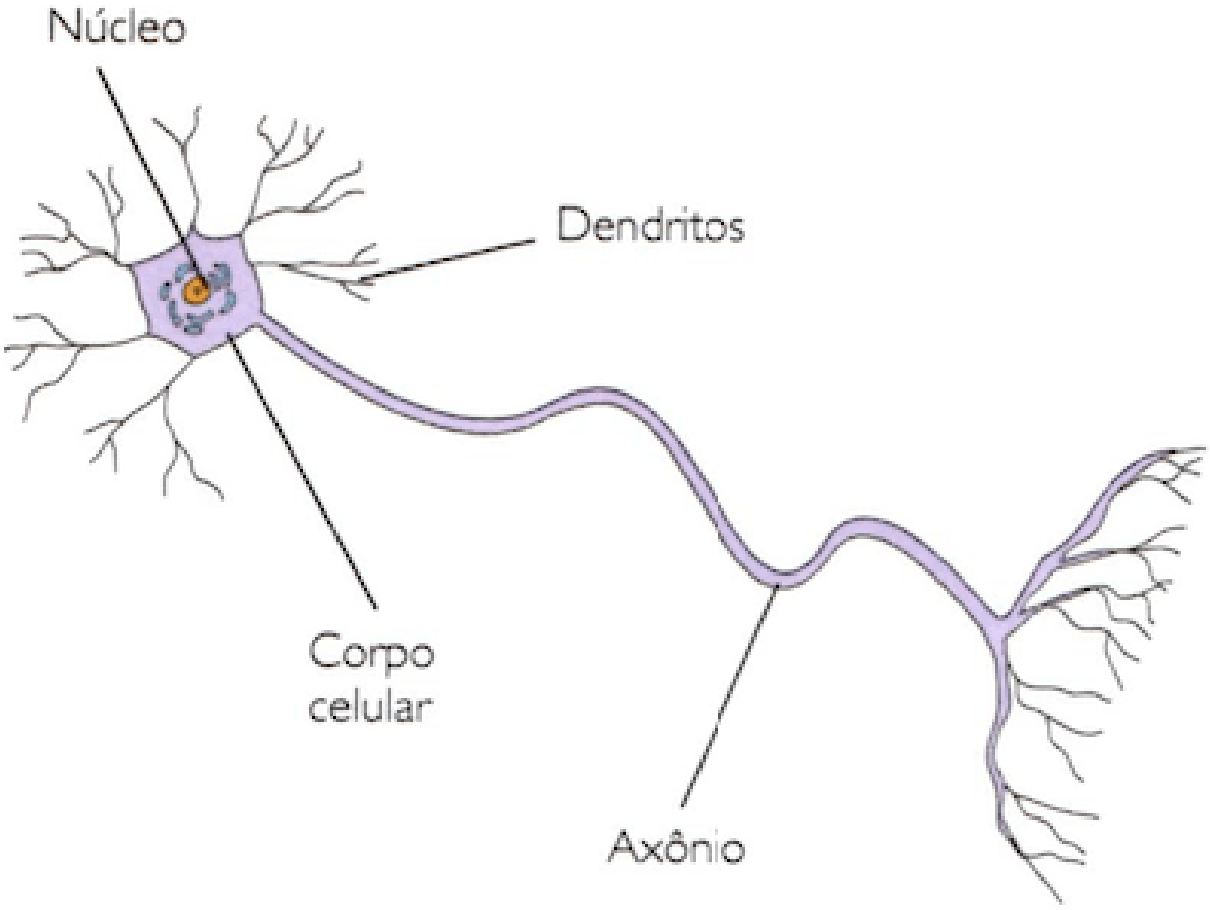


Figura 7 – Estrutura celular do neurônio. Figura retirada de (NEURÔNIOS. . . , s.d.)

Um neurônio se conecta com outro através de uma estrutura chamada de *sinapse*. A sinapse está localizada entre uma ramificação do axônio de um neurônio e o dendrito de outro. Quando o pulso eletrônico alcança a sinapse ocorre um disparo químico chamado de neurotransmissores. Os neurotransmissores se propagam através da sinapse para inibir ou potencializar a tendência de emitir um pulso eletrônico do neurônio receptor. A efetividade da sinapse pode ser ajustada pelo sinal que passa por ela, permitindo assim que a sinapse aprenda através da atividade que ela participa.

O cérebro humano é uma grande rede plana composta por neurônios de aproximadamente $2mm$ ou $3mm$ com uma superfície de aproximadamente $2200cm^2$. O cortex cerebral contém em média 10^{11} neurônios. Cada neurônio está conectado com 10^3 a 10^4 outros neurônios. Em média o cérebro humano possui aproximadamente

10^{14} a 10^{15} interconexões.

As principais características do cérebro humano que a ANN tentará simular são:

- Aprendizado: A rede aprende por experiência, não sendo necessário explicitar através de algoritmos a execução de uma certa tarefa.
- Associação: A rede é capaz de associar padrões distintos.
- Generalização: A rede é capaz de generalizar o seu conhecimento a partir de exemplos anteriores. A rede é capaz de lidar com ruídos gerados por novos padrões.
- Robustez: A rede não sofre com mal funcionamento na perda de elementos processadores e/ou conexões sinápticas.

2.1.4.2 Modelo Matemático do Neurônio

O modelo matemático do neurônio tenta simplificar o neurônio biológico, devido a complexidade do mesmo. Podemos modelar o neurônio a partir de algumas suposições:

- Todos os neurônios são sincronizados, ou seja, o sinal que passa de um neurônio para o outro leva o mesmo tempo para todas as conexões.
- Todo neurônio possui uma função de ativação, que depende dos valores de entrada do neurônio, que determina o valor de saída do neurônio. A função de ativação é independente do tempo.
- Quando o sinal passa pela sinapse, ele muda linearmente, ou seja, o sinal é multiplicado por um número chamado de peso sináptico.

O peso sináptico pode variar ao longo do tempo, fazendo com que a rede possa reagir diferentemente ao mesmo valor de entrada em diferentes momentos. Este processo é o que permite que a ANN se adapte e aprenda, apesar da simplificação em relação a rede neural biológica.

O principio básico do neurônio artificial (ROSENBLATT, 1958) foi criado na metade do século passado e não sofreu muitas alterações desde então. O neurônio é visto como um mecanismo que transforma um sinal de entrada em um sinal de saída. O modelo do neurônio artificial, também chamado de perceptron, consiste na entrada de N sinais, $\{X_1, ..., X_N\}$, na sinapse do neurônio. Então cada sinapse realiza uma modificação linear do sinal utilizando um peso sináptico $\{X_1 w_1, ..., X_N w_N\}$. Após isso o

corpo do neurônio soma os sinais recebidos obtendo um valor e adiciona um valor de bias, $b_0 w_0$,

$$S = \sum_{i=1}^{N} X_i w_i + b_0 w_o. \tag{11}$$

Por fim, uma função de ativação é aplicada à soma $F(S)$ e envia o sinal de saída:

$$Y = F\left(\sum_{i=1}^{N} X_i w_i + b_0 w_0\right). \tag{12}$$

Uma representação gráfica deste modelo pode ser vista na Figura 8 .

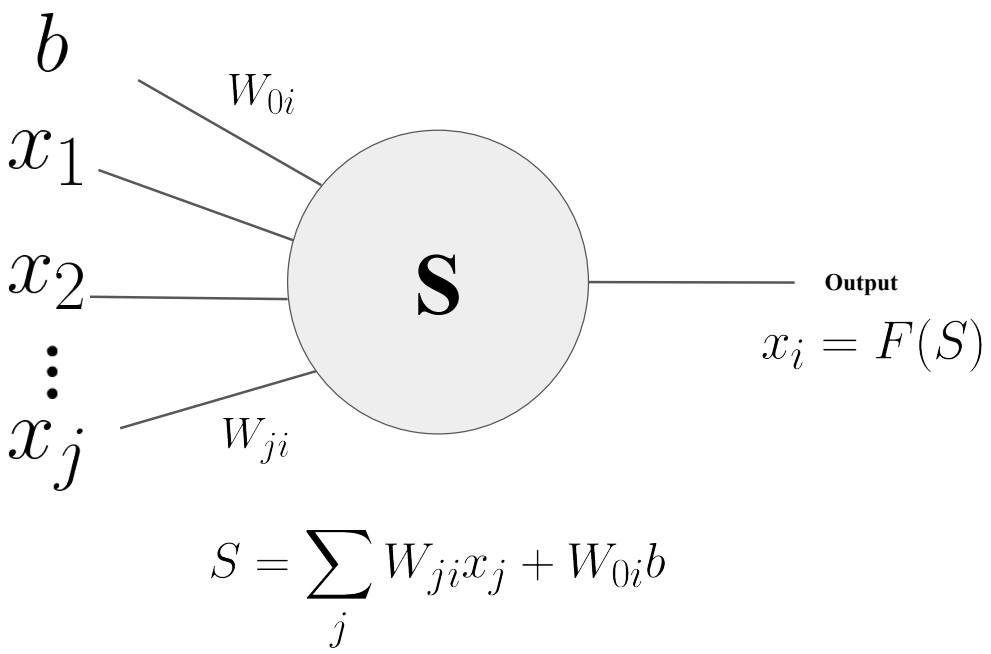


Figura 8 – Modelo matemático de um neurônio artificial (perceptron). Um único neurônio recebe as entradas X_j e retorna o valor $F(\sum_j w_{ji} X_j + b w_{oi})$, onde $F(.)$ é uma função ativação, w é o peso sináptico e b é o bias do neurônio.

Existem diferentes funções de ativação que podem ser aplicadas ao neurônio, as mais comuns são listadas na Tabela 1. Dentro de uma ANN diferentes funções de ativação podem ser utilizadas para cada camada da rede.

De maneira geral todas as ANN podem ser vistas como um conjunto de neurônios. O que irá diferenciar cada tipo de rede é exatamente como esses neurônios são distribuídos e conectados. Neste trabalho iremos explorar algumas arquiteturas específicas das ANN.

Função	Expressão Matemática	Imagem
Linear	$F(S) = kS, k \in \mathbb{R}_+$	$(-\infty, \infty)$
Sigmoid	$F(S) = \frac{1}{1+e^{-\alpha S}}, \alpha \in \mathbb{R}$	$(0, 1)$
Tangente Hiperbólica	$F(S) = \frac{e^{\alpha S}-e^{-\alpha S}}{e^{\alpha S}+e^{-\alpha S}}, \alpha \in \mathbb{R}$	$(-1, 1)$
Função Degrau	$F(S) = \begin{cases} 1, & S \geq 0 \\ 0, & S < 0 \end{cases}$	$\{0, 1\}$
ReLU	$F(S) = \log\left(1 + e^{S}\right)$	$[0, \infty)$

Tabela 1 – Tabela contendo as funções de ativação mais utilizadas na implementação de ANN, onde ReLU é a abreviação para rectified linear unit.

2.1.4.3 Multilayer perceptron (MLP)

A rede MLP (RUMELHART, D. E.; MCCLELLAND, 1986) é a rede mais simples dentro do domínio das ANNs. A MLP contém três tipos de neurônios distribuídos em camadas. Cada tipo de neurônio pode ser descrito como:

1. *Neurônio de entrada*: Os neurônios são compostos por um vetor de entrada que codifica alguma ação ou informação sobre o objeto de estudo. Os neurônios de entrada não realizam qualquer tipo de computação, eles apenas transmitem o vetor de entrada para os neurônios subsequentes. Cada variável fornecida à máquina será um elemento deste vetor.

2. *Neurônio de saída*: Esse neurônio recebe um sinal de outros neurônios e transforma ele utilizando o procedimento descrito na subseção anterior. Os valores após o processamento representam a saída de toda a rede neural. A saída é sempre um vetor onde cada neurônio correspondente a camada de saída fornecerá um elemento do vetor de saída.

3. *Neurônio Escondido*: O neurônio recebe um sinal dos neurônios de entrada ou de neurônios escondidos anteriores a ele, aplica o procedimento descrito na subseção anterior e transmite o novo sinal para neurônios subsequentes, que podem ser neurônios de saída ou escondidos.

As camadas da MLP devem conter, obrigatoriamente, apenas uma camada de entrada e saída. A rede pode conter mais de uma camada escondida, porém, redes com muitas camadas podem ter seu processo de aprendizagem comprometido. Cada neurônio de uma camada da rede, com exceção da camada de entrada, é ligado através de uma sinapse com todos os neurônios da camada anterior. O número de neurônios na camada de entrada é determinado pelo número de *features* do seu problema. Já o número de neurônios da camada de saída irá depender do problema

em si. Não há uma maneira de determinar o número de neurônios de cada camada escondida, a *priori*, sendo necessário a realização de testes empíricos para determiná-los. Um exemplo de MLP é mostrado na Figura 9.

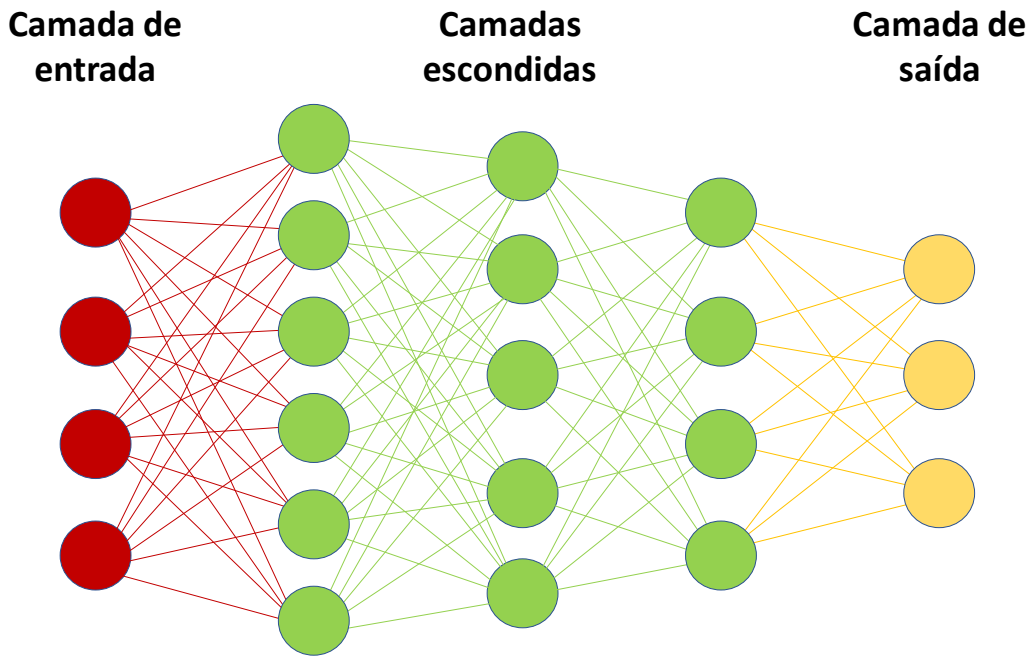


Figura 9 – Modelo esquemático de uma rede neural artificial (ANN) com a arquitetura MPL. A ANN possui uma camada de entrada, três camadas escondidas e uma camada de saída.

Para exemplificar a montagem de uma rede neural podemos considerar o problema de classificar uma fruta entre maçã, laranja, limão e mexerica. Como conjunto de dados possuímos exemplos destas frutas de onde obtivemos o comprimento e a altura e o formato da fruta. Para este caso específico teremos dois neurônios na camada de entrada, um para cada característica, e quatro neurônios na camada de saída, um para cada fruta.

Para denotar as variáveis de entrada iremos utilizar $x_1^{(j)}, ..., x_N^{(j)}$ e as saída (ou alvo) como $y^{(j)}$. O conjunto $\{x_1^{(j)}, ..., x_N^{(j)}, y^{(j)}\}$; $j = 1, 2, ..., m$ é chamado de exemplo de treinamento. Uma lista de m exemplos de treinamento é chamado de conjunto de treinamento ("*training set*"). Consideramos X como o espaço dos valores de entrada e Y como o espaço dos valores de saída. Como produto da ANN teremos uma função $h(X)$, chamada de hipótese, proporcional aos valores de entrada e aos pesos sinápticos da rede. No caso de uma ANN sem camadas escondidas teremos a seguinte expressão

$$h(X) = F\left(\sum_{i=0}^{N} w_i x_i^{(j)} + w_0 b_0\right), \tag{13}$$

em que $x_i^{(j)}$ corresponde ao j-ésimo exemplo da i-ésima variável de entrada e w_i

corresponde ao peso sináptico entre cada neurônio de entrada e saída.

Podemos considerar um caso mais geral de ANN. Definimos como X_s^r o s-ésimo neurônio da r-ésima camada. Desta maneira, uma rede com k_1 neurônios na camada de entrada, $k_2, ..., k_{r-1}$ neurônios nas camadas escondidas 2, ..., $r-1$, respectivamente e k_r neurônios na camada de saída. Desta maneira a ANN terá ao todo r camadas. Cada neurônio de uma camada é totalmente ligada a cada neurônio da camada anterior através de um peso sináptico $w_{s_{r-1},s_r}^{(r-1,r)}$, com $s_r = 1, 2, ..., k_r$ e sendo este o peso sináptico que liga o s_{r-1}-ésimo neurônio da $(r-1)$-ésima camada ao s_r-ésimo neurônio da r-ésima camada. O valor de um neurônio arbitrário na ANN será a função de ativação do mesmo aplicado a soma da multiplicação do valor de cada neurônio da camada anterior e um valor de bias com o peso sináptico associado,

$$x_{s_r}^r = F_r \left(\sum_{s_{r-1}=1}^{k_{r-1}} w_{s_{r-1},s_r}^{(r-1,r)} x_{s_{r-1}}^{r-1} + w_{s_{r-1}}^{r-1} b_r \right), \tag{14}$$

onde b_r corresponde ao bias da r-ésima camada e F_r corresponde a função de ativação da r-ésima camada.

Tão importante quanto o número de camadas e de neurônios em cada camada é a forma com que a rede será treinada. No caso de redes MLP o treinamento se dá de forma supervisionada. O objetivo do treinamento é: dado um conjunto de treinamento, quer-se aprender a função $h(X) : X \to Y$ dado que $h(x^{(j)})$ prediz satisfatoriamente o valor correspondente de $y^{(j)}$. O processo de treinamento se dá a partir da minimização da função custo, $C(w_1, ..., w_n)$, em relação a todos os pesos sinápticos da MLP. Dependendo do problema a ser abordado podemos ter distintas funções custo. Neste trabalho abordaremos as funções referentes a regressão linear e a regressão logística.

Para o caso da regressão linear, a função custo assumirá a forma da soma do erro quadrático médio entre $h(x^{(j)})$ e $y^{(j)}$ para cada conjunto $\{x_1^{(j)}, ..., x_N^{(j)}, y^{(j)}\}$. Para o caso de uma ANN que não possui camadas escondidas e apenas um neurônio na camada de saída, teremos

$$C(w_0, ..., w_j) = \frac{1}{m} \sum_{j=1}^{m} \left(h(x^{(j)}) - y^{(j)} \right)^2 = \frac{1}{m} \sum_{j=1}^{m} \left(w_j x^{(j)} - y^{(j)} \right)^2. \tag{15}$$

Descreveremos a função custo para a regressão logística através da função *cross-entropy* (entropia cruzada).

$$C(w_0, ..., w_j) = -\frac{1}{m} \sum_{j=1}^{m} \left[y^{(j)} \log \left(h(x^{(j)}) \right) + (1 - y^{(j)}) \log \left(1 - h(x^{(j)}) \right) \right]. \tag{16}$$

O processo de treinamento se dará ao minimizar a função custo em relação aos

pesos sinápticos

$$\min_{w_0,...,w_j} C(w_0, ..., w_j). \tag{17}$$

Os algoritmos de treinamento da MLP podem ser divididos em três classes. Os de primeira ordem, que correspondem a algoritmos que não utilizam derivação, apenas avaliam a função em diferentes pontos do espaço, os de segunda classe, que usam a primeira derivada; e os de terceira classe, que utilizam informações referentes a segunda derivada para a análise. Na década de 80 as redes neurais vivenciaram um grande avanço graças a adesão do algoritmo de segunda classe *Backpropagation* (RUMELHART, D. E.; MCCLELLAND, 1986). Este algoritmo foi capaz de realizar o aprendizado com uma velocidade muito superior aos demais algoritmos da época, possibilitando com que as ANN pudessem resolver diversos problemas que antes eram insolúveis. Hoje, possuímos diversos algoritmos de segunda e terceira classe que superam este algoritmo, porém, os algoritmos *Backpropagation* continuam sendo vastamente utilizado.

A ideia central no algoritmo *"Backpropagation"* é de retro-propagar, pelas camadas intermediárias, o erro gerado pelos elementos processados pela camada de saída. Este algoritmo é baseado na abordagem do gradiente descendente e pode ser descrito como:

1. **Configuração inicial:** Selecionar a arquitetura da rede e configurar os pesos sinápticos iniciais. O principal método de configuração inicial dos pesos é escolher pequenos valores com valor médio zero e distribuição uniforme.

2. **Computação feed-forward:** A rede passa o vetor de entrada camada a camada e computa a saída para todos os neurônios da rede, como descrito anteriormente. Após obter o vetor de saída, um vetor de erro é calculado tomando a diferença entre o sinal de saída, Y_i, e o rótulo fornecido pelo conjunto de treinamento, $\hat{Y}_i$

$$E_i = Y_i - \hat{Y}_i. \tag{18}$$

3. **Backpropagation:** Durante este processo os pesos são ajustados para produzir erros menores. Esses ajuste são feitos a partir do sinal de erro. Para cada unidade de saída *j* calculamos $\delta_j^{saída}$,

$$\delta_j^{saída} = e_{ij} F^{'}(S_j), \tag{19}$$

sendo e_{ij} a *j*-ésima componente do vetor erro $\boldsymbol{E_i}$ e $F^{'}$ a derivada da função ativação F. Para o neurônio escondido *j* da camada *l* calcule δ_j^{esc},

$$\delta_j^{esc} = F^{'}(S_j) \sum_k \delta_k^{l+1} w_{jk}, \tag{20}$$

onde a soma é feita sobre todos os neurônios k da próxima camada e w_{jk} é o peso sináptico que conecta os neurônios j e k. Quando todos os δ são calculados, os pesos sinápticos são ajustados, simultaneamente, usando as seguintes equações

$$w_{jk}^{t+1} := w_{jk}^{t} + \Delta w_{jk}^{t}, \tag{21}$$

$$\Delta w_{jk}^{t} = \eta \delta_j a_j^{l-1} + \mu(w_{jk}^{t} - w_{jk}^{t-1}), \tag{22}$$

tanto η como μ são números entre zero e um. Chamamos η de taxa de aprendizado e μ de momento.

A taxa de aprendizado η é o número que determina o quão rápido a ANN se autoajusta aos parâmetros no conjunto de treinamento. Este parâmetro deve ser escolhido com cuidado, pois se for muito pequeno o treinamento será muito lento e se for muito grande o resultado pode divergir. O momento μ irá determinar como os pesos sinápticos anteriores irão influenciar os posteriores. Este termo é importante para prevenir o algoritmo de convergir para mínimos locais. O valor do momento deve ser determinado através de experimentos numéricos.

Os principais problemas da utilização do método de *backpropagation* são listados a seguir:

- Obter o mínimo global da função custo não é garantido pelo método.
- O método pode levar ao *overfitting* nos casos de treinamento muito longo ou quando os exemplos de treino são raros. Podemos citar também, como causa de *overfitting*, um grande número de neurônios nas camadas escondidas.
- Não existe uma maneira exata de determinar se o algoritmo alcançou o mínimo global (melhor solução). Esse problema pode ser contornado reiniciando o algoritmo diversas vezes com leves mudanças e assim aumentar a probabilidade do algoritmo alcançar o mínimo global.

Apesar dos problemas apresentados, o *backpropagation* é vastamente utilizado. Além deste método existem diversos outros, principalmente os de terceira classe, como o algoritmo Quasi-Newton (BROYDEN *et al.*, 1973), Levenberg-Marquart (LEVENBERG, 1944; MARQUARDT, 1963) e o Adam Optimizer (KINGMA; BA, 2014).

Podemos observar o efeito das diferentes funções de ativação sobre o problema exemplo da classificação de frutas presente na Figura 10. Todos os gráficos foram produzidos utilizando duas camadas escondidas e *backpropagation*.

Realizando uma comparação entre todos os modelos de ML apresentados até o momento, podemos perceber o formato gráfico de como cada modelo separa as

diferentes classes do problema de classificação de frutas é bem distinto. A diferença existe até mesmo entre um determinado modelo, como no caso do SVM, Figura 6. Então, como selecionamos o melhor modelo de ML existente? O modelo deve ser selecionado de acordo com o problema abordado, não existe um modelo que performe melhor em qualquer problema. O melhor modelo será aquele em que a maneira como a região de classificação é delimitada se adapte melhor ao conjunto de dados do problema.

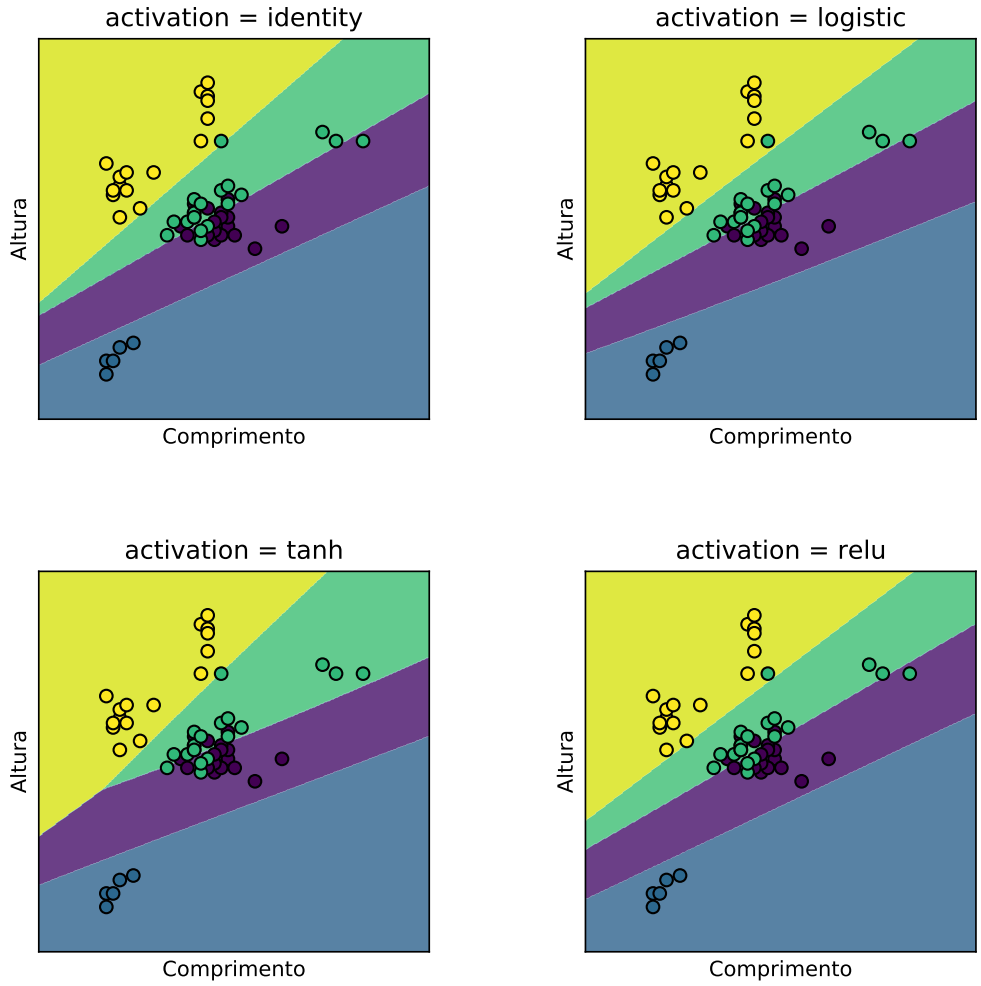


Figura 10 – Algoritmo MLP possuindo duas camadas escondidas com cem e duzentos neurônios, respectivamente, com diferentes funções de ativação. Os pontos são os exemplos contidos no conjunto de treinamento. Para o conjunto de teste foi utilizado uma grade de dados do espaço de *features* e são representados pelas áreas hachuradas. Cada cor corresponde a uma fruta diferente (amarelo = mexerica, verde = maçã, roxo = laranja, azul = limão).

2.2 ENSEMBLE LEARNING

Ensemble learning é um paradigma de ML onde múltiplos modelos (*weak-learners*) são treinados para resolver o mesmo problema e combinados para melhorar a acurácia, generalidade e robustez em relação a um único *weak-learner* presente no ensemble. Os métodos de *ensemble learning* podem ser divididos em duas grandes categorias: Homogêneos, no qual apenas um tipo de modelo de ML é utilizado para se construir o ensemble; e Heterogêneo, no qual diversos modelos de ML são utilizados para se construir o ensemble. Para a categoria homogênea podemos ter duas principais metodologias a serem utilizadas, Bagging e Boosting.

2.2.1 Bagging

A idéia por trás do bagging (BREIMAN, 1996) é fitar diversos modelos independentemente e tomar a média de suas predições para obter um modelo com menor variância. Na prática, fitar diversos modelos independentemente necessitaria um conjunto de dados muito grande. Para contornar isso utilizamos técnicas de amostragem, como bootstraping, para gerar conjuntos de dados e criar modelos que são quase independentes.

Bootstraping (LIEW, 2008) é uma técnica estatística de amostragem de conjunto de dados através de substituição. O novo conjunto de dados é chamado *bootstrap*. A técnica consiste em estimar as propriedades de um estimador de variáveis aleatórias (média, variância, etc) e é uma boa alternativa para a inferência estatística. Vamos supor que geramos a partir de um conjunto de treinamento X_{train} um novo conjunto bootstrap, $X^{'}_{train}$, com o mesmo número de observação, ou seja, $M = M^{'}$. Como o conjunto de dados foi amostrado com substituição, algumas observações podem ser repetidas e algumas outras podem estar faltando. O valor esperado para a relação de observações únicas entre X_{train} e $X^{'}_{train}$ é aproximadamente 63%. Assumindo que possuímos L amostras bootstrap (aproximação de L conjuntos de dados independentes) de tamanho B,

$$\begin{gathered}\{z_1^1, z_2^1, ..., z_B^1\},\\ \{z_1^2, z_2^2, ..., z_B^2\},\\ \vdots\\ \{z_1^L, z_2^L, ..., z_B^L\},\end{gathered} \tag{23}$$

onde z_b^l é a b-ésima observação da l-ésima amostra bootstrap, podemos então treinar L *weak-learners* aproximadamente independentes,

$$h_1(.), h_2(.), ..., h_L(.). \tag{24}$$

Agregamos cada um dos *weak-learners* através de uma métrica de média, S_L, para criar um modelo de ensemble que possui uma variância menor que os *weak-learners* que o compõem. Para o caso da classificação utilizamos como S_L o voto majoritário, ou seja, para cada observação dentro do conjunto de treinamento a classe escolhida será a que o maior número de *weak-learners* escolheu. No caso de regressão podemos usar como métrica a média do valor de cada um dos *weak-learners*,

$$S_L^{regressao} = \frac{1}{L}\sum_{l=1}^{L} h_l(.). \tag{25}$$

A grande vantagem em se utilizar o método bagging está na independência do treinamento de cada weak-learner, de maneira que este processo pode ser realizado em paralelo. O algoritmo que melhor representa este método é o *Random Forest*.

2.2.1.1 *Random Forest*

No algoritmo *Random Forest*, cada árvore de decisão dentro do ensemble é construído amostrando através de substituição (como por exemplo bootstrap) do conjunto de treinamento. Cada árvore de decisão é construída de forma a utilizar um número máximo de features presentes no conjunto de dados. Estas características podem ser vistas como fontes de aleatoriedade do problema.

Como cada árvore de decisão geralmente apresenta uma alta variância e tende para o *overfitting*, a função das duas fontes de aleatoriedade é diminuir a variança do algoritmo. A aleatoriedade injetada de alguma maneira desacopla o erro da predição, e ao tomar a média das previsões, esse erro é cancelado.

Podemos verificar na Figura 11 o comportamento do algoritmo sobre o nosso problema exemplo de classificação de frutas. Realizando uma comparação com a Figura 3, que consiste nos modelos gerados pelas árvores de decisão. Podemos perceber que o algoritmo de *Random Forest* possui uma maneira de divisão das classes parecida. Vale ressaltar que o *Random Forest* consegue alcançar uma maior acurácia e um processo de treinamento mais rápido que a do algoritmo de árvore de decisão.

