UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MARIA ANGÉLICA KRÜGER MIRANDA

CONSTRUÇÃO DE UM MÓDULO QUÂNTICO PARA O CLASSIFICADOR BASEADO EM FLORESTA DE CAMINHOS ÓTIMOS

BAURU
Novembro/2023

MARIA ANGÉLICA KRÜGER MIRANDA

CONSTRUÇÃO DE UM MÓDULO QUÂNTICO PARA O CLASSIFICADOR BASEADO EM FLORESTA DE CAMINHOS ÓTIMOS

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. João Paulo Papa
Coorientador: Prof. Dr. Felipe Fernandes Fanchini

BAURU
Novembro/2023

Maria Angélica Krüger Miranda

Construção de um módulo quântico para o classificador baseado em floresta de caminhos ótimos

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. João Paulo Papa
Orientador
Universidade Estadual Paulista “Júlio de Mesquita Filho”
Faculdade de Ciências
Departamento de Computação

Profa. Dra. Simone das Graças Domingues Prado
Universidade Estadual Paulista “Júlio de Mesquita Filho”
Faculdade de Ciências
Departamento de Computação

Prof. Dr. Kelton Augusto Pontara da Costa
Universidade Estadual Paulista “Júlio de Mesquita Filho”
Faculdade de Ciências
Departamento de Computação

Bauru, 16 de Novembro de 2023.

Dedico este trabalho ao meu namorado, cujo apoio e presença iluminam meus caminhos nos momentos mais desafiadores, e aos meus queridos amigos, que tornam a jornada da vida mais leve e divertida.

Agradecimentos

Durante esta minha jornada, muitas pessoas importantes impactaram a minha vida e fizeram com que eu chegasse até este ponto. Gostaria de expressar os meus sinceros agradecimentos à minha família, especialmente à minha mãe e aos meus tios, Ligia, Jaciara e João. Por mais caótica que a vida seja, vocês me ensinaram a continuar lutando para que meus sonhos se realizem. Obrigada por estarem ao meu lado e me apoiarem em todos os momentos possíveis.

Agradeço desde o meu primeiro até o último professor com quem tive aula e contato. Todos vocês me mostraram a importância do conhecimento, mesmo que sua busca seja desafiadora. Quero agradecer especialmente aos meus orientadores, João Paulo e Felipe, por permitirem que este projeto seja realizado e mostrarem como a vida acadêmica é desafiadora, mas gratificante.

Quero agradecer ao meu namorado, Nícolas, por ficar ao meu lado e me amar em todos os momentos. Nick, agradeço por toda sua ajuda e conselhos. Você me ensina constantemente a me manter de pé e não desistir todas às vezes que as ondas da vida tentam derrubar. Você trouxe a coragem que me faltava.

Agradeço a Toki, Nih, Cass, Gustavo, Zastim, Davizão e todos os que conheci na faculdade por tornarem meus dias menos desafiadores e por me mostrarem que vale a pena aproveitar todos os pequenos momentos cotidianos. Também quero agradecer ao Netto, Dago e Arthur por estarem comigo desde o colegial. Obrigada por todos os filmes e jogatinas que tivemos, vocês me mostraram que o tempo não precisa ser nosso inimigo.

Por fim, quero agradecer à minha versão que não desistiu. Às vezes, a jornada é difícil, mas lembre-se que você não deve hesitar. Ao olhar para trás, lembre-se do quanto você batalhou e continue seguindo em frente, pois por mais que você caia, você sempre se levantará e, caso não tenha força própria para isso, terá seus amigos e familiares para te ajudar.

Obrigada a todos que embarcaram nesta minha jornada, desde os momentos mais reflexivos e de sábios conselhos até as conversas mais banais para relaxar.

”Longe do estéril turbilhão da rua
Beneditino escreve! No aconchego
Do claustro, na paciência e no sossego
Trabalha e teima, e lima, e sofre e sua!