2.2.2 Boosting

O algoritmo de Boosting possui quase a mesma metodologia do Bagging, construímos uma família de *weak-learners* e então os agregamos para obter o ensemble que performa melhor do que os *weak-learnres* que o compõem. O Boosting se

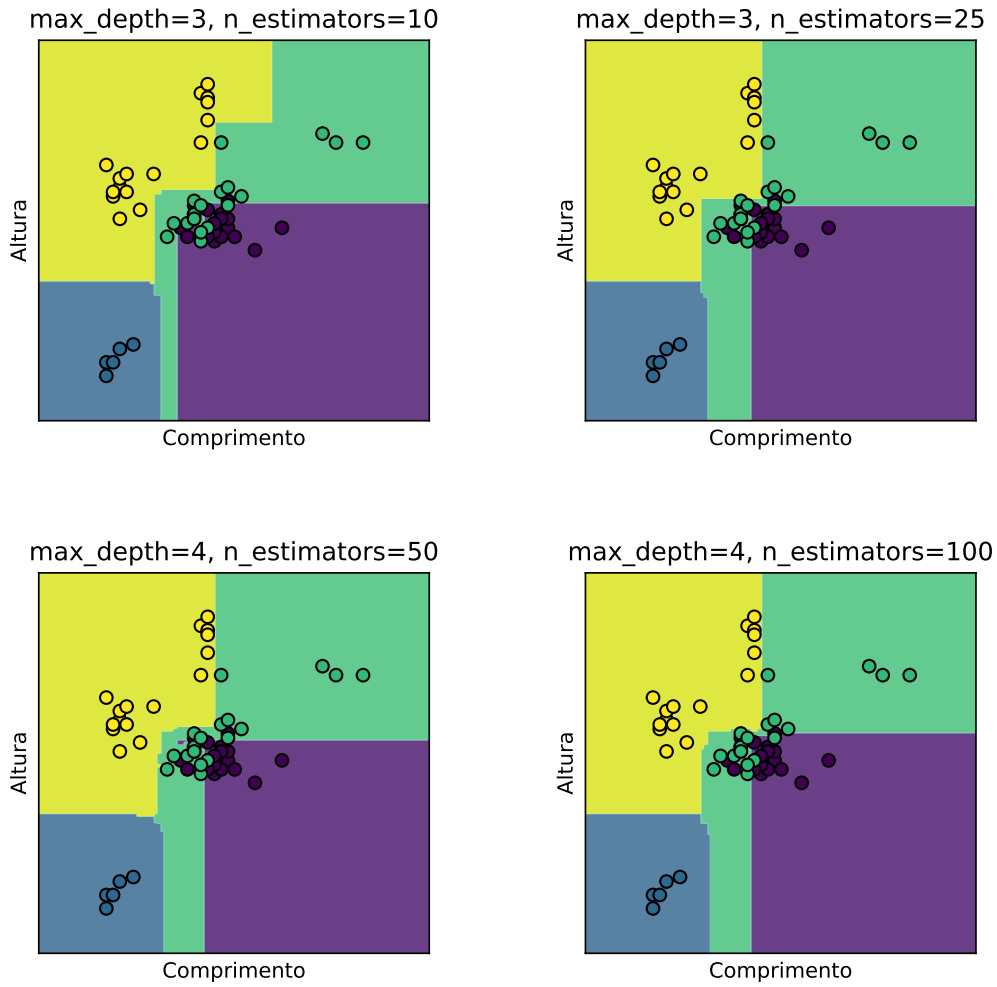


Figura 11 – Algoritmo *Random Forest*, com diferentes profundidades de *Decision Tree* e número de árvores que compõem o ensemble, para o problema de classificação de frutas. Os pontos são os exemplos contidos no conjunto de treinamento. Para o conjunto de teste foi utilizado uma grade de dados do espaço de *features* e são representados pelas áreas hachuradas. Cada cor corresponde a uma fruta diferente (amarelo = mexerica, verde = maçã, roxo = laranja, azul = limão). Os atributos max_depth significa a profundidade máxima de cada árvore de decisão, e n_estimators é o número de *weak-learners* que compõem o ensemble.

diferencia do Bagging na construção do ensemble e na forma de treinamento dos *weak-learners*.

Diferentemente do Bagging, o método de boosting utiliza um treinamento em série de seus *weak-learners*, onde cada novo modelo possui um foco nas instâncias mais difíceis do conjunto de treinamento. A escolha do modo em que o treinamento em sequência e a agregação dos *weak-learners* será realizada dependerá do algoritmo de boosting escolhido. Os dois príncipais algoritmos de boosting são Adaboosting (Adaptative boosting) e gradient boosting.

2.2.2.0.1 Adaboosting

No algoritmo Adaboosting nós definimos nosso modelo de ensemble como uma soma ponderada de *L weak-learners*,

$$S_L(.) = \sum_{l=1}^{L} w_l h_l, \tag{26}$$

em que w_l são os pesos associados a cada *weak-learner* e h_l são os *weak-learners*. Encontrar os melhores pesos e os melhores *weak-learners* pode ser um processo muito complexo para realizar em uma optimização única. Sendo necessário a utilização de um processo de optimização iterativo, que é matematicamente mais tratável, mesmo que este possa levar para um mínimo local. Basicamente o algoritmo irá adicionar os *weak-learners* um a um, procurando em cada interação o melhor possível par (peso, weak-learner) para adicionar ao modelo de ensemble. Desta maneira, definimos recorrentemente o ensemble como,

$$s_l(.) = s_{l-1} + w_l h_l, \tag{27}$$

em que w_l e h_l são escolhidos de maneira que $s_l(.)$ é o modelo que melhor se adapta ao conjunto de treinamento, e desta maneira, é o melhor melhoramento possível sobre $s_{l-1}(.)$. Podemos então denotar,

$$(w_l, h_l) = \arg\min_{w,h(.)} E(s_{l-1} + w_l h_l) = \arg\min_{w,h(.)} \sum_{n=1}^{N} L(y_n, s_{x_n} + w_l h_l(x_n)), \tag{28}$$

onde $E(.)$ é o erro de ajuste do modelo dado e $L(.)$ é a função custo. Assim, ao invés de otimizar globalmente sobre todos os *L weak-learners* da soma, nós nos aproximamos do mínimo global optimizando localmente, construindo e adicionando *weak-learners* ao ensemble um a um.

Este algorítimo se difere da técnica de *Bagging* principalmente pela forma pelo qual o ensemble é construído. Enquanto no *Bagging* o ensemble é construído com *weak-learners* que são treinados paralelamente, no *Adaboosting* os treinamentos dos *weak-learners* são realizados de forma sequencial, onde o resultado de um *weak-learner* irá influenciar o treinamento do próximo através do *resample* do conjunto de treinamento. O comportamento do algoritmo utilizando *weak-learners* baseados em árvores de decisão pode ser observado, para o problema de classificação de frutas, na Figura 12 para diferentes números de *weak-learners* que compõem o ensemble.

2.2.2.0.2 Gradient boosting (GBoosting)

Assim como no agoritmo Adaboosting o modelo de ensemble dentro do gradient boosting também é uma soma ponderada dos *weak-learners* que o compõem

$$s_L(.) = \sum_{l=1}^{L} w_l h_l. \tag{29}$$

A principal diferença no gradient boosting e do Adaboosting está no processo de optimização sequêncial. No caso do gradient boosting a optimização é realizado com o gradiente descendente. Em cada iteração nós fitamos um *weak-learner* com o gradiente do erro em relação ao atual modelo de ensemble. Matematicamente, temos

$$s_l(.) = s_{l-1} - w_l \nabla E_{s_{l-1}}(.), \tag{30}$$

onde $E(.)$ é o erro produzido por dado modelo, w_l é um peso associado ao passo e $-\nabla E_{s_{l-1}}(.)$ é o oposto do gradiente em respeito ao modelo ensemble do passo $l-1$. Podemos descrever o algoritmo atráves do seguinte pseudo código:

1. Iniciamos o modelo de ensemble com o valor constante:

$$s_0 = \arg\min_{w_0} \sum_{n=1}^{N} L(y_n, w_0). \tag{31}$$

2. para l=1 até L:

   a) Calcule $r_{l,n} = -\nabla E_{s_{l-1}}(.)$,

$$r_{l,n} = -\left[\frac{\partial L(y_n, s_{l-1}(x_n))}{\partial s_{l-1}(x_n)}\right] \text{ para } i = 1, ..., N. \tag{32}$$

   b) Treine um *weak-learner* h_l em relação ao conjunto de treinamento $\{(x_n, r_{l,n})\}$.

c) Compute o peso w_l solucionando o seguinte problema de optimização:

$$w_l = \arg\min_{w} \sum_{n=1}^{N} L(y_n, s_{l-1} + w_l h_l(x_n)). \tag{33}$$

d) Atualize o modelo

$$s_l = s_{l-1} + w_l h_l(x_n). \tag{34}$$

3. Produza o modelo final:

$$F_L(x_n) = \sum_{l=1}^{L} w_l h_l. \tag{35}$$

Podemos ver o comportamento do algoritmo, utilizando como *weak-learner* árvores de decisão, na Figura 12.

Como os algoritmos GBoosting e Adaboosting são ambos algoritmos de boosting é possível listar algumas diferenças fundamentais entre os dois algoritmos:

- **Função Custo:** No caso do Adaboosting a função custo utilizada é a exponencial, fazendo com que o algoritmo seja sensível a *outliers* [1]. Já o GBoosting tem a liberdade de utilizar qualquer função custo, o que gera uma maior robustez em relação aos *outliers*.
- **Benefícios:** O Adaboosting minimiza a função custo relacionada a qualquer erro de classificação e é melhor usado com weak-learners. O modelo foi desenhado principalmente para problemas de classificação binária. O algoritmo Gboosting foi desenvolvido para para solucionar problemas com função custo diferenciáveis e pode ser usado tanto para classificação quanto pra regressão.
- **Deficiências:** As deficiências do conjunto de dados (regiões onde existe um erro maior) são identificadas de maneiras diferentes. No Adaboosting elas são identificadas aumentando o peso relacionado a esses pontos. No caso do Gboosting elas são identificadas pelos gradientes.

2.2.3 Automated Machine Learning.

Automated Machine Learning (AutoML) consiste na automação de alguns ou, até mesmo, todas as tarefas presentes na implementação de um algoritmo de ML:

- Engenharia de *features*;

[1] outilers são dados que se diferem drasticamente do conjunto de dados ao qual eles pertencem, ou seja, são dados que fogem da normalidade e podem causar anomalias os resultados

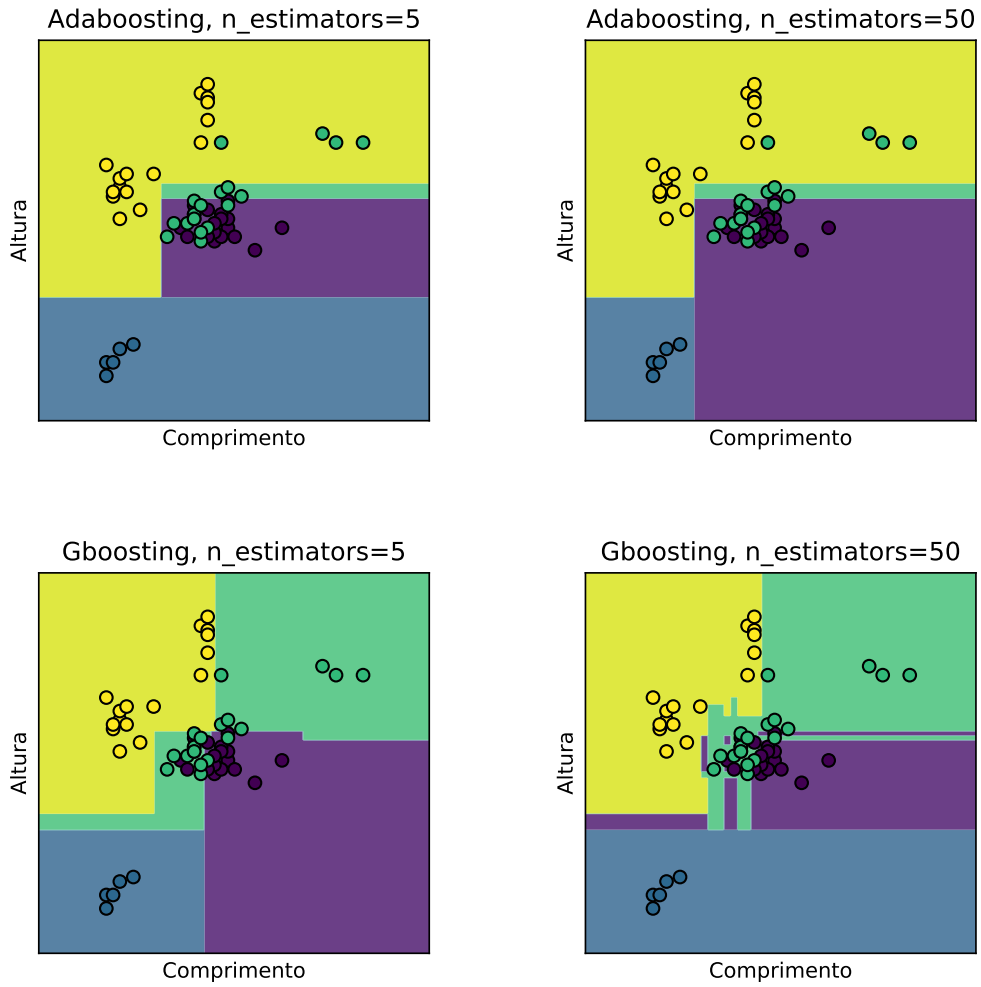


Figura 12 – Algoritmo Adaboosting e Gboosting com diferentes números de *weak-learners* aplicados ao problema de classificação de frutas. Os pontos são os exemplos contidos no conjunto de treinamento. Para o conjunto de teste foi utilizado uma grade de dados do espaço de *features* e são representados pelas áreas rachuradas. Cada cor corresponde a uma fruta diferente (amarelo = mexerica, verde = maçã, roxo = laranja, azul = limão). A variável *n*_estimator nos diz quantos *weak-learners* foram utilizados na construção do ensemble.

- Optimização de hiper-parâmetros;
- Criação de ensembles;
- Validação de modelos.

Este problema de automação foi formulado matematicamente por (THORNTON *et al.*, 2012; FEURER *et al.*, 2015a) e é conhecido como *combined algorithm selection and hyperparameter optimization problem* (CASH). Dado um conjunto de algoritmos $\mathcal{A} = \{A^{(1)}, A^{(2)}, ..., A^{(k)}\}$ com um domínio de hiper-parâmetros associados $\Lambda^{(1)}, ..., \Lambda^{(k)}$, uma métrica de função custo $L(.,.,.)$ e um conjunto de dados $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$, onde $\mathcal{X} = \{x_i, ..., x_n\}$ e $\mathcal{Y} = \{y_i, ..., y_n\}$, definimos o problema CASH como sendo a optimização desses parâmetros, ou seja,

$$A^*, \lambda^* = \arg \min_{A^{(j)} \in \mathcal{A}, \lambda \in \Lambda^{(j)}} \frac{1}{k} \sum_{i=1}^{k} L(A_\lambda^{(j)}, \mathcal{D}_{train}, \mathcal{D}_{valid}). \tag{36}$$

Para solucionar o problema do autoML um espaço de configuração contendo as possíveis combinações dos algoritmos e hiper-parâmetros é definido. No caso de redes neurais, uma dimensão extra representada pela arquitetura é adicionada ao espaço de procura. Para procurar sobre esse espaço, diferentes técnicas podem ser utilizadas (HE *et al.*, 2021):

- **Optimização Bayesiana:** É um processo de optimização iterativa adequado para funções custo caras computacionalmente. Ele consiste de dois componentes principais: (i) modelos *surrogate* para modelar a função custo e (ii) uma função de aquisição que mede o valor que será gerado pela validação da função custo em um novo ponto (SNOEK *et al.*, 2012). Esse método também é utilizado na biblioteca *Sequential Model-based Algorithm Configuration* (SMAC), que permite utilizar processos Gaussianos e *Random Forest* como modelos *surrogate* (HUTTER *et al.*, 2011).
- **Grade de pesquisa:** Cria uma grade de configurações e procura por ela. A grande vantagem deste método é que ele pode ser facilmente paralelizado.
- **Programação genética:** É uma técnica inspirada pelo processo da seleção natural, onde conceitos como cromossomos e mutações são usados para desenvolver melhores gerações de soluções para o problema.

Existem diversas projetos *Open source* que são comumente utilizados como TPOT (LE *et al.*, 2019), auto-sklearn (FEURER *et al.*, 2015b), AutoKeras (JIN *et al.*, 2019), Auto-Pytorch (MENDOZA *et al.*, 2018), entre outros.

A principal vantagem da utilização de métodos de AutoML se dá para problemas supervisionados, pois além de encontrar o melhor modelo e realizar o *tuning* dos hiperparâmetros dos mesmo, o método é capaz de encontrar os melhores tratamentos dos dados para o seu problema. A grande desvantagem está no aumento do problema de optimização, o que demanda mais tempo e recursos computacionais.

O método de AutoML será utilizado no Capítulo três da tese, onde possuímos um problema de classificação e regressão supervisionado. Vale resaltar que modelos de AutoML utiliza a técnica de ensemble learning para melhorar os resultados apresentados, assim como a base de seus *weak-learners* podem ser qualquer modelo de ML apresentado neste Capítulo.

3 CLASSIFICADOR DE EMARANHAMENTO BASEADO EM MACHINE LEARNING.

Neste capítulo iremos discutir o trabalho de criação de um classificador de emaranhamento utilizando técnicas de ML. O capítulo é dividido em cinco partes principais. A primeira corresponde à exposição do problema de classificação de emaranhamento, a segunda sobre os critérios de separabilidade utilizados para criar as classes de classificação, a terceira a metodologia de criação dos conjuntos de dados, a quarta os resultado obtidos na utilização de diversos modelos de ML e a quinta a conclusão do trabalho. Este capítulo foi publicado na revista *Quantum Information Processing* com o título *Automated Machine Learning can Classify Bound Entangled States with Tomograms*.

3.1 EMARANHAMENTO E SEPARABILIDADE

O emaranhamento é um dos principais fundamentos da mecânica quântica. No campo da Informação Quântica (IQ) o emaranhamento tem papel central em diversas implementações de protocolos quânticos. Tais protocolos tem a capacidade de resolver tarefas de maneira que não possuem equivalentes em suas contrapartidas na teoria de informação clássica. Inúmeros protocolos, como codificação superdensa (HARROW *et al.*, 2004), teleporte quântico (FURUSAWA, 1998; OLMSCHENK *et al.*, 2009; MILBURN; BRAUNSTEIN, Samuel L., 1999), código quântico de correção de erros (BENNETT *et al.*, 1996) e criptografia quântica (URSIN *et al.*, 2007; VAZIRI *et al.*, 2002; GRÖBLACHER *et al.*, 2006), têm sido implementados experimentalmente utilizando o emaranhamento como o seu principal recurso. No campo da computação quântica, o emaranhamento como principal recurso ainda é uma questão em aberto (PREVEDEL *et al.*, 2007; JOZSA; LINDEN, 2003).

3.1.1 Formalização do problema

A definição de emaranhamento é realizada através da definição de separabilidade. Consideramos que um estado quântico está emaranhado quando ele não for separável. Desta maneira, através do quarto postulado da mecânica quântica (NIELSEN; CHUANG, 2000) definimos um estado puro separável como:

Seja $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B \cong \mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2}$, onde $d_1, d_2 \geq 2$ são as dimensões de cada sub-espaço. Um estado puro $|\psi\rangle \in \mathcal{H}$ composto por duas partículas, $|\psi_A\rangle \in \mathcal{H}_A$ e $|\psi_B\rangle \in \mathcal{H}_B$, é chamado separável se e somente se ele puder ser escrito na forma:

$$|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle\,, \tag{37}$$

caso o contrário o estado é chamado de emaranhado.

Para um estado misto devemos utilizar o formalismo de matriz densidade. Um estado qualquer (puro ou misto) $\rho \in \mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2}$ é dito separável se e somente se ele puder ser escrito como:

$$\rho = \sum_{i=1}^{k} p_i \rho_i^A \otimes \rho_i^B, \tag{38}$$

onde $\{p_i\}$ é uma distribuição de probabilidade e obedece as seguintes restrições $p_i \geq 0$ e $\sum_i p_i = 1$, e ρ_i^A e ρ_i^B são estados (puro ou misto) do sub-sistema A e B.

Através desta definição, o problema de classificar o estado como emaranhado se torna um problema de classificar o estado como separável. Determinar se um estado arbitrário é separável ou não pode ser um problema complicado, já que a dimensão do espaço de Hilbert cresce exponencialmente com o número de sub-sistemas envolvidos e com a dimensão de cada sub-sistema. Devido ao crescimento exponencial podemos classificar este problema, através da complexidade computacional, como sendo um problema NP-difícil (GURVITS, 2004).

Por fim, podemos formalizar o problema de identificar a separabilidade de um sistema quântico como sendo um problema de decisão. Seja $\rho \in \mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2}$ um estado misto, o problema se reduz a dado uma matriz arbitrária $\rho_{d_1 \times d_2}$ decidir se o estado é separável. Para isso foram criados critérios para identificar a separabilidade de maneira eficiente em determinados casos.

3.2 CRITÉRIOS DE SEPARABILIDADE.

Devido à importância do problema de separabilidade para a comunidade da Informação Quântica, uma série de esforços para resolver o problema de maneira eficiente foram desenvolvidos. Tendo em vista o crescimento exponencial da dificuldade do problema de separabilidade, um critério único e eficiente de separabilidade de um sistema quântico ainda não existe. Porém, para recortes específicos do problema possuímos critérios bem estabelecidos que são implementados de maneira eficiente.

3.2.1 Decomposição de Schmidt

Para estados puros bipartidos podemos utilizar um método baseado na decomposição de Schmidt para determinar se um sistema é separável ou não.

Teorema 3.2.1. ***(Decomposição de Schmidt)****: Para todo $|\psi\rangle \in \mathcal{H}$ existe uma base ortonormal $|a_i\rangle$ e $|b_i\rangle$ de forma que*

$$|\psi\rangle = \sum_i \lambda_i |a_i\rangle \otimes |b_i\rangle , \tag{39}$$

com $\lambda_i \geq 0$ e $\sum_i \lambda_i^2 = 1$.

O coeficiente λ_i é conhecido como coeficiente de Schmidt. O número de coeficientes não nulos é chamado de rank de Schmidt $S(\psi)$ de um estado $|\psi\rangle$. O critério de separabilidade nos diz que o estado é dito separável se e somente se $S(\psi) = 1$

Prova: Tomando a expansão de um estado puro genérico, teremos:

$$|\psi\rangle = \sum_{ij} \psi_{ij} |ij\rangle , \tag{40}$$

onde $\Psi = \{\psi_{ij}\}$ é uma matriz de números complexos. Tomando a decomposição de valores singulares, toda a matriz Ψ pode ser escrita como $\Psi = UDV$, com U e V sendo matrizes unitárias e D uma matriz diagonal com elementos não negativos. Desta maneira, a expansão do estado puro ficará na forma:

$$|\psi\rangle = \sum_{ijk} U_{ki} D_{ii} V_{ij} |kj\rangle \tag{41}$$

Renomeando os estados como

$$\begin{aligned} |a_i\rangle &= \sum_i U_{ki} |k\rangle \\ |b_i\rangle &= \sum_i V_{ij} |j\rangle \end{aligned} \tag{42}$$

teremos então o mesmo estado descrito na Equação 39 com $\lambda_i = D_{ii}$. A ortogonalidade dos estados $|a_i\rangle$ e $|b_i\rangle$ segue do fato de que U e V são unitárias. Desta maneira para calcular se um estado puro é separável ou não, basta calcular o posto de Ψ. Podemos perceber também que o caso no qual o emaranhamento é máximo ocorrerá quando todos os autovalores, λ_i, forem iguais, ou seja, quando

$$|\psi\rangle = \frac{1}{\sqrt{dim\mathcal{H}_A}} \sum_{i=1}^{dim\mathcal{H}_A} |a_i\rangle \otimes |b_i\rangle . \tag{43}$$

3.2.2 Critério de Perez-Horodecki (critério PPT)

Para matrizes densidade, um dos critérios mais utilizados é o critério de Perez-Horodecki (PERES, 1996; HORODECKI, M. *et al.*, 1996). Considerando que o operador matriz densidade é um operador positivo semi-definido, ou seja, não possui autovalores negativos, e a operação de transposição, T, é um mapa positivo, que leva operadores positivos em operadores positivos, desta forma

$$T : (O \geq 0) \longmapsto (O^{'} \geq 0), \tag{44}$$

O critério irá se basear em tomar a transposta parcial de um dos sub-sistemas da matriz densidade e será definido através do seguinte teorema:

Teorema 3.2.2. *(PERES, 1996; HORODECKI, M.* et al.*, 1996) Se um estado,* $\rho \in \mathcal{H}_A \otimes \mathcal{H}_B$*, é separável, então*

$$\rho^{T_B} := (\mathbb{I}_A \otimes T_B)[\rho] \geq 0 \tag{45}$$

onde T_B *é a transposição do segundo sub-sistema e* $\mathbb{I}_A$ *é a identidade aplicada ao primeiro sub-sistema. Caso contrário se* $\rho^{T_B} := (\mathbb{I}_A \otimes T_B)[\rho]$ *for negativo, ou seja, possuir autovalores negativos, o estado é considerado emaranhado. Por esse motivo o critério pode ser chamado de critério da transposta parcial positiva (positive partial transposte, PPT)*

Em dimensões arbitrárias de dois sub-sistemas ou em sistemas contendo um número arbitrário de sub-sistemas (para $dim\mathcal{H} > 6$) o critério PPT fornece uma condição necessária mas não suficiente para separabilidade, ou seja, para dimensões maiores existem estados que permanecem positivos sobre a transposição parcial mesmo que eles sejam emaranhados (HUBER *et al.*, 2018). Chamamos tais estados de estados emaranhados presos. Desta maneira temos o seguinte teorema:

Teorema 3.2.3. *(PERES, 1996; HORODECKI, M.* et al.*, 1996) Um estado* $\rho \in \mathcal{H}_A \otimes \mathcal{H}_B$ *com* $dim\mathcal{H}_A \otimes dim\mathcal{H}_B \leq 6$ *é separável se e somente se*

$$\rho^{T_B} := (\mathbb{I}_A \otimes T_B)[\rho] \geq 0 \tag{46}$$

3.2.2.1 Emaranhamento Preso

O emaranhamento preso é caracterizado por estados emaranhados que não podem ser destilados. A destilação de emaranhamento tem como objetivo transformar estados mistos ρ em estados maximamente emaranhados utilizando apenas operações locais e comunicação clássica (LOCC, "*local operation and classical communication*") (HORODECKI, M. *et al.*, 1998). O fato dos estados presos não poderem ser destilados implica em importantes consequências na teoria da Informação Quântica, como a irreversibilidade mecânica de uma operação quântica (YANG *et al.*, 2005). Essa irreversibilidade mecânica se dá quando a partir de um estado puro produzimos um estado misto com emaranhamento preso, uma vez criado o estado ele não pode ser novamente destilado para o estado puro. Um problema em aberto dentro de estados emaranhados presos é o de que ainda não se sabe o limite e tamanho de seu espaço dentro do espaço de Hilbert (DAS, S. *et al.*, 2017).

Para o caso de dois qutrits (sistema contendo três estados quânticos mutualmente ortonormais, $\{|0\rangle, |1\rangle, |2\rangle\}$) o espaço de Hilbert possui $dim(\mathcal{H} = 9)$ e desta maneira possui estados separáveis, emaranhados e emaranhados presos. Porém, como

dito anteriormente, o tamanho do conjunto no espaço de Hilbert é desconhecido. Até o presente momento possuímos na literatura apenas algumas "famílias" de estados conhecidos (HORODECKI, P. *et al.*, 1999; ACIÉN *et al.*, 2000; HALDER *et al.*, 2019; TERHAL, 2001), que possuem os limites de cada conjunto bem definidos. Podemos citar como exemplo os estados do tipo Horodecki (HORODECKI, P. *et al.*, 1999), família de apenas um parâmetro. Os estados de Horodecki podem ser obtidos a partir da seguinte expressão:

$$\rho_\alpha^{Horo} = \frac{2}{7}\left|\phi_+^3\right\rangle\!\left\langle\phi_+^3\right| + \frac{\alpha}{7}\sigma_+ + \frac{5-\alpha}{7}\sigma_-, \quad 2 \leq \alpha \leq 5, \tag{47}$$

onde $\left|\phi_+^3\right\rangle$ é um estado maximamente emaranhado de dois qutrits,

$$\left|\phi_+^3\right\rangle = \frac{1}{\sqrt{3}}(|00\rangle + |11\rangle + |22\rangle), \tag{48}$$

e

$$\sigma_- = \frac{1}{3}(|01\rangle\langle 01| + |12\rangle\langle 12| + |20\rangle\langle 20|), \tag{49}$$

$$\sigma_+ = \frac{1}{3}(|10\rangle\langle 10| + |21\rangle\langle 21| + |02\rangle\langle 02|). \tag{50}$$

Para este estado a divisão entre emaranhado, emaranhado preso e separável é bem conhecida e obedece o seguinte critério

$$\rho_\alpha^{Horo} = \begin{cases} \text{separável}, & 2 \leq \alpha \leq 3, \\ \text{emaranhado preso}, & 3 < \alpha \leq 4, \\ \text{emaranhado}, & 4 < \alpha \leq 5. \end{cases} \tag{51}$$

3.2.3 Robustez Generalizada de Emaranhamento (*Generalized Robustness of Entanglement* (GRE)).

Outro critério de emaranhamento utilizado para a classificação de estados é a robustez de emaranhamento. Esta quantifica a quantidade de mistura com um estado separável que é necessária para destruir o emaranhamento do sistema (VIDAL, Guifré; TARRACH, 1999a). Estamos interessados em uma medida um pouco mais geral, a robustez generalizada de emaranhamento, que quantifica a quantidade de mistura que qualquer estado necessita para destruir o emaranhamento (STEINER, 2003a). Formalmente podemos definir a GRE como:

Considerando um estado de *n*–partículas $\rho \in \mathcal{D}(\mathbb{C}^{A_1} \otimes \cdots \otimes \mathbb{C}^{A_n})$, e outro estado, ρ_S, nós chamamos a robustez generalizada de emaranhamento de um estado ρ o menor valor de $s \geq 0$, de maneira que o estado,

$$\rho(s) = \frac{1}{1+s}(\rho + s\rho_s) \tag{52}$$

seja separável. O parâmetro *s* deve ser zero se o estado ρ é separável, e é sempre finito, já que o conjunto de estados separáveis esta em torno da identidade (VIDAL, Guifré; TARRACH, 1999a). O valor mínimo de *s* será encontrado na borda do conjunto de estados separáveis. Desta maneira o estado ρ_s é o estado no qual o emaranhamento de ρ é mais sensível.

3.2.4 Testemunha de emaranhamento

Os critérios de separabilidade apresentados nas seções anteriores são ferramentas que possibilitam identificar se há emaranhamento em determinado estado quântico. Uma abordagem que pode ser aplicada a qualquer sistema composto, em qualquer dimensão, assim como para sistemas multi-partidos, é a testemunha de emaranhamento. Como o próprio nome indica, a testemunha de emaranhamento é um observável que testemunha o emaranhamento de um sistema (TERHAL, 2002). De maneira simples, o critério nos diz que para uma dada matriz densidade ρ deve existir um observável *W* tal que,

$$\mathrm{Tr}(W\rho) \geq 0 \tag{53}$$

se existir emaranhamento, em ρ, a desigualdade será violada. A testemunha de emaranhamento não tem a capacidade de detectar todos os estados emaranhados, de modo que se a desigualdade for violada, podemos garantir que o estado é emaranhado, porém, se a desigualdade não for violada, nada podemos concluir sobre o estado. Formalizando esta questão, teremos que

$$\begin{cases} \mathrm{Tr}(W\rho) \geq 0, & \text{não podemos concluir nada sobre o estado,} \\ \mathrm{Tr}(W\rho) < 0, & \text{o estado é emaranhado.} \end{cases} \tag{54}$$

As testemunhas de emaranhamento possuem sua origem na análise geométrica do espaço de estados da matriz densidade e explora a convexidade do mesmo. Uma caracterização geométrica de quando $\rho \in \mathcal{D}$ está contida em um certo subconjunto convexo $\mathcal{C} \subset \mathcal{D}$ é dado pelo teorema de Hahn-Banach (LAX, 2002; ROCKAFELLAR, 1970).

Teorema 3.2.4. *(LAX, 2002; ROCKAFELLAR, 1970) (Teorema de Hahn-Banach) Seja $\mathcal{C} \subset \mathcal{D}$ um conjunto convexo de estados no espaço de Hilbert $\mathcal{H}$, então para cada $\rho \in \mathcal{C}$ deve existir um operador Hermitiano $W \in B(\mathcal{H})$ de maneira que*

$$\text{Tr}(W\rho) < 0 \; e \; \text{Tr}(W\sigma) \geq 0 \tag{55}$$

para todo $\sigma \notin \mathcal{C}$.

O teorema nos diz que um conjunto convexo e um ponto do lado de fora podem ser separados por um hiperplano W. Este hiperplano é a nossa testemunha de emaranhamento (ela testemunha estados $\rho \notin \mathcal{C}$).