Mas que na força se disfarce o emprego
Do esforço: e trama viva se construa
De tal modo, que a imagem fique nua
Rica mas sóbria, como um templo grego

Não se mostre na fábrica o suplício
Do mestre. E natural, o efeito agrade
Sem lembrar os andaimes do edifício:

Porque a Beleza, gêmea da Verdade,
Arte pura, inimiga do artifício,
É a força e a graça na simplicidade.”
A um Poeta - Olavo Bilac

Resumo

O interesse pela computação quântica tem experimentado um aumento constante, impulsionado principalmente pelas recentes inovações tecnológicas anunciadas por empresas renomadas, como a IBM e a Google. Essas inovações têm o potencial de solucionar desafios significativos em diversos setores, como financeiro, médico, físico, farmacêutico, químico, entre outros. Esse potencial advém da capacidade da computação quântica de resolver problemas que eram anteriormente considerados inviáveis. Concomitantemente a esse crescimento, o campo do aprendizado de máquina tem se destacado como uma ferramenta computacional crucial para o desenvolvimento de novas soluções e tecnologias. Desta forma, o trabalho tem por interesse implementar e avaliar um módulo quântico para o classificador baseado em floresta de caminhos ótimos para explorar a abordagem híbrida do aprendizado de máquina quântico. Os experimentos conduzidos revelaram desafios a serem superado, no entanto, a implementação demonstrou ser promissora devido à sua viabilidade, além de representar o primeiro estudo sobre a aplicação da computação quântica neste classificador em específico.

Palavras-chave: Computação Quântica; Aprendizado de Máquina; Otimização Quântica; Classificação Baseada em Floresta de Caminhos Ótimos.

Abstract

Interest in quantum computing has seen a consistent upward trend, primarily propelled by recent technological advancements unveiled by esteemed companies such as IBM and Google. These breakthroughs possess the potential to address significant challenges across various sectors, including finance, healthcare, physics, pharmaceuticals, chemistry, and others. This potential arises from the capability of quantum computing to tackle problems that were hitherto deemed impracticable. Concurrently with this burgeoning interest, the field of machine learning has emerged as a pivotal computational tool for the development of novel solutions and technologies. Hence, the objective of this research is to implement and assess a quantum module for the classifier, utilizing a forest of optimal paths to explore the hybrid approach to quantum machine learning. The conducted experiments have unveiled certain challenges that need to be surmounted. Nonetheless, the implementation has demonstrated promise owing to its feasibility, in addition to representing the pioneering study on the application of quantum computing in this specific classifier.

Keywords: Quantum Computing; Machine Learning; Quantum Optimization; Classification Based on Optimum-Path Forest.

Lista de figuras

Figura 1 – Portas lógicas quânticas 19
Figura 2 – Circuito de teletransporte quântico 20
Figura 3 – Conjunto de treinamento rotulado para aprendizado supervisionado 21
Figura 4 – Clusterização para detecção de grupos distintos 22
Figura 5 – Aprendizado semissupervisionado 22
Figura 6 – Aprendizado por reforço 23
Figura 7 – Conjunto de treinamento modelado como um grafo 24
Figura 8 – Etapa de treinamento 25
Figura 9 – Etapa de classificação 25
Figura 10 – Abordagens do aprendizado de máquina quântico 26
Figura 11 – Circuito recursivo FALQON 28
Figura 12 – Base de dados Boat com suas três classes em evidência no espaço 33
Figura 13 – Diagrama de classes simplificado da aplicação 35
Figura 14 – Fluxograma da aplicação usando o módulo desenvolvido 36
Figura 15 – Fluxo de controle do programa principal 37
Figura 16 – Código hamiltoniano 38
Figura 17 – Código FALQON 39
Figura 18 – Gráfico de energia para o experimento 1 42
Figura 19 – Probabilidade associada a cada estado para o experimento 1 43
Figura 20 – Gráfico de energia para o experimento 2 44
Figura 21 – Probabilidade associada a cada estado para o experimento 2 44

Lista de quadros

Quadro 1 – Especificações Técnicas do Acer Aspire 3 29

Lista de tabelas

Tabela 1 – Conjunto de dados 32
Tabela 2 – Base de dados com oito amostras 42
Tabela 3 – Base de dados com dez amostras 43

Lista de abreviaturas e siglas

FALQON	*Feedback-Based Quantum Optimization*
KNN	*K-Nearest Neighbors*
MST	*Minimum Spanning Tree*
OPF	*Optimum-Path Forest*
QAOA	*Quantum Aproximate Optimization Algorithm*
QML	*Quantum Machine Learning*
QUBO	*Quadratic Unconstrained Binary Optimization*
TSP	*Traveling Salesman Problem*

Sumário

1 INTRODUÇÃO 14
1.1 Problemática 15
1.2 Justificativa 15
1.3 Objetivos 16
1.3.1 Objetivo Geral 16
1.3.2 Objetivos Específicos 16
1.4 Organização do Trabalho 16