Corolário 3.2.4.1. *Um estado* $\rho \in \mathcal{D}$ *é separável se e somente se* $\text{Tr}(\rho W) \geq 0$ *para todo operador Hermitiano W de tal modo que*

$$\text{Tr}([|\psi_A\rangle\langle\psi_A| \otimes |\psi_B\rangle\langle\psi_B|]W) \geq 0, \tag{56}$$

com $|\psi_A\rangle \in \mathcal{H}_A$ *e* $|\psi_B\rangle \in \mathcal{H}_B$*. O operador W é nomeado testemunha de emaranhamento.*

Uma representação esquemática do aspecto geométrico do formalismo de testemunha de emaranhamento pode ser observada na Figura 13. Seja $\mathcal{S}$ e $\mathcal{E}$ conjuntos de estados separáveis e emaranhados respectivamente, sendo que $\mathcal{S} \subset \mathcal{E}$ e ρ uma matriz densidade que representa um estado contido em $\mathcal{E}$. Podemos, através do teorema de Hahn-Banach, construir um hiperplano W de maneira a separar $\mathcal{S}$ e $\mathcal{E}$. Porém, esta separação não irá nos fornecer todos os estados emaranhados. Este problema pode ser contornado realizando uma otimização do operador W, fazendo com que seja obtido um hiperplano que esteja na borda do conjunto dos estados separáveis $\mathcal{S}$. Para conseguirmos isolar completamente o conjunto dos separáveis $\mathcal{S}$ será necessário a criação de inúmeros hiperplanos optimizados de maneira a contornar todo o conjunto dos estados separáveis.

3.2.4.1 Construção e Optimização da Testemunha de Emaranhamento

Existem diversas metodologias para a criação de testemunhas de emaranhamento. Iremos apresentar alguns exemplos contidos na literatura, porém existem outros métodos (GÜHNE; TÓTH, 2009).

1. Considere uma matriz densidade ρ. Se existir um autovalor negativo $\lambda_- < 0$ de ρ^{T_A} associado a um autovetor $|\psi\rangle$, podemos criar uma testemunha de emaranhamento como

$$W = |\psi\rangle\langle\psi|^{T_A}. \tag{57}$$

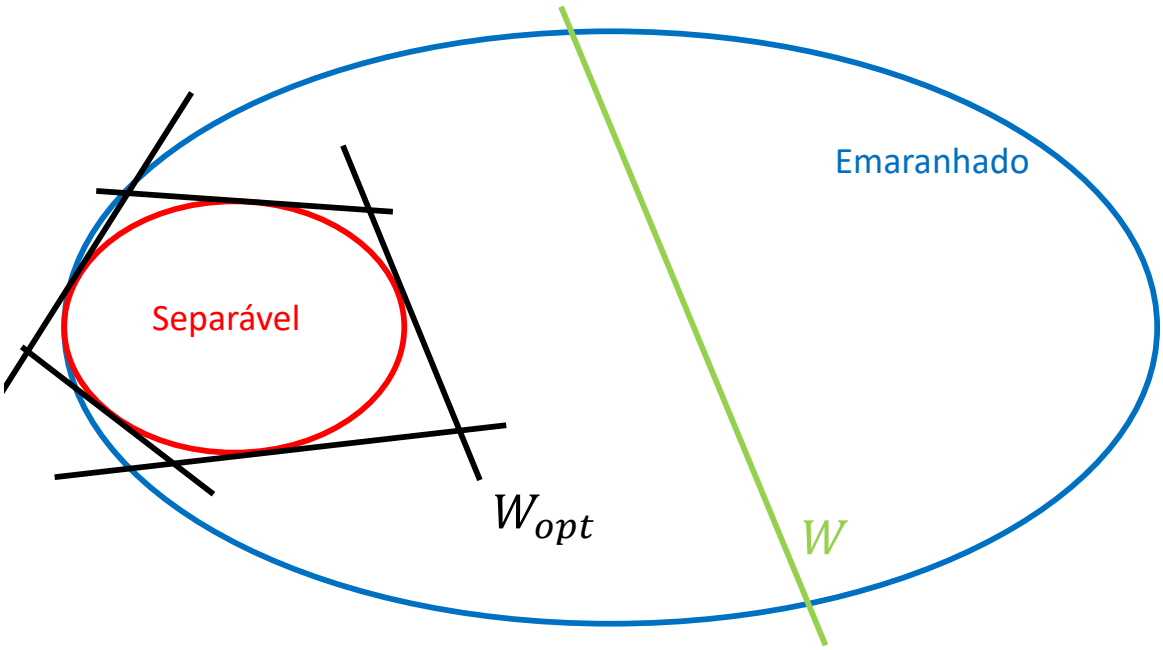


Figura 13 – Representação esquemática do espaço de estados contendo uma testemunha de emaranhamento, W, e testemunhas de emaranhamento optimizadas, W_{opt}, circulando o conjunto de estados separáveis.

Desta maneira, teremos para estados emaranhados

$$\begin{aligned} \mathrm{Tr}(W\rho) &= \mathrm{Tr}\left(|\psi\rangle\langle\psi|^{T_A}\,\rho\right) \\ &= \mathrm{Tr}\left(|\psi\rangle\langle\psi|\,\rho^{T_A}\right) \\ &= \lambda_- < 0 \end{aligned} \tag{58}$$

e para estados separáveis

$$\mathrm{Tr}(W\rho) = \mathrm{Tr}\left(|\psi\rangle\langle\psi|\,\rho^{T_A}\right) \geq 0 \tag{59}$$

2. Podemos construir uma testemunha de emaranhamento utilizando o projetor de um estado puro $|\psi\rangle$

$$W = \alpha\mathbb{I} - |\psi\rangle\langle\psi|\,. \tag{60}$$

Esta testemunha nos diz que se

$$\mathrm{Tr}(\rho\,|\psi\rangle\langle\psi|) = \langle\psi|\,\rho\,|\psi\rangle\,. \tag{61}$$

for o valor esperado do estado $|\psi\rangle$ para um estado misto ρ, então se o valor esperado extrapolar um valor crítico α, também teremos um valor esperado negativo para a testemunha e ρ será necessariamente um estado emaranhado. O valor α representa o menor valor para o qual W ainda permanece positivo e pode ser calculado pela expressão (BOURENNANE *et al.*, 2004)

$$\alpha = \max_{\rho_A \otimes \rho_B} \mathrm{Tr}(\rho\,|\psi\rangle\langle\psi|) \tag{62}$$

3. Seja ρ um estado emaranhado e σ o estado separável mais próximo de ρ. Então podemos definir uma testemunha de emaranhamento como

$$W = \frac{1}{\mathcal{N}}\{\sigma - \rho + \mathrm{Tr}[\sigma(\rho - \sigma)]\mathbb{I}\} \tag{63}$$

onde $\mathcal{N} = \sqrt{\mathrm{Tr}\big((\rho-\sigma)^\dagger(\rho-\sigma)\big)}$ é a normalização.

Após a construção de uma testemunha de emaranhamento podemos realizar sua optimização (LEWENSTEIN *et al.*, 2000) que encontre um maior número de estados emaranhados. Considerando duas testemunhas W_1 e W_2, de maneira que W_2 é capaz de detectar os mesmo estados de W_1 e mais um estado adicional. Definimos então

$$W_2 = W_1 - \alpha P \tag{64}$$

onde P é um operador positivo. Se as propriedades

$$\langle\psi|\,W_1\,|\psi\rangle = \langle a_i b_i|\,W_1\,|a_i b_i\rangle = 0 \tag{65}$$

e

$$\langle\psi|\,P\,|\psi\rangle = \langle a_i b_i|\,P\,|a_i b_i\rangle = 0 \tag{66}$$

forem cumpridas, temos que W_2 é uma testemunha mais refinada que W_1 se

$$\alpha \leq \alpha_0 := \inf_a \min_{autovalores} \left[\frac{1}{\sqrt{P_a}} W_a \frac{1}{\sqrt{P_a}}\right] \tag{67}$$

em que inf = infimo, $X_a = \langle a|\,X\,|a\rangle$, $|a\rangle \in \mathcal{H}_A$ e $|b\rangle \in \mathcal{H}_B$.

Podemos exemplificar este processo de optimização através de um estado puro com um certo grau de mistura, estado de Werner. Este estado pode ser representado por

$$\rho = p\,|\psi\rangle\langle\psi| + (1-p)\frac{\mathbb{I}}{4} \tag{68}$$

onde $p \in [0,1]$

Se o estado $|\psi\rangle$ for da forma $|\psi\rangle = a\,|00\rangle + b\,|11\rangle$ e considerando a identidade como o estado maximamente misto, teremos

$$\mathbb{I} = |00\rangle\langle 00| + |11\rangle\langle 11| + |01\rangle\langle 01| + |10\rangle\langle 10|\,, \tag{69}$$

$$|\psi\rangle\langle\psi| = a^2\,|00\rangle\langle 00| + ab\,|00\rangle\langle 11| + ab\,|11\rangle\langle 00| + b^2\,|11\rangle\langle 11| \tag{70}$$

e por consequência, nosso estado de Werner assumirá a forma

$$\begin{aligned}\rho = &\left(\frac{(1-p)}{4} + a^2\right)|00\rangle\langle 00| + \left(\frac{(1-p)}{4} + b^2\right)|11\rangle\langle 11| + \\ &+ pab(|00\rangle\langle 11| + |11\rangle\langle 00|) + \frac{(1-p)}{4}(|01\rangle\langle 01| + |10\rangle\langle 10|)\end{aligned} \tag{71}$$

Realizando a transposta parcial no sub-sistema B, teremos

$$\begin{aligned}\rho^{T_B} &= \left(\frac{(1-p)}{4} + a^2\right)|00\rangle\langle 00| + \left(\frac{(1-p)}{4} + b^2\right)|11\rangle\langle 11| \\ &\quad + pab(|01\rangle\langle 10| + |10\rangle\langle 01|) + \frac{(1-p)}{4}(|01\rangle\langle 01| + |10\rangle\langle 10|) \\ &= \begin{pmatrix} \left(\frac{(1-p)}{4} + a^2\right) & 0 & 0 & 0 \\ 0 & \frac{(1-p)}{4} & pab & 0 \\ 0 & pab & \frac{(1-p)}{4} & 0 \\ 0 & 0 & 0 & \left(\frac{(1-p)}{4} + b^2\right) \end{pmatrix}.\end{aligned} \tag{72}$$

Realizando a diagonalização de ρ^{T_B} encontramos os seguintes autovalores e seus autovetores associados:

$$\lambda_1 = \left(\frac{(1-p)}{4} + a^2\right) \Rightarrow |00\rangle\,, \tag{73}$$

$$\lambda_2 = \left(\frac{(1-p)}{4} + b^2\right) \Rightarrow |11\rangle\,, \tag{74}$$

$$\lambda_3 = \frac{(1-p)}{4} + pab \Rightarrow |\psi_+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \tag{75}$$

$$\lambda_4 = \frac{(1-p)}{4} - pab \Rightarrow |\psi_-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \tag{76}$$

Tomando o menor autovalor como base, temos que o estado de Werner será emaranhado se λ_4 for negativo, ou seja,

$$\begin{aligned} \lambda_4 &< 0, \\ \frac{1}{4} - p\left(\frac{1}{4} + ab\right) &< 0, \\ p &> \frac{1}{(1+4ab)}. \end{aligned} \tag{77}$$

Construímos então nossa testemunha como sendo a transposição parcial do projetor para este auto-estado a qual pode ser escrita como

$$\begin{aligned} W =& |\psi_-\rangle\langle\psi_-|^{T_B} \\ =& \frac{1}{2}(|01\rangle\langle 01| - |00\rangle\langle 11| - |11\rangle\langle 00| + |10\rangle\langle 10|) \\ =& \frac{1}{2}(|01\rangle\langle 01| - |00\rangle\langle 11| - |11\rangle\langle 00| + |10\rangle\langle 10| \\ &+ |00\rangle\langle 00| - |00\rangle\langle 00| + |11\rangle\langle 11| - |11\rangle\langle 11|) \\ =& \frac{1}{2}\left(\mathbb{I} - |\phi_+\rangle\langle\phi_+|\right) \\ =& \frac{1}{2}\begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} \end{aligned} \tag{78}$$

onde $|\phi_+\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$

Existe um resultado mais geral que nos diz que qualquer mapa decomponível corresponde a uma testemunha, ou seja,

$$W = P + Q^{T_B}, \tag{79}$$

onde P e Q são operadores positivos semidefinidos e a testemunha W é uma testemunha de emaranhamento decomponível. Podemos derivar uma expressão para tal testemunha (VIDAL, Guifré; TARRACH, 1999b; STEINER, 2003b). O método se baseia na ideia de utilizar um estado de borda, δ, que é um estado emaranhado preso de tal forma que para todo $\epsilon \geq 0$ e todo separável $|ab\rangle$,

$$\delta - \epsilon\,|ab\rangle\langle ab| \tag{80}$$

não é positivo ou não possui o traço parcial positivo. Os estados de borda podem ser pensados como estando na linha de divisão dos estados emaranhados presos

e emaranhados, e por isso eles são considerados o estado mais emaranhado preso possível. Se este estado existir, podemos construir um estado emaranhado preso como a soma convexa do estado de borda e um estado separável(LEWENSTEIN *et al.*, 2000, 2001). Uma representação esquemática deste espaço pode ser vista na Figura 14.

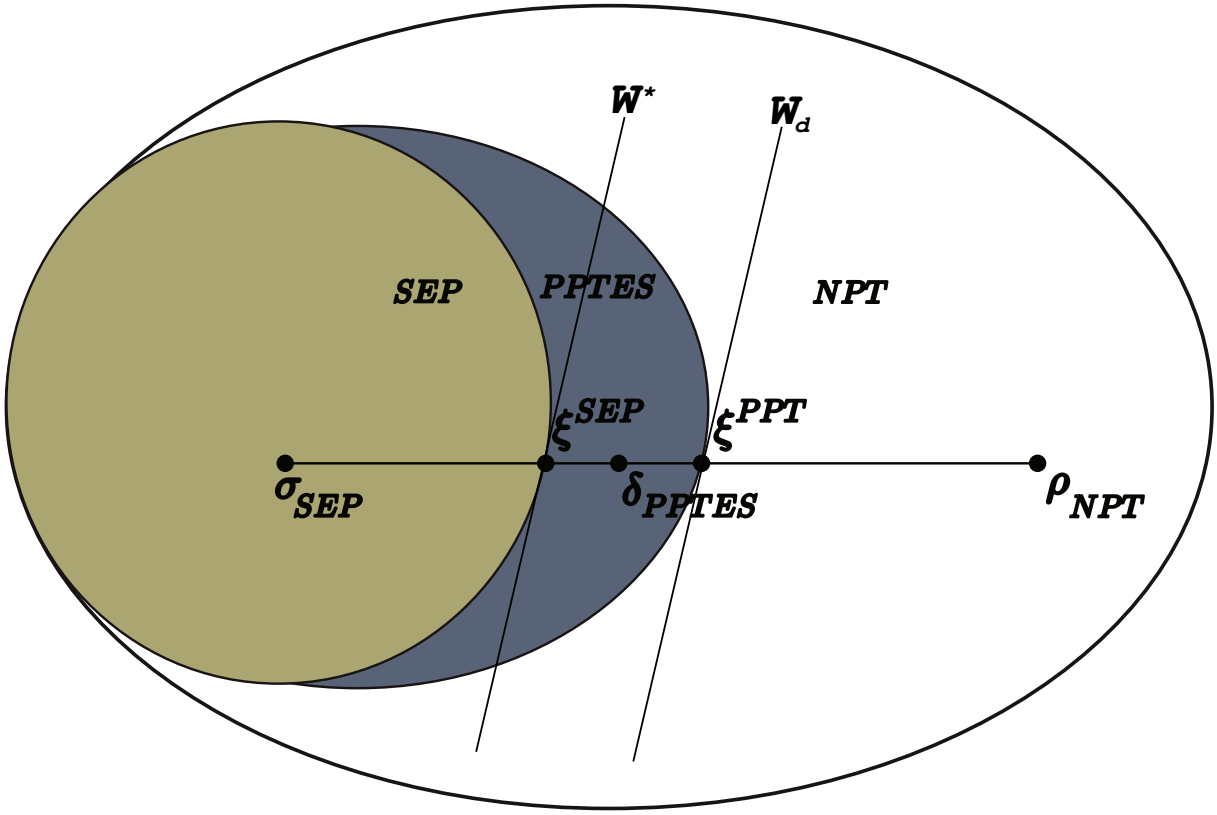


Figura 14 – Representação esquemática do espaço de estados contendo a construção de um estado emaranhado preso, δ_{PPTES} como sendo a soma convexa de um estado de borda, ξ^{PPT}, e um estado separável, σ_{sep}. O estado ξ^{SEP} representa um estado de borda do conjunto de estados separáveis e ρ^{NPT} representa um estado emaranhado qualquer.

Teorema 3.2.5. *(CLARISSE, 2006) Todo estado emaranhado preso, ρ, pode ser decomposto como*

$$\rho = (1-p)\rho_{sep} + p\delta, \tag{81}$$

onde ρ_{sep} é um estado separável e δ é um estado de borda.

Desta maneira podemos construir uma testemunha que detecte o estado de borda.

Teorema 3.2.6. *(CLARISSE, 2006)*

1. *Seja δ um estado de borda com P e Q o projetor no kernel de δ e δ^{T_b}, respectivamente. Seja*

$$W_\delta = P + Q^{T_B} \tag{82}$$

e

$$\epsilon = \inf_{|ab\rangle} \langle ab| W_\delta |ab\rangle , \tag{83}$$

então, $W_1 = W_\delta - \epsilon\mathbb{I}$ *é uma testemunha de emaranhamento não decomponível que detecta* δ*.*

2. *Qualquer testemunha não decomponível* W *pode ser escrita como*

$$W = P + Q^{T_B} - \epsilon\mathbb{I} \tag{84}$$

com

$$0 < \epsilon < \inf_{|ab\rangle} \langle ab| P + Q^{T_B} |ab\rangle , \tag{85}$$

sendo a condição de P *e* Q *serem operadores positivos semidefinidos e satisfazerem* $\mathrm{Tr}\, P\delta = \mathrm{Tr}\, Q^{T_B}\delta = 0$ *para um estado de borda qualquer.*

3.3 CLASSIFICADOR DE EMARANHAMENTO BASEADO EM ML

Nesta seção iremos descrever a criação do classificador de emaranhamento de dois qutrits. Passando pela metodologia de obtenção de dados e pela criação da ANN que irá classificar os mesmos.

3.3.1 Construção do conjunto de dados

A construção do conjunto de dados é um dos principais componentes para que os modelos de ML consigam um bom desempenho na classificação. Neste trabalho foram desenvolvidos dois conjuntos de dados. O primeiro corresponde somente a estados de Horodecki, descritos na seção anterior e o segundo corresponde a estados espalhados por todo o espaço de Hilbert de dois qutrits.

Para a construção do conjunto de dados vamos considerar um estado quântico 9×9, $\rho \in \mathcal{D} \subset \mathcal{L}(\mathcal{C}^3 \otimes \mathcal{C}^3)$, onde $\mathcal{D}$ é o espaço de estados e $\mathcal{L}$ é o conjunto de operadores lineares no espaço de Hilbert $\mathcal{H} = \mathcal{C}^3 \otimes \mathcal{C}^3$. Desta forma podemos escrever ρ como,

$$\rho = \sum_{i=1}^{d^2-1} \theta_i E_i, \tag{86}$$

onde $d = 9$, E_i forma uma base no espaço de Hilbert-Schimidt e θ_i é a solução do seguinte sistema linear (NIELSEN; CHUANG, 2000),

$$G^{-1}\vec{c} = \vec{\theta}, \quad G_{ij} = \text{tr}\left(E_i E_j\right), \quad c_i = \text{tr}\, E_i \rho. \tag{87}$$

Podemos escolher qualquer base que abranja o espaço de Hilbert-Schmidt e o Gramian G é positivo definido. Assim, caso possuímos todos $i = 1, \ldots, d^2 - 1$ valores esperados $\{c_i\}$ das medidas dos observáveis linearmente independentes $\{E_i\}$, nós possuímos dados suficientes para realizar uma tomografia de estados quânticos completa e determinar ρ unicamente usando as Equações 86 e 87. Assumimos que possuímos todos os $d^2 - 1$ coeficientes, queremos evitar de medir projetores emaranhados, desta maneira escolhemos os geradores $SIC\_POVM(3) \otimes SIC\_POVM(3)$ para os estados de Horodecki e os geradores $SU(3) \otimes SU(3)$ para o caso mais geral. Assumimos também que as medidas tomográficas das Equações 86 e 87 são perfeitas e levam a um estado válido, ou seja, possuem operadores positivos semi-definidos de traço um. As imperfeições experimentais iriam adicionar uma nova camada de erro em nossa análise e escolhemos priorizar a performance das técnicas de ML aplicados ao problema com a menor fonte de erros.

Para os estados de Horodecki foram criados dois mil exemplos de cada classe de emaranhamento através da implementação da receita dos estados em um código em MATLAB (MATLAB, 2010). O conjunto de dados corresponde aos elementos da matriz densidade, de cada estado, por medidas tomográficas representadas por SIC-POVMs (RENES *et al.*, 2004) vinculados a um rótulo representando a classe de emaranhamento. Totalizamos então seis mil exemplos de estados com dezesseis entradas.

Para o segundo conjunto de dados realizamos o sorteio aleatório de matrizes densidade através da métrica de Hilbert-Schmidt (BRAUNSTEIN, Samuel L, 1996; ZYCZKOWSKI; SOMMERS, 2001) de estados de um sistema de dois qutrits ($dim = 9$). O ideal seria possuir uma distribuição uniformemente aleatória de estados que fossem separáveis, emaranhados presos e emaranhados ao longo de todo o espaço de Hilbert. Porém, como a probabilidade de sortearmos um estado que seja emaranhado preso é muito pequena, nós adicionamos ao conjunto de dados estados emaranhados presos artificialmente construídos utilizando estados de borda (LEWENSTEIN *et al.*, 2000, 2001). Para a rotulação dos estados entre emaranhados (transposta parcial negativa (NPT)), emaranhados presos (PPTES) e separáveis (separáveis (SEP)) utilizamos o formalismo de testemunha de emaranhamento atuando em $\mathcal{L}(\mathcal{C}^3 \otimes \mathcal{C}^3)$ de maneira que:

$$\begin{aligned} &\mathrm{tr}\,(W\rho) \geq 0,\ \forall \rho \in PPTES \cup SEP, \\ &\mathrm{tr}\,(W\rho) < 0,\ \forall \rho \in NPT. \end{aligned} \tag{88}$$

Com essa testemunha, estamos interessados em uma subclasse correspondente a testemunha de emaranhamento ótima, W^*, ou seja

$$\mathrm{tr}\,(W^*\rho) \leq \min_{W \in \mathcal{M}} \{\mathrm{tr}\,(W\rho)\}, \tag{89}$$

onde $\mathcal{M}$ é a intersecção do conjunto de testemunhas de emaranhamento, $\mathcal{W}$, com algum outro conjunto $\mathcal{C}$ tal que $\mathcal{M}$ é compacta e é limitada inferiormente.

Podemos associar uma testemunha de emaranhamento a diferentes medidas de emaranhamento escolhendo de maneira adequada $\mathcal{C}$ (BRANDÃO, 2005). Em nosso caso escolhemos $\mathcal{M} = \{W \in \mathcal{W} | W \preceq \mathbb{I}$, onde $\mathbb{I}$ é o operador identidade, associando W com a GRE de forma que:

$$GR(\rho) = \min_{\sigma \in \mathcal{D}} \left(\min_{s \geq 0 \in \mathbb{R}} s : \xi^{SEP} \equiv \frac{\rho + s\sigma}{1+s} \in SEP \right) \tag{90}$$

onde ξ^{SEP} pertence à borda entre o conjunto de estados separáveis e emaranhados, como representado na Figura 14. Associamos as Equações 89 e 90 para obter

$$GR(\rho) = \max \left\{ 0, -\min_{W \equiv \mathbb{I}} \{\mathrm{tr}\,(W\rho)\} \right\}. \tag{91}$$

Realizamos então uma minimização na Equação 91 através de uma ϵ- Testemunha de emaranhamento ótima (BRANDÃO; VIANNA, 2004) utilizando o MATLAB (MATLAB, 2019), YALMIP (LÖFBERG, 2004) e SDPT3 (TOH *et al.*, 1999). Configuramos o algoritmo de maneira que a probabilidade de encontrar um estado separável com $GR(\sigma) > \epsilon$ é muito pequena. Mais especificamente, nós escolhemos

$$\mathbb{P}\{\mathrm{tr}\,(W\sigma) < -\gamma | \sigma \in SEP\} \leq \epsilon, \tag{92}$$

onde $\epsilon = \gamma = 10^{-5}$. Então, nós classificamos ρ como sendo um estado emaranhado se $GR(\rho) > \epsilon$.

Finalmente, obtemos os estados de borda ξ^{SEP} (como na Figura 14) resolvendo o problema,

$$\begin{aligned} \text{determine} \quad & \sigma \\ \text{tal que} \quad & \operatorname{tr}\sigma = 1 \\ & \sigma \succeq 0 \\ & \xi = \frac{\rho + GR(\rho)\sigma}{1 + GR(\rho)} \\ & \xi^{T_A} \succeq 0 \\ & \operatorname{tr} W\xi = 0, \end{aligned} \tag{93}$$

onde ξ^{T_A} denota a transposta parcial no primeiro subsistema de $\mathcal{L}(\mathbb{C}^3 \otimes \mathbb{C}^3)$.

Como a probabilidade de sortearmos um estado PPTES é muito pequena utilizamos um algoritmo que os cria artificialmente através de estados de borda. Para isso computamos uma testemunha ótima decomponível com o MATLAB, YALMIP, SDPT3, mas agora resolvendo o *problema semidefinido*.

$$\begin{aligned} \min_{W_d} \quad & \operatorname{tr}\rho W_d \\ \text{tal que} \quad & W_d \preceq \mathbb{I} \\ & W_d^{T_A} \succeq 0, \end{aligned} \tag{94}$$

onde W^{T_A} denota a transposta parcial no primeiro subsistema de $\mathcal{L}(\mathbb{C}^3 \otimes \mathbb{C}^3)$. Chamaremos de $GR_d(\rho)$ o emaranhamento quantificado por W_d. Esta testemunha irá encontrar a fronteira do espaço entre os estados PPTES e NPT. Nós a utilizamos para encontrar os estados de borda ξ^{PPTES} que vive nesta fronteira. Os casos em que $\xi^{SEP} \neq \xi^{PPTES}$ são especialmente úteis para nós, já que podemos construir um estado PPTES com uma combinação convexa entre estes estados (LEWENSTEIN *et al.*, 2001). Nós dizemos que eles são diferentes para uma tolerância 10^{-3} na norma 1.

Podemos então resumir nosso algoritmo para a construção do conjunto de dados como:

1. Sorteie uma matriz densidade ρ de acordo com a medida de Hilbert-Schimdt;
2. Compute $GR(\rho)$ utilizando ϵ–OEW, testemunha de emaranhamento óptima (*optimum entanglement witness*) e obtenha o estado de borda ξ^{SEP};
3. Cheque se ρ é um estado PPT;
4. se ρ é PPT e $GR(\rho) > \epsilon$, nós salvamos ρ em nosso conjunto de dados com o rótulo PPTES;

5. se ρ é PPT e $GR(\rho) < \epsilon$, nós salvamos ρ em nosso conjunto de dados com o rótulo SEP;
6. se ρ é NPT e, consequentemente, $GR(\rho) > \epsilon$, nós salvamos ρ em nosso conjunto de dados como NPT;
7. se ρ é NPT, nós também quantificamos $GR_d(\rho)$ com d–OEW e obtemos seu estado de borda ξ^{PPTES};
8. Checamos se ξ^{SEP} e ξ^{PPTES} são diferentes na norma 1, $\frac{1}{2} \parallel \xi^{SEP} - \xi^{PPTES} \parallel_1 > 10^{-3}$;
9. se eles forem diferentes, nós teremos $\delta = \frac{1}{2}\xi^{SEP} + \frac{1}{2}\xi^{PPTES}$;
10. computamos $GR(\delta)$ utilizando ϵ–OEW;
11. se $GR(\delta) > \epsilon$, nós salvamos δ em nosso conjunto de dados com o rótulo PPTES;

Construímos 3254 exemplos de cada classe em nosso conjunto de dados, com metade dos estados PPTES gerados artificialmente, totalizando 9762 exemplos. O algoritmo levou aproximadamente um mês utilizando um *cluster* de 18 cores e 90GB de RAM para sortear dez mil matrizes densidade e calcular suas ϵ–OEW com a precisão numérica desejada. Ao término, representamos a matriz densidade por um conjunto de medidas tomográficas sobre a base $SU(3) \otimes SU(3)$, totalizando oitenta entradas. O aumento do número de entradas deste conjunto demonstra a maior complexidade do mesmo em relação ao conjunto contendo os estados de Horodecki.

Como estamos incluindo estados PPTES criados artificialmente em adição aos sorteados aleatoriamente, nós precisamos nos certificar se possuímos um subconjunto justo. Para essa tarefa, nós utilizamos a fidelidade média de estados. Essa medida nos mostra o quão próximo os estados de um grupo estão uns dos outros. Na Tabela 2 nós possuímos a fidelidade média,

$$F(\rho, \sigma) = \left[\mathrm{tr} \left(\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}} \right) \right] \tag{95}$$

onde ρ e $\sigma \in \mathcal{L}(\mathcal{C}^3 \otimes \mathcal{C}^3)$ são estados não equivalentes dentro do nosso conjunto de dados. Notamos uma média da fidelidade parecida para os estados PPTES e SEP e, como esperado, os estados NPT estão mais distantes uns dos outros já que eles possuem um maior volume dentro do espaço de Hilbert (ŻYCZKOWSKI *et al.*, 1998). Esta análise é fundamental para controlar o viés e o overfitting em nossas amostras (MAY *et al.*, 2010).

Tabela 2 – Fidelidade média, $\langle F \rangle$, ao longo de cada classe de emaranhamento presente no conjunto de dados. A classe Todos se refere a todos os 9762 estados contidos em todo o conjunto de dados, enquanto SEP, PPTES e NPT se referem aos 3254 estados em cada uma destas classes.

Class	$\langle F \rangle$
Todos	0,7879
Emaranhado	0,7712
Emaranhado preso	0,7965
Separável	0.7955

3.3.2 Resultados e Discussão

3.3.2.1 Estados de Horodecki.

Para a construção e treinamento das ANN utilizadas para a classificação dos estados de Horodecki utilizamos o pacote de redes neurais do MATLAB. A arquitetura da rede utilizada neste primeiro resultado foi a arquitetura MLP com duas camadas escondidas contendo, dez neurônios cada. O algoritmo de treinamento utilizado foi o gradiente descendente escalonado.

Como descrito no capítulo anterior, a ANN tenta descobrir uma superfície que seja capaz de separar as classes apresentadas à mesma. Esta superfície será tão mais complexa quanto o seu problema. Devido a este fato, esperamos que os estados uniformemente distribuídos pelo espaço de Hilbert possuam uma superfície de separação dos estados com um formato mais complexo que os estados de Horodecki.

Para os estados de Horodecki conseguimos uma classificação de aproximadamente 100% em todas as classes, como apresentado na matriz confusão da Figura 15. Esta matriz compara a classe presente no conjunto de dados (target class) com a classe que a rede treinada nos fornece (output class) utilizando os mesmos valores de entrada. O que demonstra que a superfície de separação destes estados possui uma forma simples para estes estados.

3.3.2.2 Estados uniformemente distribuídos.

3.3.2.2.1 Engenharia de features.

Algoritmos como autoML podem fornecer valiosos *insights* ao analisar resultados parciais. Por exemplo, podemos utilizar o TPOT (LE *et al.*, 2019) para analisar a importância de cada *feature* presente em nosso conjunto de dados. O TPOT, como descrito no capítulo anterior, utiliza um algoritmo de programação genética para explorar diferentes *pipelines*[1] para cada conjunto de dados. Ele automaticamente roda testes padrões, como por exemplo, análise de tipos de dados, z-score, checa a relevância de

[1] pipelines é a divisão do fluxo de trabalho de ML em partes independentes, reutilizáveis e modulares.