2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 Computação Quântica 17
2.2 Aprendizado de Máquina 20
2.2.1 Classificador Baseado em Floresta de Caminhos Ótimos 23
2.3 Aprendizado de Máquina Quântico 26

3 METODOLOGIA 29
3.1 Ferramentas 29
3.1.1 Pipenv 30
3.1.2 OPFython 30
3.1.3 Qiskit 31
3.1.4 QuTiP 31
3.1.5 Base de Dados 32
3.2 Abordagem Proposta 33
3.2.1 Modelagem do problema do caixeiro viajante 34
3.2.2 Arquitetura Geral 35
3.2.3 Desenvolvimento da aplicação 37
3.2.4 Dificuldade no desenvolvimento 39

4 EXPERIMENTAÇÃO E ANÁLISE DOS RESULTADOS 41
4.1 Experimentos e Resultados Obtidos 41
4.1.1 Experimento 01 41
4.1.2 Experimento 02 43
4.2 Discussões 45

5 CONSIDERAÇÕES FINAIS 46
5.1 Trabalhos Futuros 46

REFERÊNCIAS 47

1 Introdução

O avanço humano está em constante atualização devido às tecnologias modernas que permitem explorar conceitos novos e concretizar ideias inatingíveis. A descoberta de novas vacinas através do mapeamento genético, a condução autônoma em veículos, as recomendações personalizadas e precisas aos compradores e até as simulações sobre o cosmos e outros campos da ciência são apenas alguns exemplos de como a sociedade usa as novas tendências tecnológicas.

Essas tendências compartilham uma característica em comum: a utilização da inteligência artificial. Os estudos nesse campo tiveram início em 1948, quando Warren McCulloh e Walter Pitts elaboraram o primeiro modelo computacional destinado a explicar o funcionamento dos neurônios (MCCULLOCH; PITTS, 1943). Ao longo dos anos, o avanço nesse campo resultou em melhorias significativas nas simulações do comportamento humano. A inteligência artificial tem desempenhado um papel fundamental na criação de obras de arte digital a partir de frases geradas pelo usuário e na busca de respostas específicas e personalizadas por meio de assistentes de chat inteligentes, como o ChatGPT da OpenAI. No entanto, é importante observar que, embora a inteligência artificial tenha uma ampla aplicabilidade, ela não se adapta a todos os cenários, especialmente quando se trata de problemas altamente complexos ou que envolvem a manipulação de grandes volumes de dados, que podem desafiar até mesmo os supercomputadores disponíveis na atualidade.

Em paralelo com a inteligência artificial, outro campo ganha espaço no mercado e nas produções científicas: a computação quântica. Em 1982, Richard Feynman alegou que os problemas de mecânica quântica só poderiam ser resolvidos com uso de computadores que operassem nos mesmos moldes (FEYNMAN, 1982). Esse novo modelo de máquina se basearia nos principais conceitos da mecânica quântica: a superposição e o emaranhamento dos estados. Além dos problemas da mecânica quântica também há problemas computacionais que podem ser solucionados por este novo paradigma, como a decomposição de grandes números em fatores primos e otimizações de problemas. Diante disso, várias empresas vêm investindo cada vez mais neste novo mundo computacional. A IBM já anunciou diversos processadores *quantum* com capacidade cada vez maiores de *qubits*, como o caso do *Eagle* de 127 *qubits* e até mesmo o futuro lançamento da empresa, o *IBM Quantum Condor* de 1121 *qubits*, previsto para final de 2023 (LOREDO, 2021).

Estudos, como o conduzido por Miano e Oliveira (2021), indicam que os desafios enfrentados pelo setor de inteligência artificial podem ser superados por meio

da aplicação da computação quântica. Isso dá origem a um novo campo de pesquisa que combina os princípios da física, da matemática e da computação, conhecido como *Quantum Machine Learning* (QML, Aprendizado de Máquina Quântico). O QML amplia a gama de problemas que podem ser resolvidos e, ao mesmo tempo, tem o potencial de reduzir o tempo necessário para o treinamento de determinados modelos de aprendizado, podendo resultar em melhorias significativas no desempenho e na eficácia destes algoritmos.