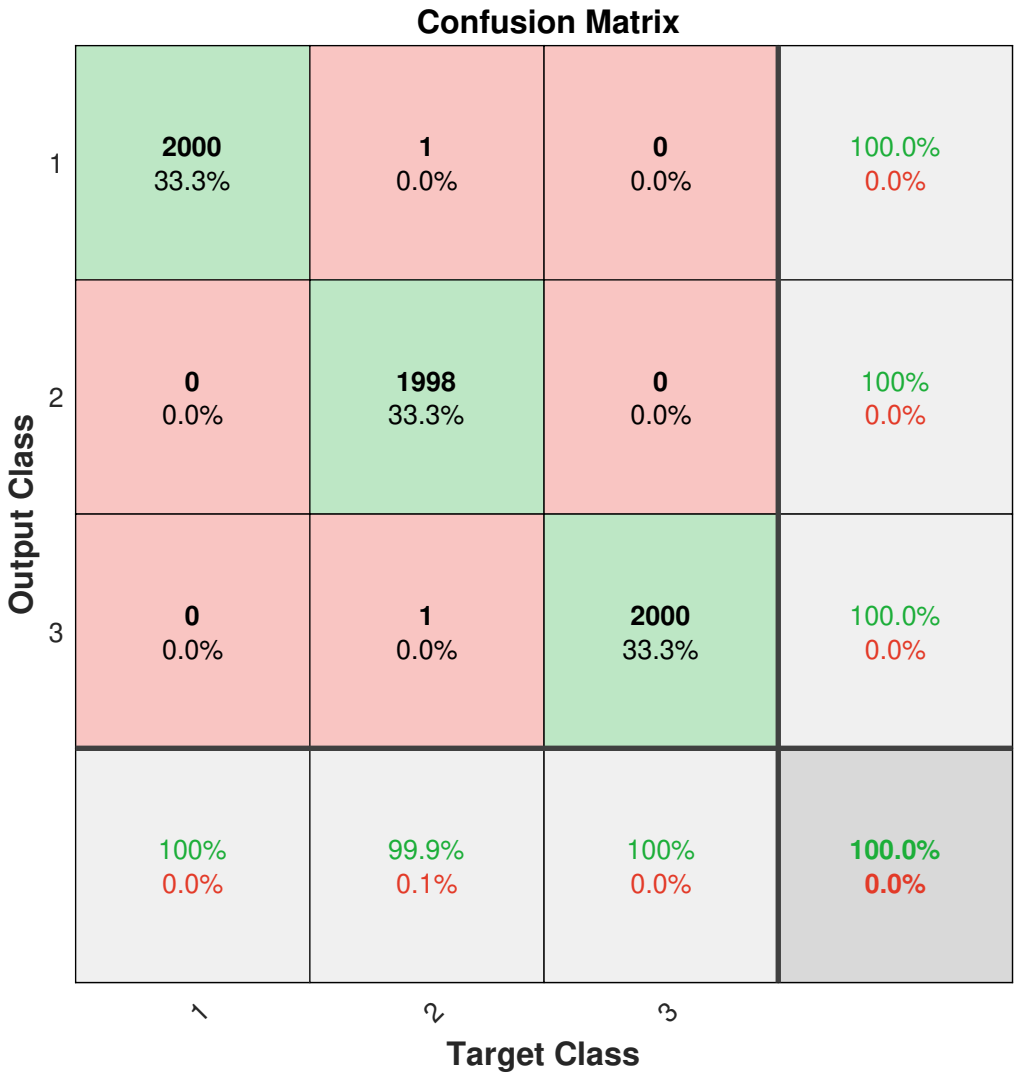


Figura 15 – Matriz confusão para os estados Horodecki. A classificação é de 100% para todas as classes. Onde a classe 1 representa os estados NPT, 2 representa os estados PPTES e 3 representa os estados SEP.

potencias maiores dos *inputs*, redução da dimensionalidade via PCA (*Principal Component Analysis*), optimização de hiper-parâmetros, e assim por diante. Esse tipo de algoritmo é um bom ponto de partida para qualquer projeto de ML, já que ele possibilita investigar se uma *feature* é mais importante que outra, que é um ponto crucial para diminuir o número de *features* durante o processo de aprendizagem, reduzir o esforço computacional e aumentar a robustez contra *over-fitting* (MEHTA *et al.*, 2019)

Nós testamos a importância dos *features* de nosso dataset utilizando o TPOT com dez gerações do algoritmo genético e uma população de trinta modelos de ML para a construção do ensemble. A optimização levou em torno de seis horas para concluir utilizando um PC e obtemos a confirmação que todas as oitenta *features*, medidas tomográficas, contribuem igualitariamente para resolver o problema. Desta maneira, não podemos reduzir a dimensionalidade dos dados sem perder informações consideráveis. Considerando que nosso *feature* c_i (Equação 87) representa o produto interno de ρ com os observáveis linearmente independentes (geradores do espaço $SU(3) \otimes SU(3)$), este resultado é esperado. Assim, o cenário onde possuímos apenas informações parciais sobre os estados não favorece o algoritmo de ML.

Para o problema de classificação, em adição às *features* do vetor $\vec{c} \in \mathbb{R}^{80}$, nós também incluímos o seus valores exponenciais, $\exp\{(c_i)\}$ para $\forall i = 1, ..., 80$ e termos de segunda ordem $c_i c_j$ para $\forall\ i = 1, ..., 80$ e $j = 1, ..., 80$ podemos desconsiderar os termos repetidos como $c_1 c_2 = c_2 c_1$, totalizando 3400 *features*. Todos os *features* são normalizados com *z*-score, isto é, centrado para possuir valor médio zero e escalado para possuir desvio padrão igual a um.

3.3.2.2.2 Classificação Binária (SEP vs. PPTES).

Depois de decidir os *features*, a primeira tarefa é o treinamento binário para diferenciar o estado entre separável ou emaranhado preso. Essa tarefa é especialmente importante quando o critério de Peres-Horodecki não é capaz de identificar o emaranhamento de um estado e nós precisamos decidir quando ρ é emaranhado. Para obter a melhor performance de cada classificador, nós dividimos o conjunto de dados em duas partes:

- $\sim$ 80% para treinamento,
- $\sim$ 20% para teste.

A idéia é testar um modelo treinado com informação nova, para assegurar que o modelo é capaz de generalizar a predição e é robusto contra overfitting. Assim, dizemos que possuímos um bom modelo quando alcançamos uma alta performance na fase de teste.

Tecnica de ML	Teste (%)	PPTES (%)	SEP (%)
TPOT	**75.3**	**76.6**	**74.1**
auto-sklearn	73.4	71.9	75.0
AutoKeras	41.6	40.0	36.0
Auto-PyTorch	62.2	63.7	60.8
AdaBoost	66.9	65.7	68.1
ANN	58.4	57.0	59.9
SVM	70.5	67.1	73.7
KNN	48.4	0.2	99.8
DT	53.0	49.6	56.5
RF	70.3	73.2	67.4

Tabela 3 – Performance do conjunto de teste para diferentes tecnicas de ML utilizando o mesmo conjunto de dados para a classificação binária SEP *vs.* PPTES. Nós mostramos a melhor performance em negrito. As técnicas de autoML são TPOT, auto-sklearn, AutoKeras e Auto-PyTorch. Os métodos convencionais são o método de ensemble AdaBoost, rede neural artificial (ANN), *support vector machine* (SVM), K primeiros vizinhos (KNN), árvore de decisão (DT), e *random florest* (RF). As porcentagens em Teste, PPTES, e SEP referente a taxa de classificação corretas (Equação 96).

Nós treinamos diferentes modelos utilizando diferentes técnicas de ML em uma janela de tempo de seis horas em um computador padrão e computamos a taxa de classificações corretas:

$$\frac{\#\text{ classificações corretas}}{\#\text{ estados}} \times 100\% \tag{96}$$

para todos os elementos no conjunto de teste e para cada classe (SEP/PPTES) separadamente. Podemos ver o resultado desta taxa de acertos na Tabela 3 com o melhor acerto destacado em negrito. Podemos notar que todas as técnicas de autoML (TPOT, auto-sklearn, AutoKeras, Auto-PyTorch) tiveram uma performance melhor que seu correspondente (ANN, SVM, KNN, DT, RF e Adaboost). No passo de teste, o TPOT classificou os estados SEP com uma acurácia de $74,1\%$ e os estados PPTES com $76,6\%$, com uma performance geral de $75,3\%$. A matriz de confusão (CM) (GOODFELLOW *et al.*, 2016) para este teste está representada na Figura 16, mostrando as taxas de acerto (verde) e de erros (vermelho).

3.3.2.2.3 Classificação Múltiplas (SEP/PPTES/NPT).

Embora nós possamos verificar se um estado quântico é PPT ou NPT eficientemente, após a classificação bem sucedida da seção anterior, nós queremos investigar uma classe a mais em nosso autoML *framework* para uma descrição do emaranhamento em um sistema de dois qutrits. A tarefa a ser realizada é a classificação em três classes distintas SEP/PPTES/NPT.

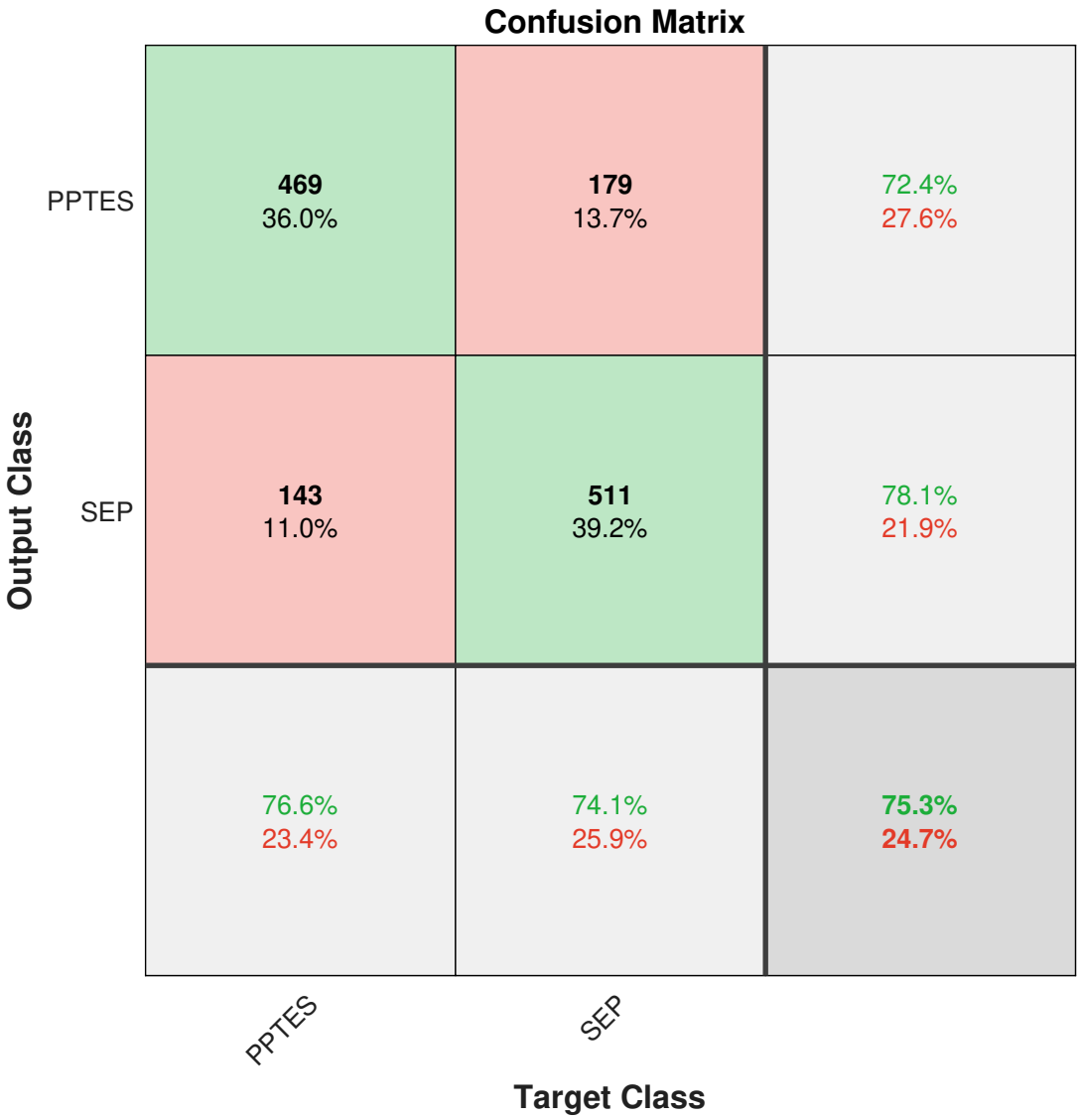


Figura 16 – Matriz Confusão do conjunto de teste do classificador TPOT. As linhas correspondem a classificação fornecida pelo classificador TPOT (SEP/PPTES) e as colunas correspondem ao rótulo verdadeiro contido no conjunto de dados (também SEP/PPTES), contendo as entradas classificadas corretamente na diagonal e as classificadas erroneamente fora da diagonal com o número de estados e a porcentagem em relação ao número total de observações representado em ambos. A última linha e coluna mostra a porcentagem de todos os estados que são classificados corretamente (verde) e incorretamente (vermelho) pertencente a cada classe. O TPOT alcançou uma performance total de 75, 3%.

Para essa tarefa dividimos novamente nosso conjunto de dados em:

- $\sim$ 80% para treinamento,
- $\sim$ 20% para teste,

e rodamos os mesmos algoritmos da seção anterior com uma janela de tempo de seis horas em um computador padrão. Nós apresentamos os resultados na Tabela 4 com o melhor resultado em negrito. Notamos que o TPOT continuou a apresentar uma performance melhor que as demais técnicas, classificando com uma acurácia de $71,4\%$ para os estados SEP, $62,4\%$ para os PPTES e $88,6\%$ para os NPT, com uma performance total de $73,8\%$. A matriz confusão para o conjunto de teste é mostrada na Figura 17.

Técnicas de ML	Teste (%)	NPT (%)	PPTES (%)	SEP (%)
TPOT	**73.8**	**88.6**	**62.4**	**71.4**
Auto-sklearn	71.9	86.1	53.2	77.2
AutoKeras	52.5	53.0	44.5	54.9
Auto-PyTorch	42.7	68.7	14.0	45.1
AdaBoost	64.2	95.6	48.1	73.5
ANN	56.0	70.3	42.4	55.6
SVM	72.9	86.5	53.7	69.6
KNN	34.1	0.4	0.6	99.6
DT	43.1	48.0	35.1	45.9
RF	69.2	85.0	45.6	76.4

Tabela 4 – Diferente técnicas de ML aplicadas ao cojunto de dados e sua acurácia correspondente para o problema de classificação multiclasse. A melhor acurácia está destacada em negrito. As técnicas de autoML são TPOT, Auto-sklearn, AutoKeras, e Auto-Pytorch. Os métodos de ensemble são Adaboost e Random Forest(RF). Os métodos convencionais são redes neurais artificiais (ANN), *suport vector machine* (SVM), K primeiros vizinhos (KNN), árvore de decisão (DF). O percentual obtido com o conjunto de teste, PPTES, SEP e NPT se referem a taxa de acerto descrita na Equação 96.

É impressionante como o TPOT e o auto-sklearn conseguem lidar com a maldição da dimensionalidade, alcançando uma alta acurácia para este cenário desafiador. O conjunto de classificação agora possui diversas bordas: entre SEP e PPTES, SEP e NPT e PPTES e NPT. Uma redução da performance geral é esperada, principalmente com um conjunto de dados de alta dimensionalidade (possuindo 3400 *features* e 9762 exemplos). Nós observamos uma baixa performance apenas nos métodos de *DeepLearning* (ANN, AutoKeras e Auto-PyTorch), conhecido por necessitar de um grande número de exemplos no conjunto de dados para alcançar uma boa convergência (GOODFELLOW *et al.*, 2016). A despeito de, com uma performance total de $73,8\%$ para

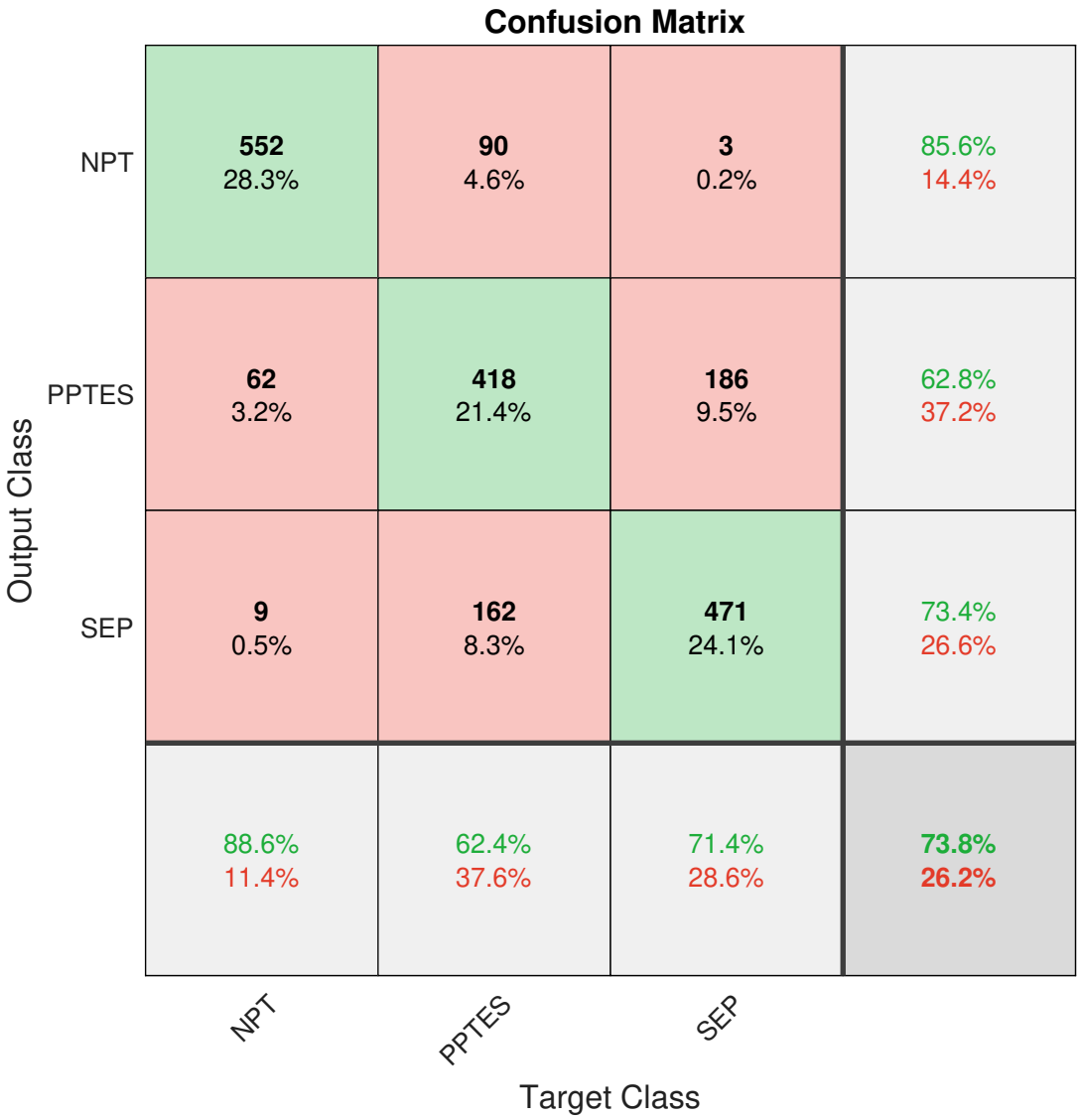


Figura 17 – Matriz Confusão do conjunto de teste do classificador TPOT. As linhas correspondem a classificação fornecida pelo classificador TPOT (NPT/PPTES/SEP) e as colunas correspondem ao rótulo verdadeiro contido no conjunto de dados (também NPT/PPTES/SEP), contendo as entradas classificadas corretamente na diagonal e as classificadas erroneamente fora da diagonal com o número de estados e a porcentagem em relação ao número total de observações representado em ambos. A última linha e coluna mostra a porcentagem de todos os estados que são classificados corretamente (verde) e incorretamente (vermelho) pertencente a cada classe. O TPOT alcançou uma performance total de 73, 8%.

um conjunto de dados contendo poucos exemplos utilizando o TPOT, demonstra-nos que ML pode ser uma nova ferramenta em potencial para investigar emaranhamento preso em sistemas de alta dimensionalidade.

3.3.2.2.4 Regressão da Robustez Generalizada

O problema de regressão para o ML é um problema mais complexo que a classificação, pois ele é entendido como um problema de múltipla classificação com infinitas classes. Aqui, o alvo é a robustez generalizada, como definida na Equação 90 e calculada para cada exemplo contido em nosso conjunto de dados através de uma ϵ-OEW, com $\epsilon = 10^{-5}$.

Apesar de começar com valores de baixa precisão, e não possuir uma conjunto de dados grande para preencher as infinitas classes, autoML ainda é capaz de recuperar a tendência da linha do quantificador. A ideia aqui é utilizar a regressão para validar o classificador anterior e estimar a quantidade de emaranhamento do estado em algum grau. Assim, altos valores para a GR prevista pode nos dar confiança sobre o emaranhamento de um estado desconhecido.

Na Tabela 5 apresentamos uma comparação entre as diferentes técnicas de ML para o problema de regressão, com o melhor modelo destacado em negrito. Como Auto-sklearn ainda é muito incipiente nessa tarefa, nós optamos por tirá-lo desta comparação. O autoML também provou ser eficiente na tarefa de regressão, apresentando um melhor resultado que as técnicas padrões. Para alcançar uma melhor performance, AutoKeras possui um *fine-tunning* extra, chamado *final fit*, que leva uma hora extra além das seis horas da janela de tempo utilizada para os algoritmos. Esse *final fit* é responsável pela redução de quase 5% do erro médio absoluto (*Mean Absolute Error*, MAE).

ML technique	MAE
TPOT	0.0373
AutoKeras	**0.0335**
Auto-PyTorch	0.0347
ANN	0.0350
SVM	0.0383
KNN	0.0500
DT	0.0649
RF	0.0463

Tabela 5 – Diferentes técnicas de ML aplicadas ao conjunto de dados e seu erro absoluto médio (MAE) no conjunto de teste. As técnicas de autoML são TPOT, AutoKeras, e Auto-Pytorch. Os métodos convencionais são redes neurais artificiais (ANN), *support vector machine* (SVM), k primeiros vizinhos (KNN), árvore de decisão (DT), *random florest* (RF).

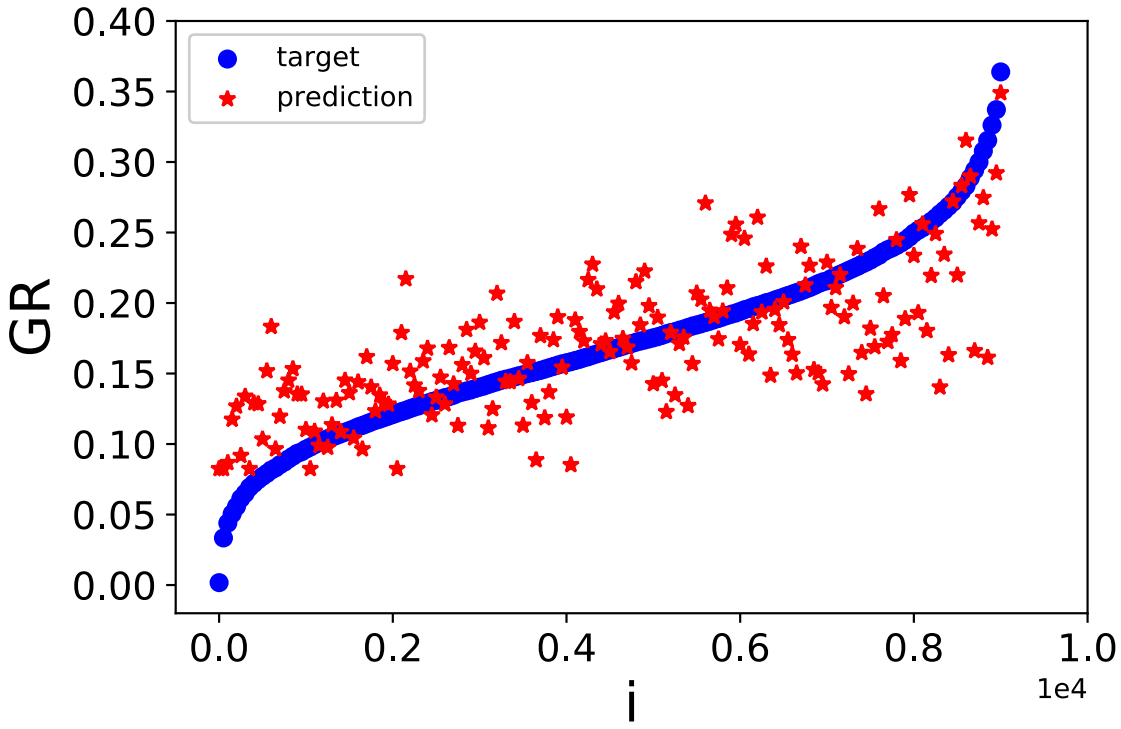


Figura 18 – Em azul (círculo) a Robustez Generalizada e em vermelho (estrela) a predição do ML considerando em torno de 9000 (reduzido em um fator de cinquenta para melhor visualização) pontos do conjunto de treinamento após um modelo de autoML ter sido treinada e implementada utilizando o pacote AutoKeras, com um erro absoluto médio (MAE) de $0,0335$. O eixo horizontal representa o rótulo i do estado ρ_i.

Nós mostramos os valores alvos (organizado do menor para o maior) e os respectivos valores fornecidos pela regressão utilizando AutoKeras na Figura 18, com MAE de $L1 = 0.0335$. Com o objetivo de uma melhor visualização do gráfico, reduzimos a densidade de pontos por um fator de cinquenta.

No caso em que queremos um modelo com uma descrição mais acurada do quantificador, devemos aumentar o número de exemplos do conjunto de dados, de maneira que a densidade de pontos em cada uma das infinitas classes represente o conjunto de acordo.

Nós podemos estabelecer um limiar no qual podemos acreditar na predição do modelo como uma função do MAE alcançado. Por exemplo, podemos definir um limiar conservador,$\delta = 3 \times L1 \approx 0,1$, então, quando o modelo prevê um GR maior que δ nós podemos dizer que o estado é emaranhado. Vale ressaltar que para valores menores ($GR < 0,1$), o valor previsto é quase um limite superior, ou seja, o verdadeiro valor é menor do que o previsto. Por outro lado, valores maiores ($GR > 0,25$) o modelo estabeleceu um limite superior e desta maneira o verdadeiro valor é maior que o previsto. Para a região intermediária ($0,1 < GR < 0,25$), nós não podemos acessar

os limites superior/inferior, ele satisfaz o critério de ser maior que o limiar δ e desta maneira podemos concluir que o estado é emaranhado.

3.3.3 Conclusão.

Neste trabalho, nós criamos uma metodologia para a criação de estados emaranhados presos uniformemente distribuídos pelo espaço de Hilbert e aplicamos a diversas técnica de ML e autoML para criar um classificador de emaranhamento em um sistema de dois qutrits. Estudamos dois sistema distintos, o primeiro referente a uma família de estados de um parâmetro de dois qutrits chamada de estados Horodecki e a segunda a estados de dois qutrits uniformemente distribuídos no espaço de Hilbert. Para os estados de Horodecki nossa rede treinada foi capaz de obter uma classificação de aproximadamente 100% para todas as classes

Nós conseguimos aplicar nosso *framework* com sucesso até mesmo na instância mais difícil do problema, quando o critério de Perez-Horodecki não consegue identificar se um estado quântico é emaranhado ou separável. Com o autoML para o problema de classificação binária (SEP *vs.* PPTES), TPOT obteve uma acurácia de 74,1

Para alcançar modelos com uma performance maior em qualquer dos testes apresentados, devemos aumentar o número de exemplos dentro do conjunto de dados. Nós treinamos nossos modelos utilizando um conjunto de dados que possui tanto estados PPTES aleatórios como artificiais, e possuímos uma evidência numérica forte de que criar estados PPTES artificiais é uma boa estratégia para contornar o problema de baixa probabilidade de sorteio de estados PPTES utilizando a medida de Hilbert-Schimdt.

O esforço computacional de criar um conjunto de dados maior pode valer a pena quando queremos decidir se um estado quântico aleatório é emaranhado eficientemente e com alta probabilidade. Desta forma, um repositório global de estados corretamente rotulados poderia ajudar a comunidade a treinar melhores máquinas e validar os resultados, poderíamos ter um grande número de grupos trabalhando com o mesmo conjunto de dados. Esse tipo de abordagem também poderia auxiliar em escalar o *framework* para maiores dimensões do espaço de Hilbert e/ou cenários com um maior número de subsistemas. As limitações em escalar o problema são:

- Conforme a dimensão aumenta a probabilidade de sortearmos estados SEP / PPTES decrescem super-exponencialmente em relação à dimensão (AUBRUN; SZAREK, 2006).
- Aumentando o número de subsistemas, o número de possíveis bipartições aumenta combinatorialmente e assim o custo computacional de construir OEWs para emaranhamento multipartite genuíno é ainda maior.

- Os requisitos experimentais para obter as medidas tomográficas também não escalam amigavelmente, já que ela aumenta quadraticamente com a dimensão.

Podemos também concluir que, dado à dificuldade do problema, nós temos fortes indicações que o autoML pode ser benéfico para tarefas de aprendizado supervisionado de todos os tamanhos e escalas, incluindo problemas com conjunto de dados pequenos, pavimentando o caminho para futuras investigações utilizando este *framework*.

4 QBOOST PARA REGRESSÃO.

Neste capítulo iremos adaptar o algoritmo QBoost (NEVEN *et al.*, 2009, 2012, 2008), um modelo de ensemble learning quântico desenvolvido para a tarefa de classificação binária, para ser capaz de resolver problemas de regressão. Aplicar a adaptação na solução de equações diferenciais parciais (EDP), especificamente, na solução da equação de Burgers unidimensional com viscosidade. O capítulo é dividido em cinco seções distintas. A primeira se refere a uma exposição sobre computação quântica adiabática. A segunda se refere a uma exposição do algoritmo QBoost e a adaptação realizada neste trabalho. A terceira descreve o algoritmo híbrido completo para regressão de EDP utilizado. A quarta nós mostramos e discutimos nossos resultados. Por último, a quinta seção apresenta a conclusão deste trabalho. Os códigos e dados utilizados neste trabalho estão disponíveis no seguinte endereço eletrônico: https://bitbucket.org/quantumcomp/qboost.git.

4.1 COMPUTAÇÃO QUÂNTICA ADIABÁTICA.

O modelo de computação quântica adiabática (AQC) explora o teorema adiabático para solucionar problemas de optimização. O teorema adiabático nos diz que dado um sistema inicialmente no *n*-ésimo autoestado de energia (normalmente o estado fundamental), o estado final do sistema permanece no mesmo autoestado se o sistema evoluir adiabaticamente (MCGEOCH, 2014). Assim, o método AQC pode resolver problemas de optimização evoluindo um hamiltoniano inicial, fácil de preparar no estado fundamental, para um hamiltoniano final, que é a codificação do problema de optimização, que permanece no estado fundamental. Em sua forma ideal AQC permite que seja realizada computação universal, o que significa que pode simular outros modelos de computação universal como o modelo de circuito, com no máximo um aumento polinomial de recursos (AHARONOV *et al.*, 2004).

Tipicamente, o modelo universal de AQC é referido como um modelo "*non-stoquastic*", ele pode simular hamiltonianos *non-stoquastic* que possui tanto elementos positivos e negativos fora da diagonal em sua representação matricial. Isso permite que o modelo AQC possa resolver qualquer problema computável por uma máquina de Turing, o que o torna em modelo universal (CRUZ-SANTOS; MORALES-LUNA, 2014). Desta maneira, se um problema pode ser desenhado para um modelo circuital, ele pode ser mapeado em um modelo AQC. Também, é importante notar que a computação no modelo AQC não precisa ocorrer no estado fundamental, o que significa que o problema pode começar no *n*-ésimo autoestado do Hamiltoniano inicial e terminar no mesmo autoestado do Hamiltoniano final de acordo com o teorema adiabático. Entretanto, como AQC normalmente é utilizada para solucionar problemas de optimização, o Hamiltoniano inicial é preparado no estado fundamental, fazendo com que o Hamil-

toniano final termine no estado fundamental, codificando a solução do problema de optimização, menor valor da função custo, (CRUZ-SANTOS; MORALES-LUNA, 2014).

Hardwares para Quantum Annealing tentam incorporar o algoritmo de AQC para solucionar problemas computacionais codificando o problema de optimização no estado fundamental do Hamiltoniano final. Deste modo, Quantum Annealing é um modelo não-universal, pois pode simular apenas Hamiltonianos "*Stoquastic*", que pode ter apenas elementos não-positivos fora da diagonal em suas representações matriciais (HORMOZI *et al.*, 2017).

De maneira geral o bloco de construção da maioria dos modelos de computação quântica é o qubit, estado quântico de dois níveis. Uma maneira conveniente de apresentar o estado quântico puro do qubit é através da base computacional $\{|0\rangle, |1\rangle\}$, que é tipicamente representada pelo vetor de estado $|\psi\rangle$ no espaço de Hilbert (SAKURAI; COMMINS, 1995):

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{97}$$

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \rightarrow \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \tag{98}$$

onde α e β são valores complexos da amplitude de probabilidade dos estados do qubit. O módulo quadrado das amplitudes de probabilidade determinam a probabilidade do estado do qubit ser medido como $|0\rangle$ ou $|1\rangle$ e ele deve ser normalizado para que o estado do qubit seja um estado quântico válido (WOLF, 2019),

$$|\alpha|^2 + |\beta|^2 = 1. \tag{99}$$

Podemos generalizar essa descrição para descrever um estado $|\Psi\rangle$ com múltiplos qubits como:

$$|\Psi\rangle = \sum_{i=0}^{N} \alpha_i \left|\psi_i^1\right\rangle \otimes \cdots \otimes \left|\psi_i^n\right\rangle \tag{100}$$

com

$$\sum_{i=0}^{N} |\alpha_i|^2 = 1. \tag{101}$$

As operações em um estado quântico para evoluir o sistema são feitas através da aplicação de operações unitárias no estado quântico inicial do sistema. A operação unitária $\hat{U}$ pode ser representada como:

$$|\psi(t_2)\rangle = \hat{U}(t_2, t_1)\,|\psi(t_1)\rangle\,, \tag{102}$$

onde $|\psi(t_2)\rangle$ é o estado no instante t_2 após a evolução unitária $\hat{U}(t_2, t_1)$ que leva o estado do instante t_1 para o instante t_2.

Para que a operação U seja unitária é necessário que a condição, $\hat{U}\hat{U}^\dagger = \hat{U}^\dagger\hat{U} = 1$ seja satisfeita.

Finalmente, a última parte para completar a descrição do modelo de computação quântica é o sistema de medições, que é uma parte crucial da computação. Por causa da natureza probabilística do resultado da medição que corresponde ao observável físico A no sistema, o valor medido de qualquer observável no sistema deve ser calculado como um valor esperado. Para cada quantidade física mensurável no sistema, deve existir um operador Hermitiano $\hat{A}$ associado a ele. A probabilidade p de medir o sistema em um estado específico m é apresentado como (NIELSEN; CHUANG, 2000);

$$p(m) = \langle\psi|\,\hat{A}_m\hat{A}_m^\dagger\,|\psi\rangle\,. \tag{103}$$

Não existem passos de tempo discretos na AQC. Ao invés, como mostrado na Figura 19, o estado dos qubits evolui gradualmente de acordo com certas forças representadas pelo Hamiltoniano $H(t)$ que faz com que o sistema evolua do Hamiltoniano inicial H_0 para o Hamiltoniano final H_f. A evolução do sistema na AQC é continua e é governada pela equação de Schrödinger, Equação 104, onde o estado de n qubits no tempo t é definido como $|\Psi(t)\rangle$ (CRUZ-SANTOS; MORALES-LUNA, 2014)

$$i\hbar\frac{\partial}{\partial t}\,|\Psi(t)\rangle = H(t)\,|\Psi(t)\rangle\,. \tag{104}$$

4.1.1 Algoritmo quântico adiabático

Em Farhi *et al.* (FARHI *et al.*, 2001), o algoritmo quântico adiabático é proposto para solucionar diversas instâncias de satisfazibilidade Booleana (3SAT), notoriamente conhecido como um problema NP-hard, usando evolução adiabática. Eles sugerem que o algoritmo quântico adiabático pode solucionar problemas de optimização difíceis se formulados como um problema de minimização de energia.

O algoritmo para solucionar um problema de optimização P é descrito da seguinte forma:

1. O sistema é iniciado no estado fundamental de um hamiltoniano H_0, facilmente preparável.

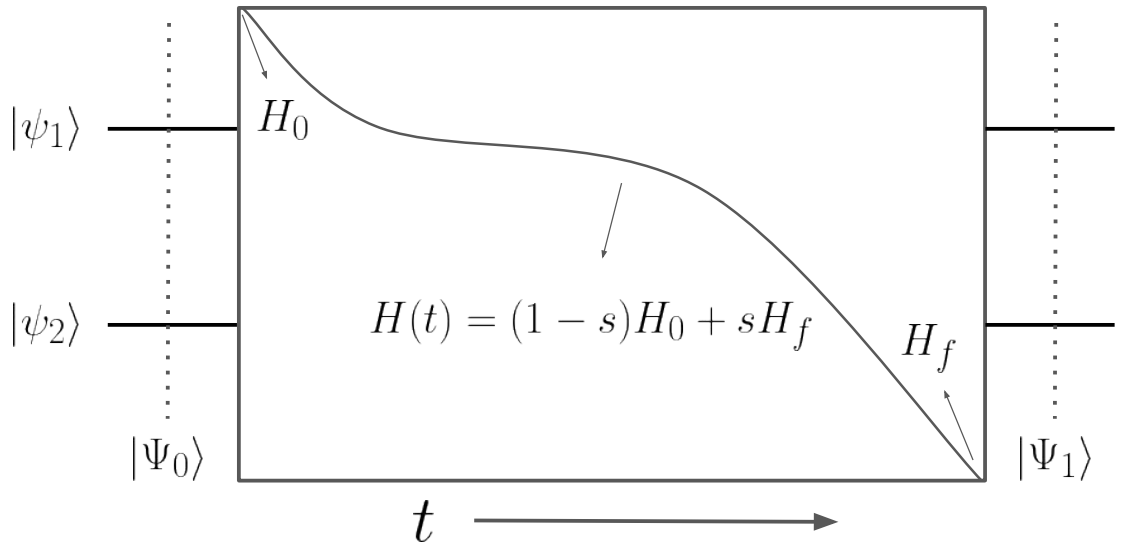


Figura 19 – Modelo de computação quântica adiabática. Onde os estados $|\psi_1\rangle$ e $|\psi_2\rangle$ são estados de qubits do sistema, $|\Psi_1\rangle$ e $|\Psi_2\rangle$ é o estado do sistema em diferentes momentos da evolução do sistema e $s = \frac{t}{T}$, sendo T o tempo total da evolução do sistema.

2. O subsistema é evoluído adiabaticamente durante um tempo T, para um hamiltoniano final, H_f, de maneira que no estado fundamental deste esteja codificada a solução do problema.

   O Hamiltoniano final evolui de acordo com:

$$H(t) = \left(1 - \frac{t}{T}\right) H_0 + \frac{t}{T} H_f = (1-s)H_0 + sH_f, \tag{105}$$

onde T é o tempo de evolução total do sistema e $s = \frac{t}{T}$.

Este algoritmo explora o teorema adiabático. O teorema em sua forma original (BORN; FOCK, 1928) nos diz que um sistema físico permanece no autoestado instantâneo se uma dada perturbação atua suficientemente devagar e se houver um *gap* entre o autovalor e o resto do espectro do Hamiltoniano. Isso significa que se um sistema estiver no estado fundamental de H_0, ele irá permanecer no estado fundamental de H_f ao final do processo se a mudança for adiabática. Para que a evolução seja adiabática, o tempo ao qual o Hamiltoniano muda precisa ser muito maior do que a escala de tempo característica para as mudanças do sistema.

Mais formalmente de acordo com (BORN; FOCK, 1928) o teorema adiabático é enunciado como:

> "Se um sistema quântico descrito por um hamiltoniano não degenerado dependente do tempo estiver inicialmente no n-ésimo autoestado de $H(t_0)$, e se $H(t)$ evoluir de forma suficientemente lenta, então o estado do sistema no tempo t_1 permanecerá no n-ésimo autoestado de $H(t_1)$."

Existem diversas maneiras de expressar matematicamente o teorema adiabático, apresentamos o desenvolvimento do teorema para o caso não degenerado no Apêndice A.

Para conseguir alcançar uma mudança adiabática no caso da computação quântica, a escala de tempo da mudança deve ser proporcional ao inverso do *gap* mínimo entre o estado fundamental $E_0(s)$ e o primeiro estado exitado $E_1(s)$ (FARHI *et al.*, 2001)

$$T >> \frac{1}{g_{min}^2}, \tag{106}$$

com

$$g_{min} = \min_{0 \leq s \leq 1} \left(E_1(s) - E_0(s)\right). \tag{107}$$

Consequentemente, o tempo total da evolução para a mudança adiabática depende do inverso do quadrado do *gap* de energia. Para esta relação, quanto menor é o *gap* de energia, mais lentamente temos que rodar o algoritmo. No caso em que os níveis de energia se cruzam, fazendo com que $g_{min} \to 0$, o tempo de execução será $T \to \infty$. Entretanto, encontrar g_{min} para avaliar a performance do AQC é tão difícil quanto encontrar a solução do problema original. Assim, limiares sobre g_{min} são tipicamente considerados para avaliar a performance da optimização AQC (CULLIMORE *et al.*, 2012).

Isso implica que a AQC para optimização algumas vezes falha em problemas NP-hard, quando o estado fundamental e o primeiro excitado se aproximam exponencialmente um do outro (DICKSON; AMIN, 2011). Assim, para evitar o cruzamento dos níveis de energia , o algoritmo necessita ser executado em um tempo exponencial para permanecer no estado fundamental. Diversos estudos na literatura (FARHI *et al.*, 2008; DAM *et al.*, 2001; JANSEN *et al.*, 2007; AMIN; CHOI, 2009) exploram as condições sobre a qual o AQC falha.

4.1.2 Quantum Annealing

Quantum Annealing (QA) é desenvolvido para solucionar problemas de optimização difíceis, onde a solução é codificada no estado fundamental final de um hamiltoniano dependente do tempo. QA realiza um processo adiabático lento para deixar o estado fundamental do hamiltoniano inicial evoluir para o estado fundamental

do hamiltoniano final, que nos dá a solução do problema (DAS, A.; CHAKRABARTI, 2008). Entretanto, ao contrário do modelo universal AQC, o QA flexibilizam diversos aspectos físicos do ambiente do sistema, fazendo com que ela não seja um modelo universal (HORMOZI *et al.*, 2017).

Quantum annealing pode ser comparado com o algoritmo de *simulated annealing* (NISHIMORI, 2015), que é utilizado para encontrar soluções ótimas através de flutuações térmicas. No *Simulated Annealing* o algoritmo implementa o análogo do resfriamento de metais para simular o procedimento de *annealing*. Pontos aleatórios são gerados na vizinhança do atual melhor ponto óptimo e a função do problema é validada nestes pontos. Se um dos valores destes pontos é melhor (ou menor) que o atual melhor ponto, então o algoritmo aceita este ponto, e o melhor valor é atualizado de acordo. Entretanto, o ponto pode ser aceito ou rejeitado se o valor validado for pior (ou maior) do que o atual melhor ponto. A decisão de aceitar ou rejeitar o ponto é probabilístico e é baseado no valor da função densidade de probabilidade da distribuição de Boltzmann-Gibbs (ARORA, 2014).

O algoritmo de *Quantum Annealing* é similar ao *Simulated Annealing* no sentido de que o parâmetro de temperatura é análogo a força do campo de tunelamento no *Quantum Annealing*, pois ele explora o tunelamento quântico na escala quântica. A ideia por trás da produção do efeito de tunelamento quântico por um campo transverso para ajudar a evoluir o sistema para o seu estado fundamental foi proposto inicialmente por (RAY *et al.*, 1989). No *Simulated Annealing*, Figura 20, a temperatura determina a probabilidade de que um estado preso em um mínimo local seja capaz de passar pela barreira, imposta por esse mínimo, e alcançar estados de menor energia. A fraca flutuação térmica possibilita o sistema navegar por sua configuração energética e alcançar níveis de energia menores. Entretanto, a força do campo transverso no *Quantum Annealing* determina a probabilidade quântica para manipular a amplitude de todos os estados em superposição. Os qubits podem não ficar presos em mínimos locais para solucionar o problema de optimização. Classicamente, se o qubit ficar preso em um mínimo local ele irá necessitar de uma energia extra para escalar a barreira de potencial para sair do mínimo local e alcançar o mínimo global. Já no caso do *Quantum Annealing* os qubits podem tunelar através dessas barreiras enquanto eles procuram pelo menor nível de energia (TAMIR; COHEN, 2015). Assim, a vantagem do *Quantum Annealing* sobre o *Simulated Annealing* está na exploração do tunelamento quântico. Entretanto, o tunelamento quântico não garante que o *Quantum Annealing* irá performar melhor que o *Simulated Annealing* por causa da existência de *gaps* de energia exponencialmente pequenos, g_{min}, para algumas estâncias difíceis.

Para executar o algoritmo *Quantum Annealing*, flutuações quânticas são induzidas adicionando um campo magnético transverso na direção do eixo-x, representado pela matiz de Pauli $\hat{\sigma}_i^x$ atuando no qubit i, obtendo o Hamiltoniano de N-qubits depen-

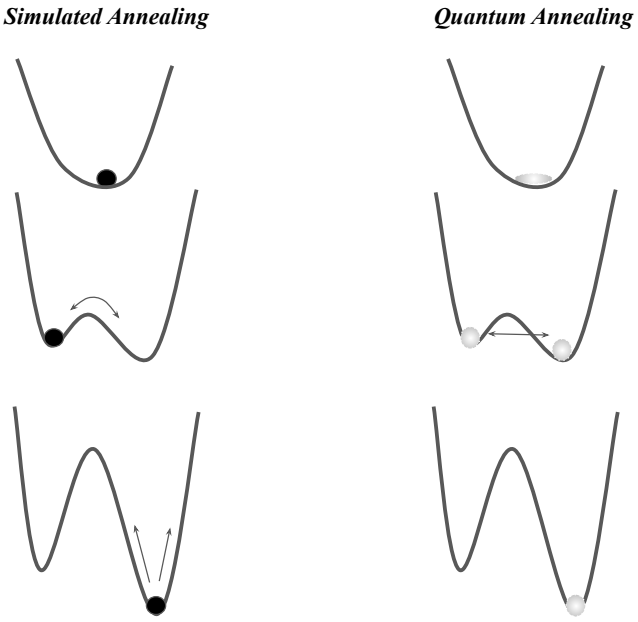


Figura 20 – Diferença entre *Simulated Annealing* e *Quantum Annealing*.

dente do tempo (CRUZ-SANTOS; MORALES-LUNA, 2014)

$$H(t) = \Gamma(t) \sum_{i=1} \Delta_i \sigma_i^x + A(t) H_f \tag{108}$$

de maneira que $\sum_{i=1} \Delta_i \sigma_i^x$ é um Hamiltoniano de tunelamento quântico não comutável adequadamente escolhido, e seu estado fundamental é uma superposição igual de estados $|0\rangle$ e $|1\rangle$. O parâmetro Δ_i é introduzido para levar em conta a probabilidade, diferente de zero, de ocorrer o tunelamento entre os dois poços de potencial dos qubits. O sistema é facilmente iniciado neste estado com $\Gamma(t) = 1$ e $A(t) = 0$. Durante o *Quantum Annealing*, o termo do campo transverso, Γ, gradualmente diminui de um para zero, e o termo A, do Hamiltoniano final, cresce de zero para um. Ao final do processo o sistema termina no estado fundamental de H_f de acordo com o teorema adiabático.

4.1.2.1 *Quantum Annealing* no Computador da D-Wave

O hardware D-Wave 2000Q implementa o algoritmo *Quantum Annealing* utilizando o seguinte Hamiltoniano:

$$\mathcal{H}_{Ising-transverso} = A(s)\mathcal{H}_0 + B(s)\mathcal{H}_f, \tag{109}$$

onde $A(s)$ e $B(s)$ são funções que controlam o processo e mudam ao longo do tempo. $\mathcal{H}_0$ é o hamiltoniano inicial e $\mathcal{H}_f$ é o hamiltoniano final, que codifica o problema a ser

optimizado. Tipicamente o processo de *annealing* começa em $s = 0$ com $A(s) >> B(s)$ e termina em $s = 1$ com $A(s) << B(s)$.

Para o sistema da D-Wave, o hamiltoniano é a soma de dois termos, o hamiltoniano inicial e o hamiltoniano final, e é representado por (CRUZ-SANTOS; MORALES-LUNA, 2014):

$$\mathcal{H}_{ising-transverso} = -A(s)\left(\sum_i \hat{\sigma}_i^x\right) + B(s)\left(\sum_i h_i\hat{\sigma}_i^z + \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z\right). \tag{110}$$

Na primeira parte do hamiltoniano, o sistema inicial é preparado no estado fundamental aplicando um forte campo magnético transverso representado pela matrix de Pauli $\hat{\sigma}_i^x$ na Equação 113, e todos os qubits i estão em uma superposição de estados de $|0\rangle$ e $|1\rangle$. O hamiltoniano inicial é diagonal na base de Hadamard, que é uma base alternativa à base computacional $\{|0\rangle, |1\rangle\}$ e é representada por $\{|+\rangle, |-\rangle\}$. A base de estados de Hadamard pode ser obtida aplicando a operação de Hadamard, Equação 117, nos estados da base computacional para criar a superposição

$$|+\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}, \tag{111}$$

$$|-\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix}. \tag{112}$$

Definimos as operações produzidas pelas matrizes de Pauli como:

- Pauli-X é equivalente a porta lógica clássica NOT. Ela mapeia $|0\rangle$ para $|1\rangle$ e $|1\rangle$ para $|0\rangle$

$$\sigma^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{113}$$

- Pauli-Y mapeia $|0\rangle$ para $i|1\rangle$ e $|1\rangle$ para $-i|0\rangle$

$$\sigma^y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}. \tag{114}$$

- Pauli-Z mapeia $|1\rangle$ para $-|1\rangle$ e $|0\rangle$ para $|0\rangle$

$$\sigma^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{115}$$

Definimos, também, a operação de Hadamard como:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{116}$$

$$\tag{117}$$

tal que

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = |+\rangle, \\ H|1\rangle &= \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle = |-\rangle. \end{aligned} \tag{118}$$

Desta maneira, para iniciar o processo de *annealing* em $t = 0$, o sistema começa no estado de menor energia do hamiltoniano inicial $\mathcal{H}_0$ na base de Hadamard, dado por:

$$|+(t=0)\rangle = \frac{1}{\sqrt{2^N}} \otimes_{i=1}^{N} (|0\rangle_i + |1\rangle_i). \tag{119}$$

No segundo termo do Hamiltoniano na Equação 110, i e j representam dois qubits vizinhos, e a interação entre eles é quantificada pela força de acoplamento $J_{i,j}$. Os coeficientes lineares h_i correspondem ao viés dos qubits, e σ_i^z e σ_j^z na Equação 110 são matrizes de Pauli que representam a quantização do campo magnético longitudinal local formando os qubits i e j, respectivamente, de acordo com o viés dos qubits h_i. Assim, a função objetivo, H_f, pode ser expressa pelo modelo de Ising.

A D-Wave suporta duas formulações, computacionalmente equivalentes, para a função objetivo:

1. **Modelo de Ising**:

$$H_{ising}(s) = \sum_{i=1} h_i s_i + \sum_{i<j} J_{i,j} s_i s_j, \tag{120}$$

onde s_i e s_j são estados que correspondem aos valores +1 e –1, que podem ser os estados de spin *up* e spin *down*.

Entretanto como estamos lidando com problemas de optimização, é mais conveniente representar os problemas utilizando valores lógicos 0 e 1 ao invés de –1 e 1.

2. ***Quadratic Unconstrain Binary Optimization*** **(QUBO)**:

$$f(x) = \sum_i Q_{i,i} x_i + \sum_{i<j} Q_{i,j} x_i x_j \tag{121}$$

Q é uma matriz triangular superior $N \times N$ de valores reais e x_i e x_j são estados que correspondem aos valores 0 e 1. Diversos problemas de optimização duros podem ser codificados propriamente no modelo QUBO (GLOVER *et al.*, 2019).

4.1.2.2 Arquitetura do hardware D-Wave

Quando um problema é submetido à unidade de processamento quântico da D-Wave (QPU), ele é traduzido e codificado em um sistema físico, esse processo é chamado de *embedding*. A arquitetura da D-Wave é simplesmente uma rede de qubits interconectados que são conectados através de "acopladores". Entretanto, os qubits na atual topologia não são totalmente conectados. Este fato deve ser levado em consideração ao mapear o problema na topologia física atual, porém, este problema normalmente é resolvido automaticamente pelo software da D-Wave. A primeira topologia desenvolvida pela D-Wave é a topologia Chimera.

A QPU da D-Wave 2000Q consiste em uma grade de 16×16 células unitárias da topologia Chimera. A célula unitária no grafo chimera é um grafo bipartite $K_{4,4}$, como mostrado na Figura 21. Cada célula unitária do grafo Chimera possui oito qubits (bolas azuis), e cada qubit é conectado com seu vizinho (conexões verdes), resultando que cada qubit se conecta com outros quatro qubits ortogonais internamente. Diferentes células unitárias são conectadas por acopladores externos (conexões vermelhas), e eles conectam os qubits paralelos na mesma linha horizontal ou vertical. Os acopladores externos dão mais liberdade para os qubits, permitindo que um qubit se conecte com no máximo seis qubits.

4.2 QBOOST

O *QBoost* foi desenvolvido por pesquisadores da *Google* e *D-Wave labs* (NEVEN *et al.*, 2009, 2012, 2008) e é o algoritmo pioneiro na área do ML quântico. O modelo básico de ML é um ensemble de K classificadores binários (*weak-learners*) $h_k(x)$, $k = 1, ..., K$, que são combinados por uma soma de pesos da forma $h(x) = sigmoid\left(\sum_{k=1}^{K} w^T f_k(x)\right)$ com $x \in \mathbb{R}^N$ e $h(x) \in \{0, 1\}$. Nós assumimos que os classificadores individuais $h_k(x)$ são treinados, e portanto os parâmetros do modelo serão omitidos.

Treinar o ensemble significa encontrar os pesos associados a cada classificador individual. Para esta tarefa, nós normalmente minimizamos dois termos simultaneamente: uma função custo, $L(w, h(x), y)$ e um termo de regularização, $R(w)$. Para o

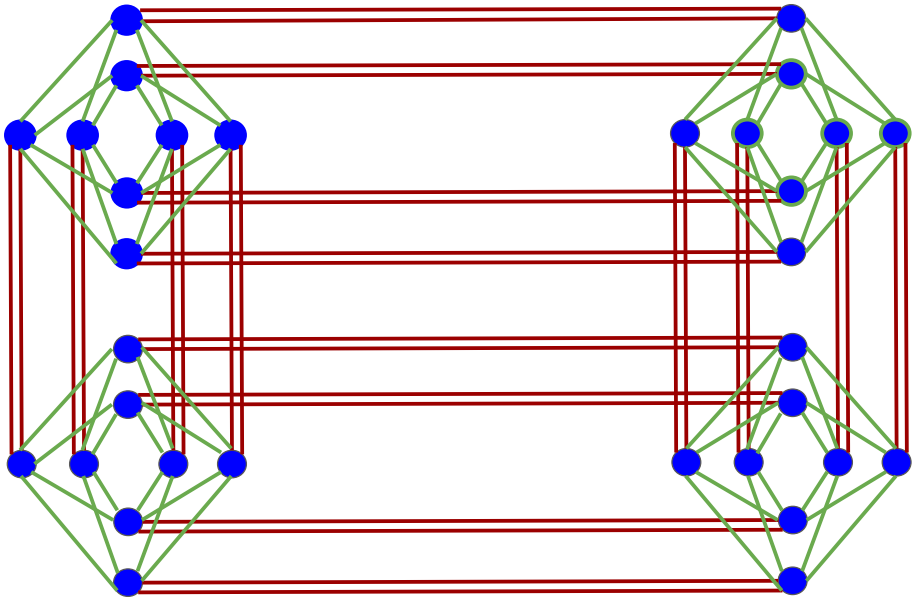


Figura 21 – Topologia do grafo Chimera, mostrando quatro células unitárias.

problema de classificação, o algoritmo minimiza o erro quadrático médio adicionado um termo de regularização L_0 para encontrar os pesos associados a cada *weak-learner*

$$J(w,h(x),y) = L(w,h(x),y) + R(w) = \frac{1}{M}\sum_{m=1}^{M}\left(h(x^{(m)}) - y^{(m)}\right)^2 + \lambda||w||_0, \tag{122}$$

onde $||.||_0$ conta o número de parâmetros diferentes de zero, λ controla a força da regularização é ajustado empiricamente, $h(x^m) = \sum_{k=1}^{K} w_k h_k(x^m)$, $y^{(m)}$ é o valor real do exemplo m e M é o número total de exemplos. Para deixarmos a equação acima na forma QUBO devemos abrir o termo quadrático.

$$J(w,h(x),y) = \frac{1}{M}\sum_{m=1}^{M}\left[\sum_{k=1}^{K} w_k h_k(x^{(m)}) - y^{(m)}\right]^2 + \lambda||w||_0 \tag{123}$$

$$J(w,h(x),y) = \frac{1}{M}\sum_{m=1}^{M}\left[\sum_{k,k'=1}^{K} w_k w_{k'} h_k(x^{(m)})h_{k'}(x^{(m)}) - \sum_{k=1}^{K} 2w_k h_k(x^{(m)})y^{(m)} + \left(y^{(m)}\right)^2\right] + \\ + \sum_{k=1}^{K} w_k\lambda. \tag{124}$$

O termo $\frac{1}{M}\sum_{m=1}^{M}\left(y^{(m)}\right)^2$ pode ser desconsiderado, já que o mesmo é uma constante positiva que não irá influenciar na optimização. Desta forma,

$$\sum_{k,k'=1}^{K} w_k w_{k'} \left[\frac{1}{M} \sum_{m=1}^{M} h_k(x^{(m)}) h_{k'}(x^{(m)}) \right] + \sum_{k=1}^{K} w_k \left[\lambda - \frac{2}{M} \sum_{m=1}^{M} h_k y^{(m)} \right], \tag{125}$$

os termos $\frac{1}{M}\sum_{m=1}^{M} h_k(x^{(m)}) h_{k'}(x^{(m)})$ e $\lambda - \frac{2}{M}\sum_{m=1}^{M} h_k(x^{(m)})$ servem como os termos de interação e força do campo para o modelo de Ising, enquanto os pesos w_k e $w_{k'}$ fazem o papel dos temor x_i e x_j.

Entretanto esta formulação exige que os pesos sejam variáveis binárias, o que impossibilita a sua aplicação em problemas de regressão. Para que seja possível tratar problemas de regressão devemos fazer algumas alterações na formulação do problema:

- Alterar o termo de regularização para a norma dois, l_2, para que o problema possa ser escrito na forma QUBO.
- Realizar uma expansão de ponto flutuante nos pesos w_k.
- Adicionar a restrição de que os pesos devem somar um, $\sum_k w_k = 1$.

Alterando o termo de regularização para a norma l_2, teremos

$$J(w, h(x), y) = \frac{1}{M} \sum_{m=1}^{M} \left[\sum_{k=1}^{K} w_k h_k(x^{(m)}) - y^{(m)} \right]^2 + \lambda ||w||_2^2 \tag{126}$$

$$\begin{aligned} J(w, h(x), y) = & \frac{1}{M} \sum_{m=1}^{M} \left[\sum_{k,k'=1}^{K} w_k w_{k'} h_k(x^{(m)}) h_{k'}(x^{(m)}) - 2y^{(m)} \sum_{k=1}^{K} w_k h_k(x^{(m)}) \right] + \\ & + \sum_{k=1}^{K} \lambda w_k w_k. \end{aligned} \tag{127}$$

Precisamos escrever nossos pesos como uma aproximação de ponto flutuante de R–bits de valores reais w_i. Seguindo a metodologia descrita em (ROGERS; SINGLETON, 2020), nós aplicamos a expansão em ponto flutuante para representar os pesos w_k da Equação 125. Para qualquer número $\chi \in [0, 2)$, a representação binária com acurácia de R qubits de resolução pode ser expressada por uma *string* de bits, $[Q_0 Q_1 Q_2 \ldots Q_R]_2$, onde $Q_r \in \{0, 1\}$ é o valor o r-ésimo bit, e o colchete indica a representação binária. Em termos da série de potência 2^{-r}, nós teremos,

$$\chi_k = \sum_{r=0}^{R-1} 2^{-r} Q_r^{(k)}. \tag{128}$$

Para representar os pesos em um domínio menos restritivo, $w_k \in [-d, 2c-d]$, nós escalamos e mudamos χ_k por,

$$w_k = c\chi_k - d. \tag{129}$$

Quando $d > 0$ e $c > d/2$, o domínio de w_k irá sempre possuir uma região positiva e uma negativa, e o valor preciso de c e d podem variar de acordo com o problema específico. Para o problema de regressão possuímos uma restrição extra, $\sum_k w_k = 1$, para reduzir uma variável e forçar uma mistura afim dos pesos. Desta maneira devemos substituir

$$\begin{aligned} \sum_{k=1}^{K} w_k &= 1, \\ \sum_{k=1}^{K-1} w_k + w_K &= 1, \\ w_K &= 1 - \sum_{k=1}^{K-1} w_k, \end{aligned} \tag{130}$$

em cada termo da Equação 127 proporcional a w_k. Mudando a notação por simplicidade $h_k(x^{(m)}) = h_k$, teremos

$$\begin{aligned} \sum_{k=1}^{K} w_k h_k &= \sum_{k=1}^{K-1} w_k h_k + w_K h_K \\ &= \sum_{k=1}^{K-1} w_k h_k + \left(1 - \sum_{k=1}^{K-1} w_k\right) h_K \\ &= \sum_{k=1}^{K-1} w_k (h_k - h_K) + h_K \\ &\sum_{k=1}^{K-1} w_k (h_k - h_K) \end{aligned} \tag{131}$$

retiramos o termo h_K pois ele não contribui para a minimização. Desta maneira teremos que

$$
\begin{aligned}
&\sum_{k=1}^{K} w_k h_k \sum_{k'=1}^{K} w_{k'} h_{k'} \\
&= \left(\sum_{k=1}^{K-1} w_k (h_k - h_K) + h_K \right) \left(\sum_{k'=1}^{K-1} w_{k'} (h_{k'} - h_K) + h_K \right) \\
&= \sum_{k,k'=1}^{K-1} w_k w_{k'} (h_k - h_K)(h_{k'} - h_K) + 2h_K \sum_{k=1}^{K-1} w_k (h_k - h_K) + h_K^2 \\
&\quad \sum_{k,k'=1}^{K-1} w_k w_{k'} (h_k - h_K)(h_{k'} - h_K) + 2h_K \sum_{k=1}^{K-1} w_k (h_k - h_K)
\end{aligned}
\tag{132}
$$

como no caso da equação 131 o termo h_K^2 não contribui com a minimização pois é uma constante, assim podemos retirá-lo da equação 132.

$$
\begin{aligned}
&\lambda \sum_{k=1}^{K} w_k w_k \\
=&\lambda \sum_{k=1}^{K-1} w_k w_k + w_K w_K \\
=&\lambda \sum_{k=1}^{K-1} w_k w_k + \lambda \left(1 - \sum_{k=1}^{K-1} w_k \right) \left(1 - \sum_{k'=1}^{K-1} w_{k'} \right) \\
=&\lambda \sum_{k,k'=1}^{K-1} \delta_{k,k'} w_k w_{k'} + \lambda \sum_{k,k'=1}^{K-1} w_k w_{k'} - 2\lambda \sum_{k=1}^{K-1} w_k + \lambda \\
&\lambda \sum_{k,k'=1}^{K-1} (1 + \delta_{k,k'}) w_k w_{k'} - 2\lambda \sum_{k=1}^{K-1} w_k .
\end{aligned}
\tag{133}
$$

Substituindo as equações 131, 132 e133 na equação 127 teremos,

$$
\begin{aligned}
J(w,h,y) = &\sum_{k,k'=1}^{K-1} w_k w_{k'} \left[\frac{1}{M} \sum_{m=1}^{M} (h_k - h_K)(h_{k'} - h_K) + \lambda (1 + \delta_{k,k'}) \right] + \\
&+ 2 \sum_{k=1}^{K-1} \left[\frac{1}{M} \sum_{m=1}^{M} (h_k - h_K)(h_K - y^{(m)}) - \lambda \right] .
\end{aligned}
\tag{134}
$$

Reescrevendo cada termo da equação 134 utilizando transformação de escala 129 teremos

$$
\begin{aligned}
& 2\sum_{k=1}^{K-1} w_k \left[\frac{1}{m}\sum_{m=1}^{M}(h_k - h_K)(h_K - y^{(m)}) - \lambda\right] \\
=& 2\sum_{k=1}^{K-1}(c\chi_k - d)\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_K - y^{(m)}) - \lambda\right] \\
& 2c\sum_{k=1}^{K-1}\chi_k\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_K - y^{(m)}) - \lambda\right],
\end{aligned}
\tag{135}
$$

onde o termo proporcional a d foi descartado pois não contribui com a minimização.

$$
\begin{aligned}
& \sum_{k,k'=1}^{K-1} w_k w_{k'}\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_{k'} - h_K) + \lambda(1+\delta_{k,k'})\right] \\
=& \sum_{k,k'=1}^{K-1}(c\chi_k - d)(c\chi_{k'} - d)\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_{k'} - h_K) + \lambda(1+\delta_{k,k'})\right] = \\
& \sum_{k,k'=1}^{K-1} c^2\chi_k\chi_{k'}\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_{k'} - h_K) + \lambda(1+\delta_{k,k'})\right] - \\
& -\sum_{k=1}^{K-1} 2cd\chi_k\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)\sum_{k'=1}^{K-1}(h_{k'} - h_K) + \sum_{k'=1}^{K-1}\lambda - \sum_{k'=1}^{K-1}\lambda\delta_{k,k'}\right] \\
=& \sum_{k,k'=1}^{K-1} c^2\chi_k\chi_{k'}\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_{k'} - h_K) + \lambda(1+\delta_{k,k'})\right] - \\
& -\sum_{k=1}^{K-1} 2cd\chi_k\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)\sum_{k'=1}^{K-1}(h_{k'} - h_K) + \lambda K\right].
\end{aligned}
\tag{136}
$$