1.1 Problemática

Durante alguns anos, a Lei de Moore tem alertado sobre a limitação física dos computadores tradicionais, especialmente em relação à quantidade máxima de transistores que um processador pode suportar (MOORE, 2006). Essa limitação impõe a necessidade de explorar alternativas viáveis para que os inúmeros desafios computacionais que se mostram intratáveis para os processadores convencionais sejam tratados. Exemplos notáveis incluem problemas como a fatoração de inteiros em tempo polinomial usando números primos e a simulação de experimentos associados à física moderna.

A análise extensiva de dados necessária para o treinamento de modelos de aprendizado de máquina tem sido impactada pelo crescente volume de informações. Isso não apenas prolonga o tempo para gerar novos modelos de aprendizado, mas também demanda uma capacidade de processamento consideravelmente maior por parte das máquinas. Fora isso, há poucos modelos quânticos comparados à diversidade dos tradicionais, fazendo com que alguns problemas não sejam explorados no contexto quântico.

1.2 Justificativa

Em um contexto onde o volume de dados cresce exponencialmente, os modelos de aprendizado de máquina convencionais enfrentam desafios significativos em lidar com essa explosão de informações. Nesse cenário, os modelos quânticos emergem como uma promissora alternativa, oferecendo um potencial substancial para lidar com a complexidade e a escala massiva de dados. Além disso, o campo da computação quântica demonstra um desempenho promissor na resolução de problemas desafiadores que, muitas vezes, são inviáveis para os modelos clássicos. Problemas como fatoração de inteiros em primos com tempo polinomial e simulações de fenômenos quânticos são exemplos que destacam a eficácia singular dos modelos de aprendizado de máquina quânticos.

Nesse contexto, a exploração da vantagem quântica para a criação de modelos de aprendizado de máquina híbridos, combinando elementos de algoritmos quânticos e clássicos, pode resultar em um aprimoramento significativo do desempenho. Além disso, vale ressaltar que, embora o campo da computação quântica esteja avançando rapidamente, ele ainda não oferece a mesma diversidade de modelos de aprendizado disponíveis no contexto clássico. Portanto, é essencial dedicar esforços à adaptação desses modelos tradicionais para o cenário quântico, uma vez que diferentes aplicações podem requerer algoritmos específicos para obter os melhores resultados.

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo principal deste projeto consiste em explorar as técnicas envolvidas no aprendizado de máquina quântico a fim de aplicar a um modelo de aprendizado tradicional e verificar como este se comporta. Para isso, será implementado um módulo quântico que realizará um procedimento do classificador supervisionado baseado em florestas de caminhos ótimos. Por fim, serão realizados testes e comparações do modelo híbrido com o clássico, como uma forma de validar o módulo implementado.

1.3.2 Objetivos Específicos

- Estudar as bases da computação quântica, algoritmos quânticos de otimização e o funcionamento do classificador baseado em floresta de caminhos ótimos;
- Estruturar e implementar o módulo quântico na biblioteca OPFython através das bibliotecas Qiskit e Qutip; e
- Realizar testes para validação do módulo construído e comparar esta versão com o modelo tradicional.

1.4 Organização do Trabalho

A presente monografia conta com mais quatro capítulos principais. O Capítulo 2 aborda os conceitos essenciais sobre a computação quântica, o aprendizado de máquina tradicional juntamente com um modelo de classificação e por fim o aprendizado máquina e otimizações quânticas. O Capítulo 3 descreve o desenvolvimento do módulo, desde as ferramentas usadas até a arquitetura e implementação do projeto. O capítulo Capítulo 4 discute os experimentos realizados e os resultados obtidos. Por fim, o Capítulo 5 apresenta as considerações finais obtidas por este trabalho, além de possíveis trabalhos futuros que possam ser desenvolvidos.

2 Fundamentação Teórica

O presente capítulo abordará conceitos fundamentais relacionados à computação quântica, ao aprendizado de máquina clássico e ao aprendizado de máquina quântico.

2.1 Computação Quântica

A representação da informação na computação é viabilizada através dos *bits*, com a capacidade de assumir um de dois valores distintos: 0 ou 1. De maneira similar, a computação quântica faz uso dos *qubits* para o armazenamento e manipulação de informações. Os *qubits*, assim como os *bits* clássicos, também podem assumir dois estados, porém possuem uma particularidade adicional: podem estar também superpostos. Em outras palavras, o qubit pode estar *entre* 0 e 1, representando assim infinitos estados quânticos. Essa propriedade singular dos *qubits* é resultado dos postulados da mecânica quântica e, para uma compreensão mais aprofundada, é necessário abordar concisamente os conceitos essenciais dessa área.