Juntando as equações 135 e 136, obteremos

$$
\begin{aligned}
J(\chi, h, y) = & \sum_{k,k'=1}^{K-1} c^2\chi_k\chi_{k'}\left[\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)(h_{k'} - h_K) + \lambda(1+\delta_{k,k'})\right] + \\
& + 2c\sum_{k=1}^{K-1}\chi_k\left\{\frac{1}{M}\sum_{m=1}^{M}(h_k - h_K)\left[(h_K - y^{(m)}) - d\sum_{k'=1}^{K-1}(h_{k'} - h_K)\right] - \lambda(1 - dK)\right\}.
\end{aligned}
\tag{137}
$$

Aplicando a expansão de ponto flutuante, equação 128, à equação 137, obteremos

$$J(Q,h,y) = \sum_{k,k'=1}^{K-1} c^2 \left[\frac{1}{M} \sum_{m=1}^{M} (h_k - h_K)(h_{k'} - h_K) + \lambda(1 + \delta_{k,k'}) \right] \sum_{r=0}^{R-1} 2^{-r} Q_r^k \sum_{r'=0}^{R-1} 2^{-r'} Q_{r'}^{k'} +$$
$$\sum_{k=1}^{K-1} 2c \left\{ \frac{1}{M} \sum_{m=1}^{M} (h_k - h_K) \left[(h_K - y^{(m)}) - d \sum_{k'=1}^{K-1} (h_{k'} - h_K) \right] - \lambda(1 - dK) \right\} \sum_{r=0}^{R-1} 2^{-r} Q_r^k. \tag{138}$$

Como os qubits físicos no processador da D-Wave são acessados por um índice linear de uma dimensão, é necessário fundir os índices (k,r) em um único índice $I = R, R+1, \ldots, KR-1$ utilizando

$$I(k,r) = kR + r, \tag{139}$$

com o mapa inverso sendo dado por $k_I =$ e $r_I = I \mod R$. Agora podemos proceder com a quantização da variável Q_I, que satisfaz a equação de autovalores $\hat{Q}_I |Q_I\rangle = Q_I |Q_I\rangle$, em que a condição de idempotência $\hat{Q}_I^{\,2} = \hat{Q}_I$ implica em autovalores $Q_I \in \{0,1\}$. Os autovetores são representados por $|Q\rangle = |Q_R\rangle \otimes |Q_{R+1}\rangle \otimes \cdots \otimes \left| Q_{(KR-1)} \right\rangle$. Desta maneira, aplicando a fusão dos índices expressos na equação 139 e a quantização da Equação 138 compõem o seguinte hamiltoniano:

$$H(Q,h,y) = \sum_{I=R, I' \neq Q_I Q_{I'} I'}^{KR-1} c^2 2^{-(r_I + r_{I'})} \left[\frac{1}{M} \sum_{m=1}^{M} (h_k - h_K)(h_{k'} - h_K) + \lambda(1 + \delta_{k,k'}) \right] Q_I Q_{I'} +$$
$$+ \sum_{I=R}^{KR-1} c 2^{-r_I+1} \left\{ \frac{1}{M} \sum_{m=1}^{M} (h_{k_I} - h_K) \left[(h_K - y) - d \sum_{k'=1}^{K-1} (h_{k_{I'}} - h_K) + c 2^{-r'-1} (h_{k_I} - h_K) \right] \right\} -$$
$$- \sum_{I=R}^{KR-1} c 2^{-r_I+1} \lambda \left(1 + dK + c 2^{-r_I}\right) Q_I. \tag{140}$$

Comparando a equação 140 com o hamiltoniano do problema QUBO dado por,

$$H_{QUBO} = \sum_i \alpha_i Q_i + \sum_{i \neq j} \beta_{ij} Q_i Q_j, \tag{141}$$

podemos identificar os coeficientes do hamiltoniano final que são os parâmetros de entrada do *quantum annealing*,

$$\alpha_i = c2_{-r_i+1}\frac{1}{M}\sum_{m=1}^{M}\sum_{m=1}^{M}(h_{k_i}-h_K)\left[(h_K-y)-d\left(\sum_{k_j=1}^{K-1}(h_{k_j}-h_K)\right)+c2^{-r_i-1}(h_{k_i}-h_K)\right]- \\ -c2_{-r_i+1}\lambda(1+dK+c2^{-r_i}) \tag{142}$$

e

$$\beta_{ij} = c^2 2^{-(r_i+r_j)}\left[\frac{1}{M}\sum_{m=1}^{M}(h_{k_i}-h_K)(h_{k_j}-h_K)+\lambda(1+\delta_{k_i,k_j})\right]. \tag{143}$$

4.3 ALGORITMO HÍBRIDO PARA SOLUCIONAR EDP.

Propomos um algoritmo híbrido que é capaz de realizar uma regressão em soluções de EDP. O componente clássico do algoritmo consiste em criar uma ANN capaz de aproximar a solução da EDP, enquanto a componente quântica irá criar um algoritmo de *boosting* para o *ensemble* das ANN clássicas para gerar uma ANN mais forte. O passo quântico do algoritmo é realizado através do *quantum annealing*, que obtem a solução mais próxima ao estado fundamental do hamiltoniano (Equação 140).

4.3.1 Componente Clássica

Para introduzir a parte clássica do algoritmo, vamos considerar uma EDP parabólica d–dimensional:

$$\begin{aligned} &\frac{\partial}{\partial t}u(t,x)+\mathcal{L}u(t,x)=0;\quad (t,x)\in[0,T]\times\Omega, \\ &u(0,x)=u_0(x), \\ &u(t,x)=g(t,x),\quad x\in\partial\Omega, \end{aligned} \tag{144}$$

em que $x \in \Omega \subset \mathbb{R}^d$, t é o tempo, $\mathcal{L}$ é um operador contendo derivadas espaciais, $u(t,x)$ é a solução da EDP, $u_0(x)$ é a condição inicial e $g(t,x)$ é a condição de contorno. Essa parte do algoritmo realiza uma regressão na solução da EDP através de uma ANN, aproximando a solução, $u(t,x) \approx f(t,x,\theta)$, onde θ é um conjunto de variáveis reais usada para representar os parâmetros internos da ANN. Seguindo o trabalho (SIRIGNANO; SPILIOPOULOS, 2018), nós utilizamos como métrica de aproximação a seguinte função custo $J(f)$,

$$
\begin{aligned}
J(f) =& \gamma_1 \left|\left| \frac{\partial}{\partial t} f(t,x,\theta) + \mathcal{L} f(t,x,\theta) \right|\right|^2_{[0,T]\times\Omega,\rho_1} + \\
&+\gamma_2 ||f(t,x,\theta) - g(t,x)||^2_{[0,T]\times\Omega,\rho_2} + \\
&+\gamma_3 ||f(0,x,\Omega) - u_0(0,x)||^2_{\Omega,\rho_3},
\end{aligned}
\tag{145}
$$

em que $||f(y)||^2_{\mathcal{Y},\rho} = \int_{\mathcal{Y}} |f(y)|^2 \rho(y) dy$ e $\rho(y)$ é a densidade de probabilidade sobre o domínio $y \in \mathcal{Y}$ e γ_i são pesos tal que $\sum_i \gamma_i = 1$. $J(f)$ mede o quão bem $f(t,x,\theta)$ satisfaz os operadores da EDP, condição de contorno e condição inicial.

O papel da ANN é encontrar os parâmetros θ que minimizam a função custo $J(f)$. Conforme $J(f) \to 0$ então a solução da ANN se aproxima da solução da EDP, ou seja, $f(t,x,\theta) \to u(t,x)$. A vantagem do algoritmo em relação a outros métodos padrões em dinâmica de fluidos computacional é que não se faz necessário criar uma malha para discretizar o espaço, que é computacionalmente caro.

Como descrito em (SIRIGNANO; SPILIOPOULOS, 2018), o algoritmo consiste dos seguintes passos:

1. Gerar pontos aleatórios $(t_n, x_n) \in [0,T]\times\Omega$, $(\tau_n, z_n) \in \partial\Omega$ e sortear pontos aleatórios a_n no domínio Ω de acordo com as respectivas densidades de probabilidades ρ_1, ρ_2 e ρ_3.

2. Calcular o erro quadrático $G(\theta_n, s_n)$ dos pontos sorteados em 1. usando o conjunto $s_n = \{(t_n, x_n), (\tau_n, z_n, a_n)\}$

$$
\begin{aligned}
G(\theta_n, s_n) =& \gamma_1 \left(\frac{\partial f}{\partial t} + \mathcal{L} f(t_n, x_n, \theta_n) \right)^2 + \\
&+\gamma_2 (f(\tau_n, z_n, \theta_n) - g(\tau_n, z_n))^2 + \\
&+\gamma_3 (f(0, a_n, \theta_n) - u_0(a_n))^2.
\end{aligned}
\tag{146}
$$

3. Atualiza os pesos θ_n usando o passo de gradiente

$$
\theta_{n+1} = \theta_n - \alpha_n \nabla_\theta G(\theta_n, s_n), \tag{147}
$$

onde α_n é chamado taxa de aprendizado e decresce quando n cresce.

4. Repitir o último passo até que o critério de convergência, $G(\theta_n, s_n) \leq \epsilon$, seja alcançado.

O gradiente $\nabla_\theta G(\theta_n, s_n)$ é um estimador de $\nabla_\theta J(f(., \theta_n))$,

$$\mathbb{E}[\nabla_\theta G(\theta_n, s_n)|\theta_n] = \nabla_\theta J(f(., \theta_n)). \tag{148}$$

Neste sentido o algoritmo gradiente descendente estocástico em média irá realizar passos descendo em direção da função objetivo $J(f(., \theta))$, isto é, $J(f(., \theta(n+1))) < J(f(., \theta_n))$ e desta maneira θ_{n+1} é um estimador melhor que θ_n.

4.3.2 Componente Quântica

A parte quântica do algoritmo consiste em realizar um *boosting* no passo de *ensemble learning* através do método de quantum annealing. Para realizar este passo utilizamos uma versão adaptada do *QBoost* (NEVEN *et al.*, 2009, 2012, 2008), descrita na seção anterior, para que o mesmo seja capaz de solucionar problemas de regressão. Especificamente, o trabalho foca no estudo do problema de regressão através de um ensemble na forma $f(t, x) = \sum_{k=1}^{K} w_k h_k(t, x)$, onde $(t, x) \in [0, T] \times \Omega$, $h(t, x) \in \mathbb{R}^k$ é um vetor de saída de uma ANN previamente treinada em (t, x), e $\mathbf{w} \in \mathbb{R}^K$ é um vetor de pesos a ser optimizado de maneira que $\sum_i w_i = 1$.

Nós descrevemos o algoritmo como:

1. Validamos os *weak-learners* produzida pela parte clássica do algoritmo para criar o conjunto de treinamento e teste com M e N amostras, respectivamente.
2. Calcular os coeficientes do QUBO com as equações 142 e 143.
3. Usar os coeficientes do QUBO como entrada do quantum annealer da D-Wave.
4. Ler a lista binária de saída do sistema da D-Wave e usar a Equação 129 para reconstruir o peso ótimo.
5. Construir a saída final do ensemble $f(t, x)$ com os pesos reconstruídos.

4.4 RESULTADOS E DISCUSSÃO

Para validar nosso método utilizamos uma EDP que é bem conhecida e possui uma solução analítica. O modelo utilizado é a equação de Burgers em uma dimensão com viscosidade, ν (BURGERS, 1948). Considerando a Equação 144 no domínio $(t, x) \in [0, T] \times [0, 2\pi]$ com as seguintes condições inicial e de contorno,

$$\begin{aligned} &\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}, \\ &u(0, x) = -2\frac{\nu}{\phi(0, x)}\frac{d\phi}{dx} + 4, \ x \in [0, 2\pi], \\ &u(t, 0) = u(t, 2\pi), \ t \in [0, T]. \end{aligned} \tag{149}$$

Sobre estas condições, a equação de Burgers apresenta uma solução analítica na forma,

$$u(t,x) = -2\frac{\nu}{\phi(t,x)}\frac{d\phi}{dx} + 4, \quad (t,x) \in [0,\, T] \times [0, 2\pi], \tag{150}$$

onde

$$\phi(t,x) = \exp\left\{\frac{-(x-4t)^2}{4\nu(t+1)}\right\} + \exp\left\{\frac{-(x+4t-2\pi)^2}{4\nu(t+1)}\right\}. \tag{151}$$

Na Tabela 6 apresentamos os *weak-learners* correspondentes a parte clássica do algoritmo usados para criar o ensemble (os selecionados estão destacados em negrito). Todas as ANN foram treinadas com 2.000 instâncias através de sorteio aleatório de 10 pontos no espaço com $x \in [0, 2\pi]$ para cada um dos tempos $t \in \{0,\ 0,05,\ 0,15,\ 0,20,\ 0,30,\ 0,35,\ 0,40,\ 0,50\}$ para o conjunto de treinamento e $t \in \{0,10,\ 0,25,\ 0,45\}$ para o conjunto de teste, que resulta em $M = 160.000$ e $N = 60.000$ instancias de treinamento e teste, respectivamente. As densidades de probabilidade ρ_1, ρ_2 e ρ_3 foram obtidas considerando os valores da função $u(t,x)$, $u(0,x)$ e $u(t,0) = u(t,2\pi)$, respectivamente, validada para cada par (t,x) como definido anteriormente. Nós escolhemos $\epsilon = 550$. É possível perceber que não existe relação predeterminada entre a arquitetura, número de neurônios e número de camadas escondidas com a função custo. A escolha dos *weak-learners* que compõem o ensemble segue os seguintes critérios:

- O ensemble deve possuir o *weak-learner* com a melhor performance entre todos. Desta maneira nosso *framework* pode ser validado em relação a performance do ensemble.
- Os demais *weak-learners* devem possuir uma performance média para ruim. Para demonstrar que a utilização de *weak-learners* fracos podem produzir um ensemble bom.

Nós rodamos o nosso algoritmo, solucionando o problema QUBO dado pela Equação 141 através do pacote Ocean da D-Wave de três maneiras distintas:

- *Exact Solver*. Esta função diagonaliza o Hamiltoniano, representado por 141, e encontra o estado fundamental, e assim, a solução do problema.
- *Quantum annealing*. Foi rodado no computador dw_2000Q_6, que opera com uma temperatura média de $13,5 \pm 1,0$ *mK* (INC, 2019). Cada chamada do processador consiste em 10.000 rodadas, com o tempo máximo de *annealing* sendo 20 μs.

Tabela 6 – *Weak-learners* da equação de Burgers unidimensional com viscosidade. O *weak-learner* $[l_1, l_2, \cdots, l_i, \cdots, l_q]$ representa uma ANN com q camadas contendo l_i neurônios na i–ésima camada. Os valores da função custo (equação 146) foram calculadas usando o conjunto de teste. O parâmetro tempo/# épocas representa a dificuldade de treinamento da ANN. As linhas em negrito correspondem aos *weak-learners* selecionados para compor o ensemble.

weak-learner	Função custo(teste)	tempo/# épocas	(#neurônios) $\times$ (#camadas)
[10,30,60,90,110,150]	0.015924	0.973962	2700
[180,90,30,30,90,180]	0.011367	1.027333	3600
[30,90,180,180,90,30]	0.010340	1.380745	3600
[60,180,60,180,60,180]	0.010368	1.274568	4320
[20,20,20,20,20,20]	**0.002017**	**0.284775**	**720**
[10,20,30,40,50,60]	0.004267	0.473657	1260
[60,50,40,30,20,10]	0.004805	0.526772	1260
[60,30,10,10,30,60]	**0.008058**	**1.266518**	**760**
[10, 30, 60, 60, 30, 10]	0.007132	1.788494	760
[20, 60, 20, 60, 20, 60]	0.002611	1.511132	1440
[150,90,30,90,150]	0.011307	1.568148	2550
[30,90,150,90,30]	0.008128	1.767931	1950
[30, 120, 30, 120, 30]	0.007504	1.046415	1650
[120, 30, 120, 30, 120]	**0.004283**	**1.619587**	**2100**
[10, 30, 50, 70, 90]	0.006257	1.249093	1250
[90, 70, 50, 30, 10]	0.009561	1.480899	1250
[25, 25, 25, 25, 25]	0.003077	0.523823	625
[15, 15, 15, 15, 15]	0.002436	0.304417	375
[150, 60, 60, 150]	0.012856	0.657789	1680
[60, 150, 150, 60]	0.013863	0.809011	1680
[90, 30, 90, 30]	0.001743	0.456623	960
[30, 90, 90, 30]	0.006373	0.472136	960
[90, 60, 30, 10]	0.005259	0.418490	760
[10, 30, 60, 90]	0.013063	0.399743	760
[30, 30, 30, 30]	0.003695	0.312810	480
[60, 10, 10, 60]	0.002284	0.308086	560
[10, 60, 60, 10]	0.003263	0.337634	560
[45, 10, 45, 10]	0.005858	0.245707	440
[10, 45, 10, 45]	0.005785	0.256790	440
[40, 30, 20, 10]	0.002407	0.279718	400
[10, 20, 30, 40]	0.002154	0.274698	400
[10, 10, 10, 10]	0.002184	0.193450	160
[150, 60, 150]	0.002990	0.805101	1080
[60, 150, 60]	0.013825	1.251153	810
[10, 45, 90]	0.004367	0.911909	435
[90, 45, 10]	0.005408	0.603616	435
[40, 40, 40]	0.002905	0.469319	360
[90, 30, 90]	0.005547	1.179521	630
[30, 90, 30]	0.003455	1.066604	450
[10, 20, 30]	**0.001574**	**0.276350**	**180**
[30, 20, 10]	0.011371	0.204170	180
[10, 10, 10]	0.001884	0.158107	90

- *Simulated Annealing*. Como no caso do *Quantum Annealing*, cada vez que chamamos o algoritmo, 10.000 rodadas são realizadas com 1.000 sweeps cada e β, que é um parâmetro proporcional ao inverso da temperatura, variando entre $0,1$ e 4. Nenhuma optimização foi realizada de maneira a encontrar melhores soluções via *simulated annealing*..

Os pesos são representados no intervalo $w_k \in [-3, 3]$, o que significa que $c = d = 3$. A dependência do termo de regularização em função da função custo é mostrado na Figura 22. Podemos perceber que qualquer aumento no termo de regularização resulta em um aumento na função custo. Podemos concluir que o melhor resultado ocorre quando o termo de regularização é nulo. Essa conclusão é suportada pelo fato de que não observamos a presença de *overfitting* e *underfitting* na Figura 23. Na Figura 23, mostramos a comparação entre a solução produzida pelo ensemble, com $\lambda = 0$, e a solução analítica da equação de Burgers unidimensional para os tempos presentes no conjunto de teste. É possível observar que a solução produzida por cada um dos métodos é muito similar a solução analítica.

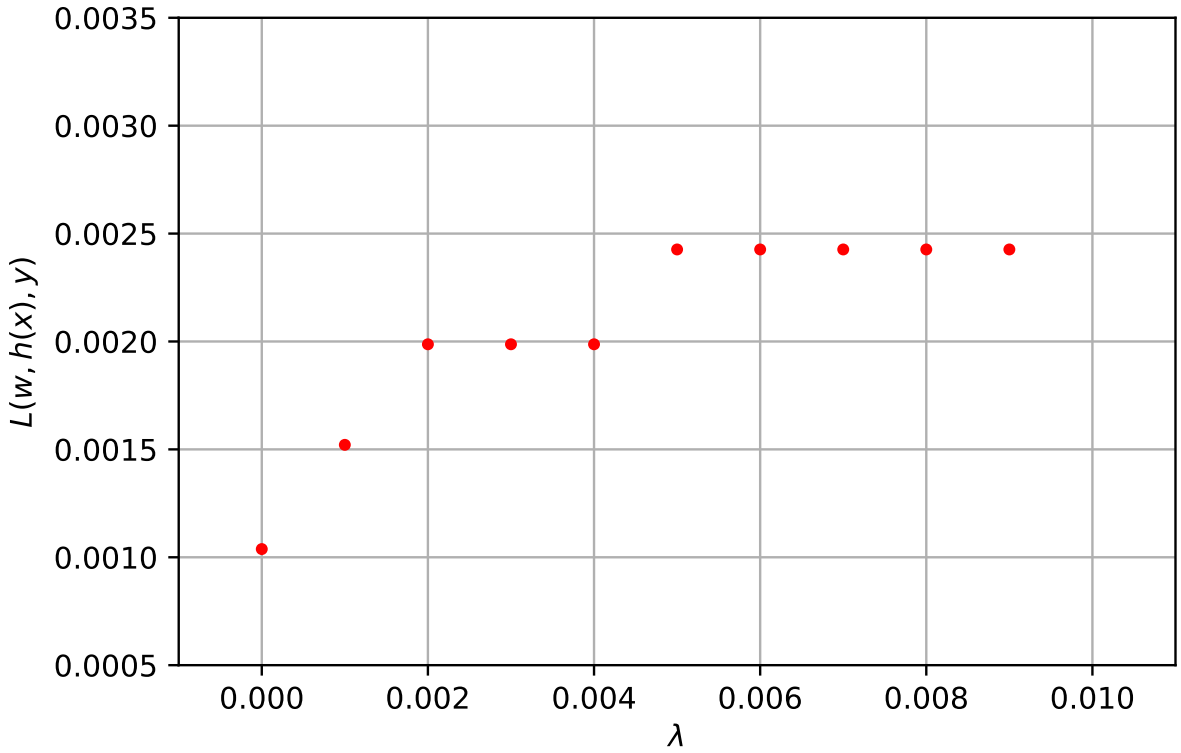
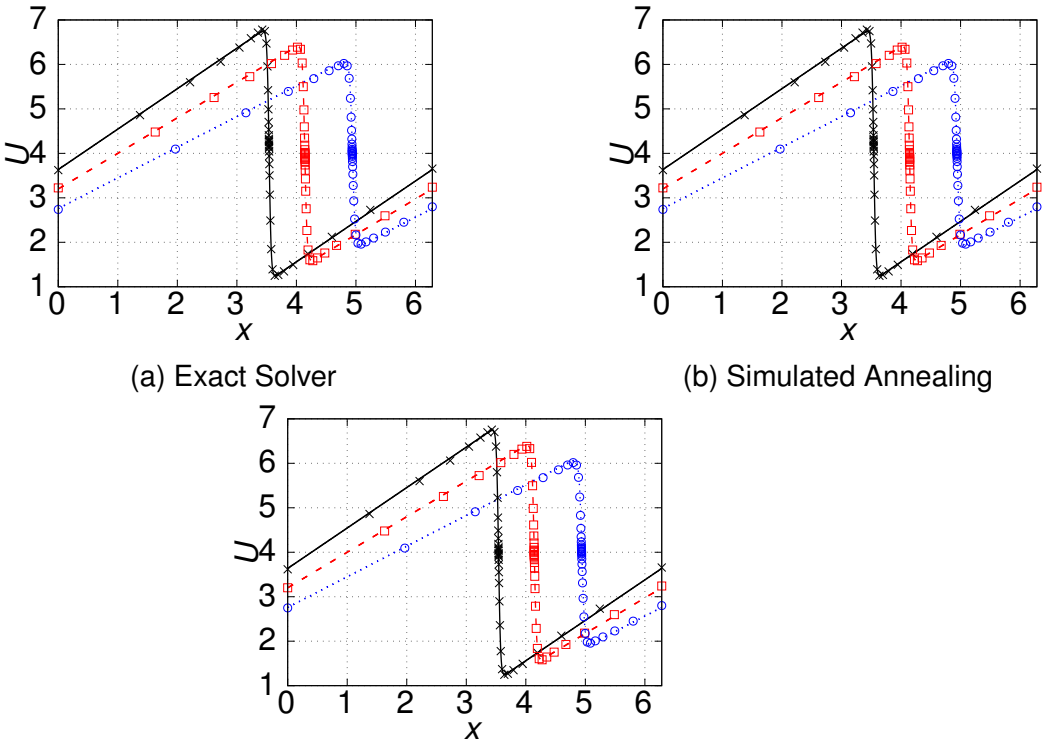


Figura 22 – Função Custo, $L(w, h(x), y)$, como função do parâmetro de regularização, λ.

A função custo, Equação 140, do nosso ensemble foi validada no conjunto de teste, como mostrado na Tabela 7, para diferentes números de qubits de precisão, R. Em todos os casos o ensemble performou melhor do que todos os *weak-learners*



(a) Exact Solver

(b) Simulated Annealing

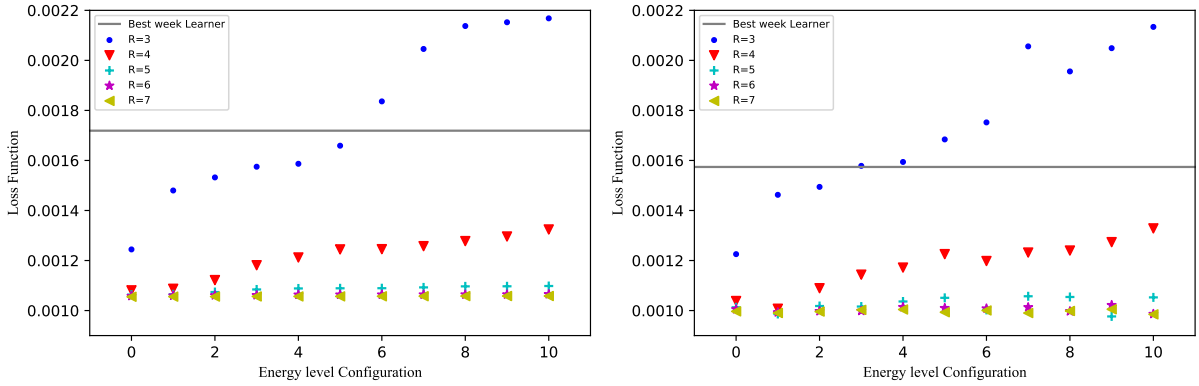
(c) Quantum Annealing

Figura 23 – Solução analítica $U_a(x)$ (linhas) e a solução do Qboost $U_{qb}(x)$ (marcadores) com $R = 6$ do conjunto de teste para a equação de Burgers em uma dimensão com viscosidade. Da esquerda para a direita, nós temos curvas para $t = 0,1$, $t = 0,25$ e $t = 0,45$, respectivamente. A solução são obtidas usando três funcionalidades diferentes do pacote Ocean da D-Wave: (a) Diagonalização exata do hamiltoniano (Equação 140) atráves do *Exact Solver*, (b) solução clássica através do *simulated annealing*, e (c) solução quântica através do sistema 2000Q.

utilizados para compô-lo, presentes na Figura 24 e Tabela 6. Para o *simulated annealing*, podemos notar que o valor alcança um mínimo de aproximadamente $0,0010$, que é alcançada com uma precisão de cinco qubits e não é reduzida com adição de novos qubits. Considerando o *quantum annealing* nós também observamos um valor mínimo, da função custo, em torno de $0,0010$, que ocorre para $R \geq 6$. Como comparação realizamos uma optimização padrão, nós utilizamos o gradiente descendente estocástico para minimizar a Equação 140. Após 10.000 rodadas em um laptop padrão rodando com precisão dupla, nós obtemos um valor médio de $0,0010200$ para $L(w, h(x), y)$ com uma variância de $1,46 \times 10^{-5}$. Nós notamos que este valor é pior que o melhor resultado produzido pela $2000Q$ QPU de acordo com a Tabela 7, $0,0009779$. Esse resultado garante a convergência mais rápida provida pelo *quantum annealing*. Vale ressaltar que diferentes rodadas do *quantum annealing* podem resultar em um pequeno *gap* no valor do resultado, desta forma os resultados do *quantum annealing* são equivalentes aos demais métodos.

Tabela 7 – Valores da função custo e do tempo decorrido para encontrar a solução para diferentes qubits de precisão (Equação 140) e diferentes métodos. O parâmetro R é a precisão da expansão do ponto flutuante, enquanto os métodos usados são funções especiais do pacote Ocean da D-Wave, provendo a diagonalização exata através do *Exact Solver*, o annealing clássico através do *simulated annealing*, e o **quantum annealing** rodado no 2000Q QPU. O tempo de annealing é o mesmo para todos os níveis de precisão nas soluções do *quantum annealing*.

	Exact Solver		*Simulated Annealing*		*Quantum Annealing*
Precisão (R)	Função Custo	Tempo (s)	Função Custo	Tempo (s)	Função Custo
3	0.0012847	0.002001	0.0012847	1111.0742	0.0012847
4	0.0010182	0.016004	0.0010543	1654.2170	0.0010543
5	0.0010131	0.124011	0.0010131	2631.2131	0.0009863
6	0.0010052	0.968069	0.0009856	4314.6367	0.0009927
7	**0.0009967**	10.027727	0.0010084	4463.0229	0.0010227
8	-	-	0.0010191	5490.6962	0.0010250
9	-	-	0.0009908	6807.2573	**0.0009779**
10	-	-	0.0010528	9473.4266	0.0010079
11	-	-	0.0010324	11167.2495	0.0010291
12	-	-	**0.0009820**	13716.8256	0.0009871
13	-	-	0.0010566	14157.8434	0.0010012
14	-	-	0.0009937	16095.1590	0.0010071



(a) Training set. (b) Test set.

Figura 24 – Valores da função custo obtidas pelo método QBoost dos primeiros onze níveis de energia do hamiltoniano final (Equação 140) levando em consideração diferentes números de qubit de precisão. O número 0 representa o nível de energia relacionado ao estado fundamental do hamiltoniano final. (a) e (b) se referem ao conjunto de treinamento e teste, respectivamente. Em ambos os gráficos, todos os níveis de energia para precisões $R > 3$ possuem uma solução associada que é melhor do que a fornecida pelo melhor *weak-learner* que compõem o ensemble, como mostrado pela linha horizontal.

A dependência da função custo para diferentes números de qubits de precisão $R = \{3, 4, ..., 7\}$ é mostrada na Figura 24, obtida pelo método Qboost para (a) o conjunto de treinamento e (b) o conjunto de teste nos primeiros onze níveis de energia do hamiltoniano final (Equação 140). Para fins de comparação, a função custo do melhor *weak-learner* também é mostrado (como a linha horizontal na Figura 24). Como resultado, o ensemble com $R > 4$ performou melhor que o melhor *weak-learner* para todos os níveis de energia considerados. Nós também observamos que, conforme aumentamos o número de qubits de precisão, o valor da função custo se aproximam umas das outras para diferentes níveis de energia do hamiltoniano final. Desta forma, podemos considerar que, para precisões $R > 5$, até mesmo soluções que não alcançaram o estado fundamental do hamiltoniano final podem ser consideradas válidas, já que os erros relacionados a estes estados são muito próximos, com uma tolerância pré-definida de $0,00005$.

A simulação utilizando o *Exact solver* possui uma limitação em relação ao número de qubits de precisão devido ao aumento exponencial da memória requerida para execução. Nós fomos capazes de rodar apenas até a precisão $R = 7$ em nosso computador pessoal com 32 GB RAM. Vale ressaltar que, conforme aumentamos R, o tempo de execução aumenta aproximadamente de uma ordem de magnitude. O tempo de computação para o *simulated annelaing* cresce suavemente. Porém, ele começa em um valor seis ordens de magnitude maior que o tempo gasto com o *Exact solver*.

Embora tenhamos obtido bons resultados mesmo para um número pequeno de qubits de precisão, nós listamos as seguintes ressalvas:

- Nós estamos minimizando a função custo (Equação 140), que consiste em encontrar os valores para os pesos w_k sobre o domínio real para uma função quadrática. Isso é um problema de mínimos quadrados, que pode ser resolvido eficientemente por um computador clássico (BOYD; VANDENBERGHE, 2004). Entretanto, mesmo neste caso é justo procurar por um *speedup* polinomial através do *quantum annealing*.

- As variáveis contínuas foram expressas utilizando a expansão em ponto flutuante (ROGERS; SINGLETON, 2020). Essa metologia resulta em um estreitamento exponencial dos níveis de energia do Hamiltoniano final conforme o número de qubits de precisão aumenta. Nós observamos esse comportamento da Equação 138 da seguinte maneira. Como as variáveis $Q_r^k \in \{0, 1\}$ e as pequenas contribuições da equação provém da maior potência de r em 2^{-r}, nós observamos que a diferença entre níveis de energia adjacentes sempre será multiplicados por um fator que decresce exponencialmente com a precisão das variáveis continuas, Figura 24. Apesar desta propriedade não desejada para a computação quântica baseada na evolução adiabática, em nosso exemplo isso parece não interferir

significativamente para a solução do problema, como pode ser vista na Tabela 7. Além do que, como estamos procurando por pesos w_k para compor o *ensemble* de *weak-learners*, a precisão de w_k pode ser compensada pela adição de mais *weak-learners*.

É importante mencionar que uma comparação entre o Qboost para problemas de regressão e a formulação clássica padrão aplicados para a solução da Equação de Burgers, dado o corrente estágio de desenvolvimento dos *quantum annealers*, não é muito justa. Por exemplo, com um algoritmo de diferença finita aplicada a Equação de Burgers 1D nós obtemos um erro quadrático médio de $1,51 \times 10^{-8}$ para o mesmo conjunto de teste, $t \in \{0.10, 0.25, 0.45\}$. Apesar de conseguir um resultado que é cinco ordens de magnitude melhor, a maior deficiência dos *quantum annealers* é o alto nível de ruído (ZABORNIAK; SOUSA, 2021) e o número limitado de qubits (LU; BRAUNSTEIN, Samuel L., 2013), que impactam a precisão das soluções quânticas. Enquanto um laptop rodando com previsão dupla utiliza 64 bits, o número máximo de qubits de precisão em nosso exemplo é 14. Por essa razão, nós estamos restritos a prova de conceito do nosso método.

4.5 CONCLUSÃO

Uma adaptação do algoritmo *QBoost* para solucionar problemas de regressão utilizando a aproximação de ponto-flutuante para representar variáveis reais foi aplicada com sucesso na solução da equação de Burgers unidimensional com viscosidade. Nosso *framework* não resulta em *overfitting* ou *underfitting* e possui um bom acordo com a solução analítica (veja Figura 23).

A aproximação de ponto-flutuante deve ser utilizada com cuidado, já que o gap de energia decresce rapidamente com o aumento da precisão da aproximação, entretanto as soluções aceitas estão muito próximas do estado fundamental, e desta maneira, é uma boa aproximação à solução ótima. Outro problema em potencial tem relação com a conectividade entre os qubits, que pode limitar o número de *weak-learners* utilizados no ensemble.

5 CONCLUSÃO.

Nesta tese abordamos dois problemas distintos no campo do Machine Learning Quântico. O primeiro na área CQ (execução em um computador clássico de um conjunto de dados quântico) onde desenvolvemos um classificador de emaranhamento para um sistema de dois qutrits. O segundo na área QC (execução em um computador quântico de um conjunto de dados clássico) onde desenvolvemos um regressor da equação de Burgers unidimensional com viscosidade baseado em ensemble learning.

No segundo capítulo expomos fundamentos teóricos do campo de ML, apresentando o funcionamento dos modelos através de um problema simples de classificação de frutas no primeiro Capítulo.

No terceiro Capítulo, nós criamos uma metodologia para a criação de estados emaranhados presos uniformemente distribuídos pelo espaço de Hilbert e aplicamos diversas técnica de ML e autoML para criar um classificador de emaranhamento em um sistema de dois qutrits. Estudamos dois sistema distintos, o primeiro referente a uma família de estados de um parâmetro de dois qutrits, chamada de estados Horodecki e a segunda estados de dois qutrits uniformemente distribuídos no espaço de Hilbert. Para os estados de Horodecki nossa rede treinada foi capaz de obter uma classificação de aproximadamente 100% para todas as classes. Como principais resultados podemos destacar:

1. Conseguimos aplicar nosso *framework* com sucesso até mesmo na instância mais difícil do problema, quando o critério de Perez-Horodecki não consegue identificar se um estado quântico é emaranhado ou separável.

    - Para o problema de classificação binária (SEP *vs*. PPTES), o TPOT obteve uma acurácia de $74,1\%$ na classificação de estados SEP e $76,6\%$ na classificação de estados PPTES, com uma performance geral de $75,3\%$.
    - Para o problema de classificação múltipla (SEP *vs*. PPTES *vs*. NPT), o TPOT classificou com acurácia $71,4\%$ para os SEP, $62,4\%$ para os PPTES e $88,6\%$ para os NPT e uma performance geral de $73,8\%$.
    - Para o problema de regressão da métrica de robustez generalizada com um limiar conservativo de confiança de $GR_{ML} > 0,1$, encontramos um erro absoluto de $0,0335$.

2. Para alcançar modelos com uma performance maior em qualquer dos testes apresentados, devemos aumentar o número de exemplos dentro do conjunto de dados.

3. Utilizamos em nosso conjunto de dados tanto estados aleatórios como estados artificiais, e obtemos uma evidência numérica forte que utilizar estados artificiais

é uma boa estratégia para contornar o problema de baixa probabilidade de sorteio de estados PPTES utilizando a medida de Hilbert-Schimdt.

4. Podemos concluir que, dado a dificuldade do problema, nós temos forte indicações que o autoML pode ser benéfico para tarefas de aprendizado supervisionado de todos os tamanhos e escalas, incluindo problemas com conjunto de dados pequenos, pavimentando o caminho para futuras investigações utilizando este *framework*.

No quarto Capítulo, aplicamos com sucesso uma adaptação do algoritmo QBoost para solucionar problemas de regressão utilizando a aproximação de ponto-flutuante para representar variáveis reais na solução da equação de Burgers unidimensional com viscosidade. Podemos destacar como principais resultados:

1. Nosso *framework* não resultou em *overfitting* ou *underfitting* e possui um bom acordo com a solução analítica.

2. A aproximação de ponto-flutuante deve ser utilizada com cuidado, já que o gap de energia decresce rapidamente com o aumento da precisão da aproximação. Entretanto, as soluções aceitas estão muito próximas do estado fundamental, e desta maneira, é uma boa aproximação à solução ótima.

3. Nos atuais hardwares existe uma limitação física em relação a conectividade entre os qubits, que pode limitar o número de *weak-learners* utilizados no ensemble.

4. Com o desenvolvimento atual dos hardwares que implementam QA a utilização da expansão em ponto flutuante só possui efetividade se aplicada em problemas que não exijam alta precisão. Isso se deve a grande limitação de conectividade entre os qubits. Uma alternativa para tal situação seria a criação de hardwares especializados na solução de determinados problemas em que a conectividade seria maximizada.

REFERÊNCIAS

ACIÉN, A.; ANDRIANOV, A.; COSTA, L.; JANÉ, E.; LATORRE, J. I.; TARRACH, R. Generalized Schmidt Decomposition and Classification of Three-Quantum-Bit States. **Physical Review Letters**, American Physical Society (APS), v. 85, n. 7, p. 1560–1563, ago. 2000. DOI: 10.1103/physrevlett.85.1560. Disponível em: https://doi.org/10.1103/physrevlett.85.1560.

AHARONOV, Dorit; DAM, Wim van; KEMPE, Julia; LANDAU, Zeph; LLOYD, Seth; REGEV, Oded. Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation, 2004. eprint: arXiv:quant-ph/0405098.

ALTMAN, N. S. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. **The American Statistician**, Informa UK Limited, v. 46, n. 3, p. 175–185, ago. 1992. DOI: 10.1080/00031305.1992.10475879. Disponível em: https://doi.org/10.1080/00031305.1992.10475879.

AMIN, M. H. S.; CHOI, V. First-order quantum phase transition in adiabatic quantum computation. **Physical Review A**, American Physical Society (APS), v. 80, n. 6, dez. 2009. DOI: 10.1103/physreva.80.062326. Disponível em: https://doi.org/10.1103%2Fphysreva.80.062326.

ARORA, Jasbir. **Introduction to Optimum Design**. 2. ed. [*S.l.*]: Academic Press, mai. 2014.

AUBRUN, Guillaume; SZAREK, Stanisław J. Tensor products of convex sets and the volume of separable states on *N* qudits. **Phys. Rev. A**, American Physical Society, v. 73, p. 022109, 2 fev. 2006. DOI: 10.1103/PhysRevA.73.022109. Disponível em: https://link.aps.org/doi/10.1103/PhysRevA.73.022109.

BAKATOR, Mihalj; RADOSAV, Dragica. Deep Learning and Medical Diagnosis: A Review of Literature. **Multimodal Technologies and Interaction**, MDPI AG, v. 2, n. 3, p. 47, ago. 2018. DOI: 10.3390/mti2030047. Disponível em: https://doi.org/10.3390/mti2030047.

BALL, NICHOLAS M.; BRUNNER, ROBERT J. DATA MINING AND MACHINE LEARNING IN ASTRONOMY. **International Journal of Modern Physics D**, World Scientific Pub Co Pte Lt, v. 19, n. 07, p. 1049–1106, jul. 2010. DOI:

10.1142/s0218271810017160. Disponível em:
https://doi.org/10.1142/s0218271810017160.

BENEDETTI, Marcello; LLOYD, Erika; SACK, Stefan; FIORENTINI, Mattia. Parameterized quantum circuits as machine learning models. **Quantum Science and Technology**, IOP Publishing, v. 4, n. 4, p. 043001, 2019.

BENNETT, Charles H.; DIVINCENZO, David P.; SMOLIN, John A.; WOOTTERS, William K. Mixed-state entanglement and quantum error correction. **Physical Review A**, American Physical Society (APS), v. 54, n. 5, p. 3824–3851, nov. 1996. DOI: 10.1103/physreva.54.3824. Disponível em: https://doi.org/10.1103/physreva.54.3824.

BORN, M.; FOCK, V. Beweis des Adiabatensatzes. **Zeitschrift fur Physik**, v. 51, n. 3-4, p. 165–180, mar. 1928. DOI: 10.1007/BF01343193.

BOSE, Indranil; MAHAPATRA, Radha K. Business data mining — a machine learning perspective. **Information & Management**, Elsevier BV, v. 39, n. 3, p. 211–225, dez. 2001. DOI: 10.1016/s0378-7206(01)00091-x. Disponível em: https://doi.org/10.1016/s0378-7206(01)00091-x.

BOURENNANE, Mohamed *et al.* Experimental Detection of Multipartite Entanglement using Witness Operators. **Phys. Rev. Lett.**, American Physical Society, v. 92, p. 087902, 8 fev. 2004. DOI: 10.1103/PhysRevLett.92.087902. Disponível em: https://link.aps.org/doi/10.1103/PhysRevLett.92.087902.

BOYD, Stephen; VANDENBERGHE, Lieven. **Convex Optimization**. [*S.l.*]: Cambridge University Press, mar. 2004. DOI: 10.1017/cbo9780511804441. Disponível em: https://doi.org/10.1017/cbo9780511804441.

BRANDÃO, Fernando G. S. L. Quantifying entanglement with witness operators. **Phys. Rev. A**, American Physical Society, v. 72, p. 022310, 2 ago. 2005. DOI: 10.1103/PhysRevA.72.022310. Disponível em: https://link.aps.org/doi/10.1103/PhysRevA.72.022310.

BRANDÃO, Fernando G. S. L.; VIANNA, Reinaldo O. Robust semidefinite programming approach to the separability problem. **Physical Review A - Atomic, Molecular, and Optical Physics**, v. 70, n. 6, 2004. ISSN 10502947. DOI: 10.1103/PhysRevA.70.062309. arXiv: 0405008 [quant-ph].

BRAUNSTEIN, Samuel L. Geometry of quantum inference. **Physics Letters A**, Elsevier, v. 219, n. 3-4, p. 169–174, 1996.

BREIMAN, Leo. Bagging predictors. **Machine Learning**, Springer Science e Business Media LLC, v. 24, n. 2, p. 123–140, ago. 1996. DOI: 10.1007/bf00058655. Disponível em: https://doi.org/10.1007/bf00058655.

BROYDEN, C. G.; DENNIS, J. E.; MORÉ, JORGE J. On the Local and Superlinear Convergence of Quasi-Newton Methods. **IMA Journal of Applied Mathematics**, Oxford University Press (OUP), v. 12, n. 3, p. 223–245, 1973. DOI: 10.1093/imamat/12.3.223. Disponível em: https://doi.org/10.1093/imamat/12.3.223.

BURGERS, J.M. A Mathematical Model Illustrating the Theory of Turbulence. *In*: VON MISES, Richard; VON KÁRMÁN, Theodore (Ed.). [*S.l.*]: Elsevier, 1948. v. 1. (Advances in Applied Mechanics). P. 171–199. DOI: https://doi.org/10.1016/S0065-2156(08)70100-5. Disponível em: https://www.sciencedirect.com/science/article/pii/S0065215608701005.

CHEN, S.; BILLINGS, S. A. Neural networks for nonlinear dynamic system modelling and identification. **International Journal of Control**, Informa UK Limited, v. 56, n. 2, p. 319–346, ago. 1992. DOI: 10.1080/00207179208934317. Disponível em: https://doi.org/10.1080/00207179208934317.

CHEN, S.; BILLINGS, S. A.; GRANT, P. M. Non-linear system identification using neural networks. **International Journal of Control**, Informa UK Limited, v. 51, n. 6, p. 1191–1214, jan. 1990. DOI: 10.1080/00207179008934126. Disponível em: https://doi.org/10.1080/00207179008934126.

CILIBERTO, Carlo; ROCCHETTO, Andrea; RUDI, Alessandro; WOSSNIG, Leonard. Statistical Limits of Supervised Quantum Learning. arXiv, 2020. DOI: 10.48550/ARXIV.2001.10477. Disponível em: https://arxiv.org/abs/2001.10477.

CLARISSE, Lieven. Construction of bound entangled edge states with special ranks. **Physics Letters A**, Elsevier BV, v. 359, n. 6, p. 603–607, dez. 2006. DOI: 10.1016/j.physleta.2006.07.045. Disponível em: https://doi.org/10.1016/j.physleta.2006.07.045.

CORTES, Corinna; VAPNIK, Vladimir. Support-vector networks. **Machine Learning**, Springer Science e Business Media LLC, v. 20, n. 3, p. 273–297, set. 1995. DOI: 10.1007/bf00994018. Disponível em: https://doi.org/10.1007/bf00994018.

CRUZ-SANTOS, William; MORALES-LUNA, Guillermo. **Approximability of Optimization Problems through Adiabatic Quantum Computation**. [*S.l.*]: Morgan & Claypool Publishers, 2014. (Synthesis Lectures on Quantum Computing). DOI: 10.2200/S00596ED1V01Y201409QMC009. Disponível em: https://doi.org/10.2200/S00596ED1V01Y201409QMC009.

CULLIMORE, M; EVERITT, M J; ORMEROD, M A; SAMSON, J H; WILSON, R D; ZAGOSKIN, A M. Relationship between minimum gap and success probability in adiabatic quantum computing. **Journal of Physics A: Mathematical and Theoretical**, IOP Publishing, v. 45, n. 50, p. 505305, nov. 2012. DOI: 10.1088/1751-8113/45/50/505305. Disponível em: https://doi.org/10.1088%2F1751-8113%2F45%2F50%2F505305.

DAM, W. van; MOSCA, M.; VAZIRANI, U. How powerful is adiabatic quantum computation? *In*: PROCEEDINGS 42nd IEEE Symposium on Foundations of Computer Science. [*S.l.*: *s.n.*], 2001. P. 279–287. DOI: 10.1109/SFCS.2001.959902.

DAS, Arnab; CHAKRABARTI, Bikas K. Colloquium: Quantum annealing and analog quantum computation. **Rev. Mod. Phys.**, American Physical Society, v. 80, p. 1061–1081, 3 set. 2008. DOI: 10.1103/RevModPhys.80.1061. Disponível em: https://link.aps.org/doi/10.1103/RevModPhys.80.1061.

DAS, Sreetama; CHANDA, Titas; LEWENSTEIN, Maciej; SANPERA, Anna; DE, Aditi sen; SEN, Ujjwal. The separability versus entanglement problem, jan. 2017.

DEUTSCH, David. Quantum theory, the Church–Turing principle and the universal quantum computer. **Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences**, v. 400, p. 117–97, 1985.

DICKSON, Neil G.; AMIN, M. H. S. Does Adiabatic Quantum Optimization Fail for NP-Complete Problems? **Physical Review Letters**, American Physical Society (APS), v. 106, n. 5, fev. 2011. DOI: 10.1103/physrevlett.106.050502. Disponível em: https://doi.org/10.1103%2Fphysrevlett.106.050502.

DIGALAKIS, V.; ROHLICEK, J.R.; OSTENDORF, M. ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition. **IEEE Transactions on Speech and Audio Processing**, Institute of Electrical e Electronics Engineers (IEEE), v. 1, n. 4, p. 431–442, 1993. DOI: 10.1109/89.242489. Disponível em: https://doi.org/10.1109/89.242489.

DUNJKO, Vedran; BRIEGEL, Hans J. Machine learning & artificial intelligence in the quantum domain: a review of recent progress. **Reports on Progress in Physics**, IOP Publishing, v. 81, n. 7, p. 074001, jun. 2018. DOI: 10.1088/1361-6633/aab406. Disponível em: https://doi.org/10.1088/1361-6633/aab406.

FANG, W.-C.; SHEU, B.J.; CHEN, O.T.-C.; CHOI, J. A VLSI neural processor for image data compression using self-organization networks. **IEEE Transactions on Neural Networks**, Institute of Electrical e Electronics Engineers (IEEE), v. 3, n. 3, p. 506–518, mai. 1992. DOI: 10.1109/72.129423. Disponível em: https://doi.org/10.1109/72.129423.

FARHI, Edward; GOLDSTONE, Jeffrey; GUTMANN, Sam; LAPAN, Joshua; LUNDGREN, Andrew; PREDA, Daniel. A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem. **Science**, v. 292, n. 5516, p. 472–475, 2001. DOI: 10.1126/science.1057726. eprint: https://www.science.org/doi/pdf/10.1126/science.1057726. Disponível em: https://www.science.org/doi/abs/10.1126/science.1057726.

FARHI, Edward; GOLDSTONE, Jeffrey; GUTMANN, Sam; NAGAJ, Daniel. How to make the quantum adiabatic algorithm fail. **International Journal of Quantum Information**, World Scientific, v. 6, n. 03, p. 503–516, 2008.

FEURER, Matthias; KLEIN, Aaron; EGGENSPERGER, Katharina; SPRINGENBERG, Jost; BLUM, Manuel; HUTTER, Frank. Efficient and Robust Automated Machine Learning. *In*: CORTES, C.; LAWRENCE, N.; LEE, D.; SUGIYAMA, M.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2015a. Disponível em: https://proceedings.neurips.cc/paper/2015/file/11d0e6287202fced83f79975ec59a3a6-Paper.pdf.

FEURER, Matthias; KLEIN, Aaron; EGGENSPERGER, Katharina; SPRINGENBERG, Jost; BLUM, Manuel; HUTTER, Frank. Efficient and Robust Automated Machine Learning. *In*: CORTES, C.; LAWRENCE, N. D.; LEE, D. D.;

SUGIYAMA, M.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 28**. [*S.l.*]: Curran Associates, Inc., 2015b. P. 2962–2970. Disponível em: http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf.

FUNAHASHI, Ken-Ichi. On the approximate realization of continuous mappings by neural networks. **Neural Networks**, Elsevier BV, v. 2, n. 3, p. 183–192, jan. 1989. DOI: 10.1016/0893-6080(89)90003-8. Disponível em: https://doi.org/10.1016/0893-6080(89)90003-8.

FURUSAWA, A. Unconditional Quantum Teleportation. **Science**, American Association for the Advancement of Science (AAAS), v. 282, n. 5389, p. 706–709, out. 1998. DOI: 10.1126/science.282.5389.706. Disponível em: https://doi.org/10.1126/science.282.5389.706.

GELENBE, E.; KOUBI, V.; PEKERGIN, F. Dynamical random neural network approach to the traveling salesman problem. *In*: PROCEEDINGS of IEEE Systems Man and Cybernetics Conference - SMC. [*S.l.*]: IEEE. DOI: 10.1109/icsmc.1993.384945. Disponível em: https://doi.org/10.1109/icsmc.1993.384945.

GEORGE, D. Magoulas and Andriana Prentza. **Machine Learning in Medical Applications, Proceeding machine learning and its applications: Advance lectures**, p. 300–307, 2001.

GLOVER, Fred; KOCHENBERGER, Gary; DU, Yu. Quantum Bridge Analytics I: a tutorial on formulating and using QUBO models. **4OR**, Springer Science e Business Media LLC, v. 17, n. 4, p. 335–371, nov. 2019. DOI: 10.1007/s10288-019-00424-y. Disponível em: https://doi.org/10.1007/s10288-019-00424-y.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. [*S.l.*]: MIT Press, 2016. http://www.deeplearningbook.org.

GRÖBLACHER, Simon; JENNEWEIN, Thomas; VAZIRI, Alipasha; WEIHS, Gregor; ZEILINGER, Anton. Experimental quantum cryptography with qutrits. **New Journal of Physics**, IOP Publishing, v. 8, n. 5, p. 75–75, mai. 2006. DOI: 10.1088/1367-2630/8/5/075. Disponível em: https://doi.org/10.1088/1367-2630/8/5/075.

GÜHNE, Otfried; TÓTH, Géza. Entanglement detection. **Physics Reports**, Elsevier BV, v. 474, n. 1-6, p. 1–75, abr. 2009. DOI: 10.1016/j.physrep.2009.02.004. Disponível em: https://doi.org/10.1016/j.physrep.2009.02.004.

GURVITS, Leonid. Classical complexity and quantum entanglement. **Journal of Computer and System Sciences**, Elsevier BV, v. 69, n. 3, p. 448–484, nov. 2004. DOI: 10.1016/j.jcss.2004.06.003. Disponível em: https://doi.org/10.1016/j.jcss.2004.06.003.

HALDER, Saronath; BANIK, Manik; GHOSH, Sibasish. Family of bound entangled states on the boundary of the Peres set. **Physical Review A**, American Physical Society (APS), v. 99, n. 6, jun. 2019. DOI: 10.1103/physreva.99.062329. Disponível em: https://doi.org/10.1103/physreva.99.062329.

HARROW, Aram; HAYDEN, Patrick; LEUNG, Debbie. Superdense Coding of Quantum States. **Physical Review Letters**, American Physical Society (APS), v. 92, n. 18, mai. 2004. DOI: 10.1103/physrevlett.92.187901. Disponível em: https://doi.org/10.1103/physrevlett.92.187901.

HASTIE, Trevor; TIBSHIRANI, Robert; FRIEDMAN, Jerome. **The elements of statistical learning**. 2. ed. New York, NY: Springer, fev. 2009. (Springer series in statistics).

HE, Xin; ZHAO, Kaiyong; CHU, Xiaowen. AutoML: A survey of the state-of-the-art. **Knowledge-Based Systems**, Elsevier BV, v. 212, p. 106622, jan. 2021. DOI: 10.1016/j.knosys.2020.106622. Disponível em: https://doi.org/10.1016%2Fj.knosys.2020.106622.

HEBB, D. O. Organization of behavior. New York: Wiley, 1949, pp. 335, 4.00. **Journal of Clinical Psychology**, Wiley, v. 6, n. 3, p. 307–307, jul. 1950. DOI: 10.1002/1097-4679(195007)6:3<307::aid-jclp2270060338>3.0.co;2-k. Disponível em: https://doi.org/10.1002/1097-4679(195007)6:3%3C307::aid-jclp2270060338%3E3.0.co;2-k.

HIYAMA, T.; KOUZUMA, S.; IMAKUBO, T.; ORTMEYER, T.H. Evaluation of neural network based real time maximum power tracking controller for PV system. **IEEE Transactions on Energy Conversion**, Institute of Electrical e Electronics Engineers (IEEE), v. 10, n. 3, p. 543–548, 1995. DOI: 10.1109/60.464880. Disponível em: https://doi.org/10.1109/60.464880.

HORMOZI, Layla; BROWN, Ethan W.; CARLEO, Giuseppe; TROYER, Matthias. Nonstoquastic Hamiltonians and quantum annealing of an Ising spin glass. **Physical Review B**, American Physical Society (APS), v. 95, n. 18, mai. 2017. DOI: 10.1103/physrevb.95.184416. Disponível em: https://doi.org/10.1103/physrevb.95.184416.

HORODECKI, Michał; HORODECKI, Paweł; HORODECKI, Ryszard. Mixed-State Entanglement and Distillation: Is there a “Bound” Entanglement in Nature? **Physical Review Letters**, American Physical Society (APS), v. 80, n. 24, p. 5239–5242, jun. 1998. DOI: 10.1103/physrevlett.80.5239. Disponível em: https://doi.org/10.1103/physrevlett.80.5239.

HORODECKI, Michał; HORODECKI, Paweł; HORODECKI, Ryszard. Separability of mixed states: necessary and sufficient conditions. **Physics Letters A**, Elsevier BV, v. 223, n. 1-2, p. 1–8, nov. 1996. DOI: 10.1016/s0375-9601(96)00706-2. Disponível em: https://doi.org/10.1016/s0375-9601(96)00706-2.

HORODECKI, Paweł; HORODECKI, Michał; HORODECKI, Ryszard. Bound Entanglement Can Be Activated. **Physical Review Letters**, American Physical Society (APS), v. 82, n. 5, p. 1056–1059, fev. 1999. DOI: 10.1103/physrevlett.82.1056. Disponível em: https://doi.org/10.1103/physrevlett.82.1056.

HUBER, Marcus; LAMI, Ludovico; LANCIEN, Cécilia; MÜLLER-HERMES, Alexander. High-Dimensional Entanglement in States with Positive Partial Transposition. **Physical Review Letters**, American Physical Society (APS), v. 121, n. 20, nov. 2018. DOI: 10.1103/physrevlett.121.200503. Disponível em: https://doi.org/10.1103/physrevlett.121.200503.

HUTTER, F.; HOOS, H. H.; LEYTON-BROWN, K. Sequential Model-Based Optimization for General Algorithm Configuration. *In*: PROC. OF LION-5. [*S.l.*: *s.n.*], 2011. P. 507–523.

INC, D-Wave Systems. **QPU-Specific Physical Properties: DW_2000Q_6**. [*S.l.*: *s.n.*], 2019. eprint: D-WaveUserManual09-1215A-C.

JANSEN, Sabine; RUSKAI, Mary-Beth; SEILER, Ruedi. Bounds for the adiabatic approximation with applications to quantum computation. **Journal of Mathematical Physics**, American Institute of Physics, v. 48, n. 10, p. 102111, 2007.

JARVIS, J. F. A Method for Automating the Visual Inspection of Printed Wiring Boards. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Institute of Electrical e Electronics Engineers (IEEE), PAMI-2, n. 1, p. 77–82, jan. 1980. DOI: 10.1109/tpami.1980.4766975. Disponível em: https://doi.org/10.1109/tpami.1980.4766975.

JIN, Haifeng; SONG, Qingquan; HU, Xia. Auto-Keras: An Efficient Neural Architecture Search System. *In*: ACM. PROCEEDINGS of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. [*S.l.*: *s.n.*], 2019. P. 1946–1956.

JOZSA, Richard; LINDEN, Noah. On the role of entanglement in quantum-computational speed-up. **Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences**, The Royal Society, v. 459, n. 2036, p. 2011–2032, ago. 2003. DOI: 10.1098/rspa.2002.1097. Disponível em: https://doi.org/10.1098/rspa.2002.1097.

KINGMA, Diederik P.; BA, Jimmy. **Adam: A Method for Stochastic Optimization**. [*S.l.*: *s.n.*], 2014. eprint: arXiv:1412.6980.

KOHONEN, Teuvo. **Self-Organization and Associative Memory**. [*S.l.*]: Springer Berlin Heidelberg, 1989. DOI: 10.1007/978-3-642-88163-3. Disponível em: https://doi.org/10.1007/978-3-642-88163-3.

KOHONEN, Teuvo. Self-organized formation of topologically correct feature maps. **Biological Cybernetics**, Springer Nature, v. 43, n. 1, p. 59–69, 1982. DOI: 10.1007/bf00337288. Disponível em: https://doi.org/10.1007/bf00337288.

KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. **Communications of the ACM**, AcM New York, NY, USA, v. 60, n. 6, p. 84–90, 2017.

LAX, Peter. **Functional analysis**. New York: Wiley, 2002. ISBN 978-0-471-55604-6.

LE, TT; FU, W; MOORE, JH. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. **Bioinformatics (Oxford, England)**, 2019.

LEVENBERG, Kenneth. A method for the solution of certain non-linear problems in least squares. **Quarterly of Applied Mathematics**, American Mathematical Society

(AMS), v. 2, n. 2, p. 164–168, jul. 1944. DOI: 10.1090/qam/10666. Disponível em: https://doi.org/10.1090/qam/10666.

LEWENSTEIN, M.; KRAUS, B.; CIRAC, J. I.; HORODECKI, P. Optimization of entanglement witnesses. **Physical Review A**, American Physical Society (APS), v. 62, n. 5, out. 2000. DOI: 10.1103/physreva.62.052310. Disponível em: https://doi.org/10.1103/physreva.62.052310.

LEWENSTEIN, M.; KRAUS, B.; HORODECKI, P.; CIRAC, J. I. Characterization of separable states and entanglement witnesses. **Physical Review A**, American Physical Society (APS), v. 63, n. 4, mar. 2001. DOI: 10.1103/physreva.63.044304. Disponível em: https://doi.org/10.1103/physreva.63.044304.

LIEW, Venus. An overview on various ways of bootstrap methods. **MPRA Paper**, jan. 2008.

LLOYD, Seth. Universal quantum simulators. **Science**, American Association for the Advancement of Science, v. 273, n. 5278, p. 1073–1078, 1996.

LÖFBERG, J. YALMIP : A Toolbox for Modeling and Optimization in MATLAB. *In*: PROCEEDINGS of the CACSD Conference. Taipei, Taiwan: [*s.n.*], 2004.

LOW, Yucheng; BICKSON, Danny; GONZALEZ, Joseph; GUESTRIN, Carlos; KYROLA, Aapo; HELLERSTEIN, Joseph M. Distributed GraphLab. **Proceedings of the VLDB Endowment**, VLDB Endowment, v. 5, n. 8, p. 716–727, abr. 2012. DOI: 10.14778/2212351.2212354. Disponível em: https://doi.org/10.14778/2212351.2212354.

LU, Songfeng; BRAUNSTEIN, Samuel L. Quantum decision tree classifier. **Quantum Information Processing**, Springer Science e Business Media LLC, v. 13, n. 3, p. 757–770, nov. 2013. DOI: 10.1007/s11128-013-0687-5. Disponível em: https://doi.org/10.1007/s11128-013-0687-5.

MARKOV, Igor L. Limits on fundamental limits to computation. **Nature**, Nature Publishing Group, v. 512, n. 7513, p. 147–154, 2014.

MARQUARDT, Donald W. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. **Journal of the Society for Industrial and Applied Mathematics**,

Society for Industrial & Applied Mathematics (SIAM), v. 11, n. 2, p. 431–441, jun. 1963. DOI: 10.1137/0111030. Disponível em: https://doi.org/10.1137/0111030.

MASUTTI, Thiago A.S.; CASTRO, Leandro N. de. A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem. **Information Sciences**, Elsevier BV, v. 179, n. 10, p. 1454–1468, abr. 2009. DOI: 10.1016/j.ins.2008.12.016. Disponível em: https://doi.org/10.1016/j.ins.2008.12.016.

MATLAB. **version 7.10.0 (R2010a)**. Natick, Massachusetts: The MathWorks Inc., 2010.

MATLAB. **version 9.6 (R2019b)**. Natick, Massachusetts: The MathWorks Inc., 2019.

MAY, Robert J; MAIER, Holger R; DANDY, Graeme C. Data splitting for artificial neural networks using SOM-based stratified sampling. **Neural Networks**, Elsevier, v. 23, n. 2, p. 283–294, 2010.

MCCULLOCH, Warren S.; PITTS, Walter. A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, Springer Science e Business Media LLC, v. 5, n. 4, p. 115–133, dez. 1943. DOI: 10.1007/bf02478259. Disponível em: https://doi.org/10.1007/bf02478259.

MCGEOCH, Catherine C. Adiabatic Quantum Computation and Quantum Annealing: Theory and Practice. Morgan & Claypool Publishers LLC, v. 5, n. 2, p. 1–93, jul. 2014. DOI: 10.2200/s00585ed1v01y201407qmc008. Disponível em: https://doi.org/10.2200/s00585ed1v01y201407qmc008.

MEHTA, Pankaj; BUKOV, Marin; WANG, Ching-Hao; DAY, Alexandre G.R.; RICHARDSON, Clint; FISHER, Charles K.; SCHWAB, David J. A high-bias, low-variance introduction to Machine Learning for physicists. **Physics Reports**, Elsevier BV, v. 810, p. 1–124, mai. 2019. ISSN 0370-1573. DOI: 10.1016/j.physrep.2019.03.001. Disponível em: http://dx.doi.org/10.1016/j.physrep.2019.03.001.

MENDOZA, Hector; KLEIN, Aaron; FEURER, Matthias; SPRINGENBERG, Jost Tobias; URBAN, Matthias; BURKART, Michael; DIPPEL, Max; LINDAUER, Marius; HUTTER, Frank. Towards Automatically-Tuned Deep Neural Networks. *In*: HUTTER, Frank; KOTTHOFF, Lars; VANSCHOREN, Joaquin (Ed.).

AutoML: Methods, Sytems, Challenges. [*S.l.*]: Springer, dez. 2018. To appear. cap. 7, p. 141–156.

MILBURN, G. J.; BRAUNSTEIN, Samuel L. Quantum teleportation with squeezed vacuum states. **Physical Review A**, American Physical Society (APS), v. 60, n. 2, p. 937–942, ago. 1999. DOI: 10.1103/physreva.60.937. Disponível em: https://doi.org/10.1103/physreva.60.937.

MNIH, Volodymyr; KAVUKCUOGLU, Koray; SILVER, David; GRAVES, Alex; ANTONOGLOU, Ioannis; WIERSTRA, Daan; RIEDMILLER, Martin. Playing atari with deep reinforcement learning. **arXiv preprint arXiv:1312.5602**, 2013.

MODARES, A.; SOMHOM, S.; ENKAWA, T. A self-organizing neural network approach for multiple traveling salesman and vehicle routing problems. **International Transactions in Operational Research**, Wiley, v. 6, n. 6, p. 591–606, nov. 1999. DOI: 10.1111/j.1475-3995.1999.tb00175.x. Disponível em: https://doi.org/10.1111/j.1475-3995.1999.tb00175.x.

NEURÔNIOS. [*S.l.*]: https://www.coladaweb.com/biologia/histologia/neuronios. Accessed: 2022-11-22.

NEVEN, Hartmut; DENCHEV, Vasil S.; ROSE, Geordie; MACREADY, William G. QBoost: Large Scale Classifier Training withAdiabatic Quantum Optimization. *In*: HOI, Steven C. H.; BUNTINE, Wray (Ed.). **Proceedings of the Asian Conference on Machine Learning**. Singapore Management University, Singapore: PMLR, abr. 2012. (Proceedings of Machine Learning Research), p. 333–348. Disponível em: http://proceedings.mlr.press/v25/neven12.html.

NEVEN, Hartmut; DENCHEV, Vasil S.; ROSE, Geordie; MACREADY, William G. **Training a Binary Classifier with the Quantum Adiabatic Algorithm**. [*S.l.*: *s.n.*], 2008. eprint: arXiv:0811.0416.

NEVEN, Hartmut; DENCHEV, Vasil S.; ROSE, Geordie; MACREADY, William G. **Training a Large Scale Classifier with the Quantum Adiabatic Algorithm**. [*S.l.*: *s.n.*], 2009. eprint: arXiv:0912.0779.

NIELSEN, M.; CHUANG, I. **Quantum Computation and Quantum Information**. 1. ed. [*S.l.*]: Cambridge University Press, 2000.

NISHIMORI, H. Comparison of quantum annealing and simulated annealing. **The European Physical Journal Special Topics**, Springer Science e Business Media LLC, v. 224, n. 1, p. 15–16, fev. 2015. DOI: 10.1140/epjst/e2015-02338-0. Disponível em: https://doi.org/10.1140/epjst/e2015-02338-0.

OLMSCHENK, S.; MATSUKEVICH, D. N.; MAUNZ, P.; HAYES, D.; DUAN, L.-M.; MONROE, C. Quantum Teleportation Between Distant Matter Qubits. **Science**, American Association for the Advancement of Science (AAAS), v. 323, n. 5913, p. 486–489, jan. 2009. DOI: 10.1126/science.1167209. Disponível em: https://doi.org/10.1126/science.1167209.

ONGUN, G.; HALICI, U.; LEBLEBICIOGLU, K.; ATALAY, V.; BEKSAC, M.; BEKSAC, S. Feature extraction and classification of blood cells for an automated differential blood count system. *In*: IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222). [*S.l.*]: IEEE. DOI: 10.1109/ijcnn.2001.938753. Disponível em: https://doi.org/10.1109/ijcnn.2001.938753.

OZBAYOGLU, Ahmet Murat; GUDELEK, Mehmet Ugur; SEZER, Omer Berat. **Deep Learning for Financial Applications : A Survey**. [*S.l.*]: arXiv, 2020. DOI: 10.48550/ARXIV.2002.05786. Disponível em: https://arxiv.org/abs/2002.05786.

PATEL, Jigar; SHAH, Sahil; THAKKAR, Priyank; KOTECHA, K. Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. **Expert Systems with Applications**, Elsevier BV, v. 42, n. 1, p. 259–268, jan. 2015a. DOI: 10.1016/j.eswa.2014.07.040. Disponível em: https://doi.org/10.1016/j.eswa.2014.07.040.

PATEL, Jigar; SHAH, Sahil; THAKKAR, Priyank; KOTECHA, K. Predicting stock market index using fusion of machine learning techniques. **Expert Systems with Applications**, Elsevier BV, v. 42, n. 4, p. 2162–2172, mar. 2015b. DOI: 10.1016/j.eswa.2014.10.031. Disponível em: https://doi.org/10.1016/j.eswa.2014.10.031.

PERES, Asher. Separability Criterion for Density Matrices. **Physical Review Letters**, American Physical Society (APS), v. 77, n. 8, p. 1413–1415, ago. 1996. DOI: 10.1103/physrevlett.77.1413. Disponível em: https://doi.org/10.1103/physrevlett.77.1413.

PIURI, V.; SCOTTI, F. Morphological classification of blood leucocytes by microscope images. *In*: 2004 IEEE International Conference onComputational Intelligence for Measurement Systems and Applications, 2004. CIMSA. [*S.l.*]: IEEE. DOI: 10.1109/cimsa.2004.1397242. Disponível em: https://doi.org/10.1109/cimsa.2004.1397242.

PREVEDEL, Robert; ASPELMEYER, Markus; BRUKNER, Caslav; ZEILINGER, Anton; JENNEWEIN, Thomas D. Photonic entanglement as a resource in quantum computation and quantum communication. **Journal of the Optical Society of America B**, The Optical Society, v. 24, n. 2, p. 241, jan. 2007. DOI: 10.1364/josab.24.000241. Disponível em: https://doi.org/10.1364/josab.24.000241.

QUINLAN, J.R. **Machine Learning**, Springer Nature, v. 1, n. 1, p. 81–106, 1986. DOI: 10.1023/a:1022643204877. Disponível em: https://doi.org/10.1023/a:1022643204877.

RASOULI, Kabir; HSIEH, William W.; CANNON, Alex J. Daily streamflow forecasting by machine learning methods with weather and climate inputs. **Journal of Hydrology**, Elsevier BV, v. 414-415, p. 284–293, jan. 2012. DOI: 10.1016/j.jhydrol.2011.10.039. Disponível em: https://doi.org/10.1016/j.jhydrol.2011.10.039.

RAY, P.; CHAKRABARTI, B. K.; CHAKRABARTI, Arunava. Sherrington-Kirkpatrick model in a transverse field: Absence of replica symmetry breaking due to quantum fluctuations. **Phys. Rev. B**, American Physical Society, v. 39, p. 11828–11832, 16 jun. 1989. DOI: 10.1103/PhysRevB.39.11828. Disponível em: https://link.aps.org/doi/10.1103/PhysRevB.39.11828.

RENES, Joseph M.; BLUME-KOHOUT, Robin; SCOTT, A. J.; CAVES, Carlton M. Symmetric informationally complete quantum measurements. **Journal of Mathematical Physics**, AIP Publishing, v. 45, n. 6, p. 2171–2180, jun. 2004. DOI: 10.1063/1.1737053. Disponível em: https://doi.org/10.1063/1.1737053.

ROCKAFELLAR, R. Tyrrell. **Convex analysis**. Princeton, N. J.: Princeton University Press, 1970. (Princeton Mathematical Series).

ROGERS, Michael L.; SINGLETON, Robert L. Floating-Point Calculations on a Quantum Annealer: Division and Matrix Inversion. **Frontiers in Physics**, v. 8, 2020.

ISSN 2296-424X. DOI: 10.3389/fphy.2020.00265. Disponível em: https://www.frontiersin.org/articles/10.3389/fphy.2020.00265.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, American Psychological Association (APA), v. 65, n. 6, p. 386–408, 1958. DOI: 10.1037/h0042519. Disponível em: https://doi.org/10.1037/h0042519.

RUMELHART, D. E.; MCCLELLAND, J. L. **Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations**. [*S.l.*]: MIT Press, 1986.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, Springer Nature, v. 323, n. 6088, p. 533–536, out. 1986. DOI: 10.1038/323533a0. Disponível em: https://doi.org/10.1038/323533a0.

SAKI, Abdullah Ash; ALAM, Mahabubul; GHOSH, Swaroop. **Study of Decoherence in Quantum Computers: A Circuit-Design Perspective**. [*S.l.*]: arXiv, 2019. DOI: 10.48550/ARXIV.1904.04323. Disponível em: https://arxiv.org/abs/1904.04323.

SAKURAI, J. J.; COMMINS, Eugene D. Modern Quantum Mechanics, Revised Edition. Edição: S. F. Tuan. **American Journal of Physics**, American Association of Physics Teachers (AAPT), v. 63, n. 1, p. 93–95, jan. 1995. DOI: 10.1119/1.17781. Disponível em: https://doi.org/10.1119/1.17781.

SALZBERG, Steven L. C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993. **Machine Learning**, Springer Science e Business Media LLC, v. 16, n. 3, p. 235–240, set. 1994. DOI: 10.1007/bf00993309. Disponível em: https://doi.org/10.1007/bf00993309.

SARANDY, M. S.; WU, L.-A.; LIDAR, D. A. Consistency of the Adiabatic Theorem. **Quantum Information Processing**, Springer Science e Business Media LLC, v. 3, n. 6, p. 331–349, dez. 2004. DOI: 10.1007/s11128-004-7712-7. Disponível em: https://doi.org/10.1007/s11128-004-7712-7.

SCHULD, Maria; PETRUCCIONE, Francesco. **Supervised learning with quantum computers**. 1. ed. Cham, Switzerland: Springer International Publishing, set. 2018. (Quantum Science and Technology).

SEIF, Alireza; LANDSMAN, Kevin A; LINKE, Norbert M; FIGGATT, Caroline; MONROE, C; HAFEZI, Mohammad. Machine learning assisted readout of trapped-ion qubits. **Journal of Physics B: Atomic, Molecular and Optical Physics**, IOP Publishing, v. 51, n. 17, p. 174006, ago. 2018. DOI: 10.1088/1361-6455/aad62b. Disponível em: https://doi.org/10.1088/1361-6455/aad62b.

SELTZER, Michael L.; YU, Dong; WANG, Yongqiang. An investigation of deep neural networks for noise robust speech recognition. *In*: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. [*S.l.*]: IEEE, mai. 2013. DOI: 10.1109/icassp.2013.6639100. Disponível em: https://doi.org/10.1109/icassp.2013.6639100.

SEROV, Valery. The Riemann–Lebesgue Lemma. *In*: APPLIED Mathematical Sciences. [*S.l.*]: Springer International Publishing, 2017. P. 33–35. DOI: 10.1007/978-3-319-65262-7_6. Disponível em: https://doi.org/10.1007/978-3-319-65262-7_6.

SHARMA, Navin; SHARMA, Pranshu; IRWIN, David; SHENOY, Prashant. Predicting solar generation from weather forecasts using machine learning. *In*: 2011 IEEE International Conference on Smart Grid Communications (SmartGridComm). [*S.l.*]: IEEE, out. 2011. DOI: 10.1109/smartgridcomm.2011.6102379. Disponível em: https://doi.org/10.1109/smartgridcomm.2011.6102379.

SHOR, P.W. Algorithms for quantum computation: discrete logarithms and factoring. *In*: PROCEEDINGS 35th Annual Symposium on Foundations of Computer Science. [*S.l.*: *s.n.*], 1994. P. 124–134. DOI: 10.1109/SFCS.1994.365700.

SHOR, Peter W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. **SIAM Journal on Computing**, Society for Industrial & Applied Mathematics (SIAM), v. 26, n. 5, p. 1484–1509, out. 1997. DOI: 10.1137/s0097539795293172. Disponível em: https://doi.org/10.1137/s0097539795293172.

SILVER, David *et al.* Mastering the game of Go with deep neural networks and tree search. **nature**, Nature Publishing Group, v. 529, n. 7587, p. 484–489, 2016.

SIRIGNANO, Justin; SPILIOPOULOS, Konstantinos. DGM: A deep learning algorithm for solving partial differential equations. **Journal of Computational Physics**, Elsevier

BV, v. 375, p. 1339–1364, dez. 2018. DOI: 10.1016/j.jcp.2018.08.029. Disponível em: https://doi.org/10.1016/j.jcp.2018.08.029.

SNOEK, Jasper; LAROCHELLE, Hugo; ADAMS, Ryan P. Practical bayesian optimization of machine learning algorithms. **Advances in neural information processing systems**, v. 25, 2012.

STEINER, Michael. Generalized robustness of entanglement. American Physical Society (APS), v. 67, n. 5, mai. 2003a. DOI: 10.1103/physreva.67.054305. Disponível em: https://doi.org/10.1103/physreva.67.054305.

STEINER, Michael. Generalized robustness of entanglement. **Phys. Rev. A**, v. 67, n. 5, p. 054305, 2003b. DOI: 10.1103/PhysRevA.67.054305.

TAMIR, Boaz; COHEN, Eliahu. **Notes on Adiabatic Quantum Computers**. [*S.l.*]: arXiv, 2015. DOI: 10.48550/ARXIV.1512.07617. Disponível em: https://arxiv.org/abs/1512.07617.

TERHAL, Barbara M. A family of indecomposable positive linear maps based on entangled quantum states. **Linear Algebra and its Applications**, Elsevier BV, v. 323, n. 1-3, p. 61–73, jan. 2001. DOI: 10.1016/s0024-3795(00)00251-2. Disponível em: https://doi.org/10.1016/s0024-3795(00)00251-2.

TERHAL, Barbara M. Detecting quantum entanglement. **Theoretical Computer Science**, Elsevier BV, v. 287, n. 1, p. 313–335, set. 2002. DOI: 10.1016/s0304-3975(02)00139-1. Disponível em: https://doi.org/10.1016/s0304-3975(02)00139-1.

TETKO, Igor V. **Neural Processing Letters**, Springer Nature, v. 16, n. 2, p. 187–199, 2002. DOI: 10.1023/a:1019903710291. Disponível em: https://doi.org/10.1023/a:1019903710291.

THORNTON, Chris; HUTTER, Frank; HOOS, Holger H.; LEYTON-BROWN, Kevin. **Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms**. [*S.l.*]: arXiv, 2012. DOI: 10.48550/ARXIV.1208.3719. Disponível em: https://arxiv.org/abs/1208.3719.

TIEN, F.-C.; YEH, C.-H.; HSIEH, K.-H. Automated visual inspection for microdrills in printed circuit board production. **International Journal of Production Research**,

Informa UK Limited, v. 42, n. 12, p. 2477–2495, jun. 2004. DOI: 10.1080/00207540310001659656. Disponível em: https://doi.org/10.1080/00207540310001659656.

TOH, Kim-Chuan; TODD, Michael J; TÜTÜNCÜ, Reha H. SDPT3—a MATLAB software package for semidefinite programming, version 1.3. **Optimization methods and software**, Taylor & Francis, v. 11, n. 1-4, p. 545–581, 1999.

URSIN, R. *et al.* Entanglement-based quantum communication over 144 km. **Nature Physics**, Springer Science e Business Media LLC, v. 3, n. 7, p. 481–486, jun. 2007. DOI: 10.1038/nphys629. Disponível em: https://doi.org/10.1038/nphys629.

VAZIRI, Alipasha; WEIHS, Gregor; ZEILINGER, Anton. Experimental Two-Photon, Three-Dimensional Entanglement for Quantum Communication. **Physical Review Letters**, American Physical Society (APS), v. 89, n. 24, nov. 2002. DOI: 10.1103/physrevlett.89.240401. Disponível em: https://doi.org/10.1103/physrevlett.89.240401.

VEERACHARY, M.; SENJYU, T.; UEZATO, K. Neural-network-based maximum-power-point tracking of coupled-inductor interleaved-boost-converter-supplied pv system using fuzzy controller. **IEEE Transactions on Industrial Electronics**, Institute of Electrical e Electronics Engineers (IEEE), v. 50, n. 4, p. 749–758, ago. 2003. DOI: 10.1109/tie.2003.814762. Disponível em: https://doi.org/10.1109/tie.2003.814762.

VIDAL, Guifré; TARRACH, Rolf. Robustness of entanglement. American Physical Society (APS), v. 59, n. 1, p. 141–155, jan. 1999a. DOI: 10.1103/physreva.59.141. Disponível em: https://doi.org/10.1103/physreva.59.141.

VIDAL, Guifré; TARRACH, Rolf. Robustness of entanglement. **Phys. Rev. A**, American Physical Society, v. 59, p. 141–155, 1 jan. 1999b. DOI: 10.1103/PhysRevA.59.141. Disponível em: https://link.aps.org/doi/10.1103/PhysRevA.59.141.

WIDROW, Bernard; HOFF, Marcian E. Adaptive Switching Circuits. *In*: 1960 IRE WESCON Convention Record, Part 4. New York: IRE, 1960. P. 96–104.

WOLF, Ronald de. **Quantum Computing: Lecture Notes**. [*S.l.*: *s.n.*], 2019. eprint: arXiv:1907.09415.

WULSIN, D F; GUPTA, J R; MANI, R; BLANCO, J A; LITT, B. Modeling electroencephalography waveforms with semi-supervised deep belief nets: fast classification and anomaly measurement. **Journal of Neural Engineering**, IOP Publishing, v. 8, n. 3, p. 036015, abr. 2011. DOI: 10.1088/1741-2560/8/3/036015. Disponível em: https://doi.org/10.1088/1741-2560/8/3/036015.

WULSIN, Drausin; BLANCO, Justin; MANI, Ram; LITT, Brian. Semi-Supervised Anomaly Detection for EEG Waveforms Using Deep Belief Nets. *In*: 2010 Ninth International Conference on Machine Learning and Applications. [*S.l.*]: IEEE, dez. 2010. DOI: 10.1109/icmla.2010.71. Disponível em: https://doi.org/10.1109/icmla.2010.71.

YANG, Dong; HORODECKI, Michał; HORODECKI, Ryszard; SYNAK-RADTKE, Barbara. Irreversibility for All Bound Entangled States. **Physical Review Letters**, American Physical Society (APS), v. 95, n. 19, out. 2005. DOI: 10.1103/physrevlett.95.190501. Disponível em: https://doi.org/10.1103/physrevlett.95.190501.

ZABORNIAK, Tristan; SOUSA, Rogerio de. Benchmarking Hamiltonian Noise in the D-Wave Quantum Annealer. **IEEE Transactions on Quantum Engineering**, Institute of Electrical e Electronics Engineers (IEEE), v. 2, p. 1–6, 2021. DOI: 10.1109/tqe.2021.3050449. Disponível em: https://doi.org/10.1109%2Ftqe.2021.3050449.

ZYCZKOWSKI, Karol; SOMMERS, Hans-Jürgen. Induced measures in the space of mixed quantum states. **Journal of Physics A: Mathematical and General**, IOP Publishing, v. 34, n. 35, p. 7111, 2001.

ŻYCZKOWSKI, Karol; HORODECKI, Paweł; SANPERA, Anna; LEWENSTEIN, Maciej. Volume of the set of separable states. **Physical Review A**, APS, v. 58, n. 2, p. 883, 1998.

Apêndices

APÊNDICE A – TEOREMA ADIABÁTICO PARA O CASO NÃO DEGENERADO

Neste trabalho seguiremos a seguinte referencia (SARANDY *et al.*, 2004). Levaremos em conta um sistema físico de dimensão finita que evolui através da equação de Schrödinger no caso em que o espectro de energia é não degenerado. Vamos considerar um sistema que evolui através da equação de Schrödinger dependente do tempo:

$$i\hbar\frac{\partial}{\partial t}\left|\psi(t)\right\rangle = H(t)\left|\psi(t)\right\rangle, \tag{152}$$

onde $H(t)$ é o hamiltoniano e $\left|\psi(t)\right\rangle \in \mathcal{H}^D$ é um estado quântico do sistema que pertence a um espaço de Hilbert de dimensão D.

Caso não Degenerado:

Vamos assumir que o espectro de energia de H seja completamente discreto e não degenerado. Podemos então assumir uma base instantânea de autoestados de forma que:

$$H(t)\left|n(t)\right\rangle = E_n\left|n(t)\right\rangle, \tag{153}$$

onde $\left|n(t)\right\rangle$ é uma base de estados ortonormais, $\left\langle k(t)\middle|n(t)\right\rangle = \delta_{n,k}$ com $k, n = 1, ..., D$ e E_n são as autoenergias instantâneas.

Seja $\left|\psi(t)\right\rangle$ uma solução da equação de Schrödinger. Podemos expandir este estado em termos da base de autoestados descrita acima,

$$\left|\psi(t)\right\rangle = \sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle, \tag{154}$$

onde o coeficiente $c_n(t)$ é uma função complexa dependente do tempo, tal que

$$\sum_{n=1}^{D}|c_n(t)|^2 = 1. \tag{155}$$

Substituindo a equação 154 na equação 152, teremos para o primeiro termo

$$
\begin{aligned}
i\hbar\frac{\partial}{\partial t}\left|\psi(t)\right\rangle =& i\hbar\frac{\partial}{\partial t}\left(\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle\right) \\
=& i\hbar\sum_{n=1}^{D}\left(\frac{\partial}{\partial t}c_n(t)\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle \\
&+ i\hbar\sum_{n=1}^{D} c_n(t)\left(\frac{-1}{\hbar}E_n\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle \\
&+ i\hbar\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left(\frac{\partial}{\partial t}\left|n(t)\right\rangle\right),
\end{aligned}
\tag{156}
$$

e o segundo termo será

$$
H(t)(t)\left|\psi(t)\right\rangle = \sum_{n=1}^{D} E_n(t)c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle . \tag{157}
$$

Igualando os termos, teremos:

$$
\begin{aligned}
\sum_{n=1}^{D} E_n(t)c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle =& i\hbar\sum_{n=1}^{D}\left(\frac{\partial}{\partial t}c_n(t)\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle \\
&+ i\hbar\sum_{n=1}^{D} c_n(t)\left(\frac{-1}{\hbar}E_n(t)\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle \\
&+ i\hbar\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left(\frac{\partial}{\partial t}\left|n(t)\right\rangle\right) \\
i\hbar\sum_{n=1}^{D}\left(\frac{\partial}{\partial t}c_n(t)\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left|n(t)\right\rangle =& -i\hbar\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left(\frac{\partial}{\partial t}\left|n(t)\right\rangle\right)
\end{aligned}
\tag{158}
$$

Multiplicando a equação 158 por $\langle k(t)|$, teremos

$$
\begin{aligned}
&\sum_{n=1}^{D}\left(\frac{\partial}{\partial t}c_n(t)\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left\langle k(t)\middle|n(t)\right\rangle = -\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left\langle k(t)\middle|\left(\frac{\partial}{\partial t}n(t)\right)\right\rangle \\
&\left(\frac{\partial}{\partial t}c_k(t)\right)e^{\frac{-1}{\hbar}\int_0^t dt' E_k(t')} = -\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' E_n(t')}\left\langle k(t)\middle|\left(\frac{\partial}{\partial t}n(t)\right)\right\rangle \\
&\left(\frac{\partial}{\partial t}c_k(t)\right) = -\sum_{n=1}^{D} c_n(t)e^{\frac{-1}{\hbar}\int_0^t dt' g_{nk}(t')}\left\langle k(t)\middle|\left(\frac{\partial}{\partial t}n(t)\right)\right\rangle
\end{aligned}
\tag{159}
$$

onde $g_{nk}(t^{'}) = (E_n(t^{'}) - E_k(t^{'}))$.

Derivando a equação 153, podemos reescrever o termo $\left\langle k(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle$ de uma maneira mais conveniente. Desta maneira tomando a derivada temporal, teremos

$$\begin{gathered}\frac{\partial}{\partial t}\left(H(t)\left|n(t)\right\rangle\right) = \frac{\partial}{\partial t}\left(E_n(t)\left|n(t)\right\rangle\right) \\ \left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle + H\left(\frac{\partial}{\partial t}\left|n(t)\right\rangle\right) = \left(\frac{\partial}{\partial t}E_n(t)\right)\left|n(t)\right\rangle + E_n(t)\left(\frac{\partial}{\partial t}\left|n(t)\right\rangle\right).\end{gathered} \tag{160}$$

Multiplicando a equação 160 por $\langle k(t)|$, teremos

$$\begin{aligned}\langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle + \langle k(t)|H\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle &= \langle k(t)|\left(\frac{\partial}{\partial t}E_n(t)\right)\left|n(t)\right\rangle + \langle k(t)|E_n(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle \\ \langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle + E_k\left\langle k(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle &= \left(\frac{\partial}{\partial t}E_n(t)\right)\delta_{n,k} + E_n(t)\left\langle k(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle \\ \langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle &= \left(\frac{\partial}{\partial t}E_n(t)\right)\delta_{n,k} + (E_n(t) - E_k(t))\left\langle k(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle \\ \langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle &= \left(\frac{\partial}{\partial t}E_n(t)\right)\delta_{n,k} + g_{nk}(t)\left\langle k(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle.\end{aligned} \tag{161}$$

A partir da equação 161 podemos concluir que

1. para $k = n$

$$\frac{\partial}{\partial t}E_n = \langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle; \tag{162}$$

2. para $k \neq n$

$$\left\langle k(t)\left|\frac{\partial}{\partial t}n(t)\right.\right\rangle = \frac{\langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle}{g_{nk}} \tag{163}$$

Substituindo a equação 163 na equação 159, teremos

$$\frac{\partial}{\partial t}c_k(t) = -c_k\left\langle k(t)\left|\frac{\partial}{\partial t}k(t)\right.\right\rangle - \sum_{n\neq k,n=1}^{D} c_n\frac{\langle k(t)|\left(\frac{\partial}{\partial t}H\right)\left|n(t)\right\rangle}{g_{nk}}e^{\frac{-1}{\hbar}\int_0^t dt^{'}g_{nk}(t^{'})}. \tag{164}$$

A equação 164 descreve a dinâmica dos coeficientes $c_k(t)$. No teorema adiabático é exigido que a evolução seja independente de cada autoestado instantâneo do hamiltoniano, ou seja, na equação 164 não deve existir termos de acoplamento de

autoestados. Desta maneira devemos eliminar os termos de acoplamento da Equação 164 afim de se adaptar ao teorema adiabático.

Multiplicando o lado direito da equação 164 por um fator de fase $e^{-i\gamma_k(t)}$, com $\gamma_k(t) = i\int_0^t dt' \left\langle k(t') \middle| \frac{\partial}{\partial t} k(t') \right\rangle$. Desta maneira

$$\begin{aligned}\left(\frac{\partial}{\partial t} c_k(t)\right) e^{-i\gamma_k} =& - c_k \left\langle k(t) \middle| \frac{\partial}{\partial t} k(t) \right\rangle e^{-i\gamma_k(t)} \\ & - \sum_{n\neq k, n=1}^{D} c_n(t) \frac{\langle k(t)| \left(\frac{\partial}{\partial t} H\right) |n(t)\rangle}{g_{nk}} e^{-i\gamma_k(t)} e^{\frac{-1}{\hbar}\int_0^t dt' g_{nk}}.\end{aligned} \tag{165}$$

Como,

$$\begin{aligned}\frac{\partial}{\partial t}(c_n(t)) e^{-i\gamma_k(t)} =& \frac{\partial}{\partial t}\left(c_n(t) e^{-i\gamma_k(t)}\right) - c_n(t) \frac{\partial}{\partial t}\left(e^{-i\gamma_k(t)}\right) \\ =& \frac{\partial}{\partial t}\left(c_n(t) e^{-i\gamma_k(t)}\right) - c_n(t) \left\langle k(t) \middle| \frac{\partial}{\partial t} k(t) \right\rangle e^{-i\gamma_k(t)}\end{aligned} \tag{166}$$

Substituindo o resultado acima na equação 165, obteremos

$$\begin{aligned}\frac{\partial}{\partial t}\left(c_k(t) e^{-i\gamma_k(t)}\right) =& - \sum_{n\neq k, n=1}^{D} c_n(t) \frac{\langle k(t)| \left(\frac{\partial}{\partial t} H\right) |n(t)\rangle}{g_{nk}} e^{-i\gamma_k(t)} e^{\frac{-i}{\hbar}\int_0^t dt' g_{nk}(t')} \\ =& - \sum_{n\neq k, n=1}^{D} \frac{F_{nk}}{g_{nk}} e^{\frac{-i}{\hbar}\int_0^t dt' g_{nk}(t')},\end{aligned} \tag{167}$$

onde $f_{nk} = c_n(t) \langle k(t)| \left(\frac{\partial}{\partial t} H\right) |n(t)\rangle e^{-i\gamma_k(t)}$. Integrando a equação 167 no intervalo $t \in [0, T]$, onde T é o tempo total da evolução

$$\begin{aligned}\int_0^T \frac{\partial}{\partial t}\left(c_k(t) e^{-i\gamma_k(t)}\right) dt =& - \sum_{n\neq k, n=1}^{D} \int_0^T \frac{F_{nk}}{g_{nk}} e^{\frac{-i}{\hbar}\int_0^t dt' g_{nk}(t')} dt \\ c_k(T) e^{-i\gamma_k(T)} =& c_n(0) - i\hbar \sum_{n\neq k, n=1}^{D} \int_0^T \frac{F_{nk}}{g_{nk}} \frac{d}{dt}\left(e^{\frac{-i}{\hbar}\int_0^T dt' g_{nk}(t')}\right) dt \\ =& c_n(0) - i\hbar \sum_{n\neq k, n=1}^{D} \int_0^T \frac{d}{dt}\left(\frac{F_{nk}}{g_{nk}^2} e^{\frac{-i}{\hbar}\int_0^T dt' g_{nk}(t')}\right) dt \\ & + i\hbar \sum_{n\neq K, n=1}^{D} \int_0^T \frac{d}{dt}\left(\frac{F_{nk}}{g_{nk}^2}\right) e^{\frac{-i}{\hbar}\int_0^T dt' g_{nk}(t')} dt.\end{aligned} \tag{168}$$

A primeira integral do lado direito da expressão 168 é simplesmente

$$\int_0^T \frac{d}{dt}\left(\frac{F_{nk}}{g_{nk}^2} e^{\frac{-i}{\hbar}\int_0^T dt' g_{nk}}\right) dt = \frac{F_{nk}(T)}{g_{nk}^2(T)} e^{\frac{-i}{\hbar}\int_0^T g_{nk}(t')dt'} - \frac{F_{nk}(0)}{g_{nk}^2(0)}. \tag{169}$$

Para solucionar a segunda integral utilizamos a mudança de variável $t = sT$, onde $0 \leq s \leq 1$. Assim, ficamos com

$$\int_0^T \frac{d}{dt}\left(\frac{F_{nk}}{g_{nk}^2}\right) e^{\frac{-i}{\hbar}\int_0^T g_{nk}dt'} dt = \int_0^1 \frac{d}{ds}\left(\frac{F_{nk}}{g_{nk}^2}\right) e^{\frac{-iT}{\hbar}\int_0^s g_{nk}ds'} ds. \tag{170}$$

De acordo com o lema de Riemann-Lebesgue (SEROV, 2017),

$$\lim_{|\alpha|\to\infty} \int_a^b f(x) e^{i\alpha x} dx = 0. \tag{171}$$

Juntando o lema de Reimann-Lebesgue 171 com a equação 170, obtemos

$$\lim_{T\to\infty} \int_0^1 \frac{d}{ds}\left(\frac{F_{nk}}{g_{nk}^2}\right) e^{\frac{-iT}{\hbar}\int_0^s g_{nk}(s')ds'} ds = 0. \tag{172}$$

Substituímos então as equações 169 e 172 na equação 168, obtemos a seguinte expressão para os coeficientes C_n

$$\begin{aligned} c_n(T) &= c_n(0)e^{i\gamma_k(T)} + i\hbar \sum_{n\neq k, n=1}^{D} \left[\frac{F_{nk}(0)}{g_{nk}^2(0)} - \frac{F_{nk}(T)}{g_{nk}^2(T)} e^{\frac{-i}{\hbar}\int_0^T g_{nk}dt'}\right] e^{i\gamma_k(T)}, \\ &= c_n(0)e^{i\gamma_k(T)} + i\hbar \sum_{n\neq k, n=1}^{D} \left[\frac{c_{nk}(0)\langle k|\dot{H}|n\rangle}{g_{nk}^2(0)} - \frac{c_{nk}(T)\langle k|\dot{H}|n\rangle}{g_{nk}^2(T)} e^{\frac{-i}{\hbar}\int_0^T g_{nk}dt'} e^{-i\gamma_k(T)}\right] e^{i\gamma_k(T)}. \end{aligned} \tag{173}$$

Impondo que o sistema evolui adiabaticamente, devemos eliminar os termos com acoplamento entre os estados $|n\rangle$ e $|k\rangle$. Desta maneira temos que o termo $\frac{\langle k|\dot{H}|n\rangle}{g_{nk}^2(t)}$ deve ser muito menor que 1 para qualquer tempo t, pois $e^{-\frac{i}{\hbar}\int_0^t g_{nk}dt'}$ é uma função limitada. Mais especificamente, a seguinte condição deve ser imposta sobre o sistema:

$$\mathcal{F} << \mathcal{G}^2, \tag{174}$$

$$\mathcal{F} = \hbar \max_{0\leq t\leq T} \left|\left\langle n\left|\frac{dH}{dt}\right|k\right\rangle\right|, \tag{175}$$

$$\mathcal{G} = \min_{0\leq t\leq T} |g_{nk}|. \tag{176}$$

As condições de máximo e mínimo sobre $\mathcal{F}$ e $\mathcal{G}$ respectivamente, indica que estamos lidando com o pior caso possível. Esta condição nos diz que para todo par de níveis de energia E_n e E_k, o máximo valor obtido para o elemento de matriz da variação temporal do Hamiltoniano deve ser muito menor que o mínimo valor obtido para a diferença de energia entre estes dois níveis. Esta condição garante que a evolução ocorra em um regime adiabático.