A mecânica quântica é descrita pelos pesquisadores Nielsen e Chuang (2010) como uma ferramenta matemática utilizada para descrever teorias físicas mais precisas, onde os *qubits*, representam estados quânticos e estes são descritos através da notação de Dirac (1982). Sendo assim, o estado poderá ser representado por um vetor coluna, denominado por *ket* ou vetor de estado, cujos elementos pertencem ao conjunto dos números complexos, ou seja:

$$|\psi\rangle = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix}, \tag{2.1}$$

onde $|\psi\rangle \in \mathbb{C}^n$ e $a_n \in \mathbb{C}$. Ao aplicar a operação de conjugação transposta neste *ket* $|\psi\rangle$ é obtido um vetor linha conhecido como *bra* e representado da seguinte maneira:

$$\langle\psi| = |\psi\rangle^\dagger = \begin{bmatrix} a_0^* & a_1^* & \cdots & a_n^* \end{bmatrix}. \tag{2.2}$$

Após a definição dos vetores de estado, torna-se viável identificar os estados fundamentais da computação quântica, aqueles que formam a base computacional, representados pelos *kets*:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{e} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{2.3}$$

Os outros infinitos estados que o *qubit* pode assumir é matematicamente representado abaixo,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{2.4}$$

na qual o estado $|\psi\rangle$ é uma combinação linear da base computacional. Esta característica do *qubit* é conhecida como princípio da superposição e está presente nos postulados da mecânica quântica. Os coeficientes α e β pertencem ao conjunto dos números complexos e representam as amplitudes de probabilidade desse estado. Determinar o módulo desses valores permite a obtenção das probabilidades associadas aos estados do conjunto de qubits. Assim, é possível concluir que os estados quânticos são normalizados, uma vez que a soma dos quadrados dos módulos dos coeficientes é igual a um, isto é, $|\alpha|^2 + |\beta|^2 = 1$.

O princípio da superposição é essencial para a criação de emaranhamentos nos estados quânticos, permitindo um aumento exponencial na capacidade de processamento (IBM, 2023). Para compreender o emaranhamento, considere um sistema composto por dois ou mais *qubits*, cuja representação é obtida através do produto tensorial, formando assim um sistema global. Quando esses *qubits* se tornam emaranhados, a descrição do estado não pode mais ser feita de maneira independente nos subconjuntos individuais. Na Equação abaixo, tem-se um exemplo de sistema separável,

$$|\psi_0\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |10\rangle\right) = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes |0\rangle, \tag{2.5}$$

enquanto os estados emaranhados conhecidos como Estados de Bell são dados por

$$\begin{aligned}
|\Psi^{00}\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \\
|\Psi^{01}\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \\
|\Psi^{10}\rangle &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \\
|\Psi^{11}\rangle &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle).
\end{aligned} \tag{2.6}$$

Os *kets* $|0\rangle$ e $|1\rangle$ representam valores mensuráveis em um sistema quântico, como os *spins* de elétrons ou as polarizações de um fóton de luz. Essas grandezas físicas mensuráveis são descritas por meio de operadores chamados de observáveis. Quando a medição de um observável é realizada, um estado $|\psi\rangle$ qualquer sofre um colapso, e o resultado obtido é um dos possíveis autovalores do operador associado, correspondendo a uma das propriedades mensuráveis do sistema. Dessa forma, as medições em estados superpostos induzem ao colapso, alterando o estado inicial, e o resultado da medição é de natureza probabilística. Esse fenômeno contrasta com a medição clássica, pois esta geralmente revela um valor único e determinístico.

Os circuitos quânticos são constituídos por portas lógicas quânticas, as quais efetuam transformações nos estados quânticos por meio de operadores. Esses operadores lineares são representados por matrizes que realizam tais transformações sem modificar o grau de pureza de um estado quântico, mantendo-o normalizado. Sendo assim, essas operações podem ser visualizadas como rotações na esfera de Bloch, uma representação gráfica dos estados quânticos, permitindo uma compreensão visual das mudanças e transformações ocorridas. A Figura 1 ilustra as principais portas lógicas quânticas, suas representações matriciais e seus efeitos sobre a base computacional.

Figura 1 – Portas lógicas quânticas

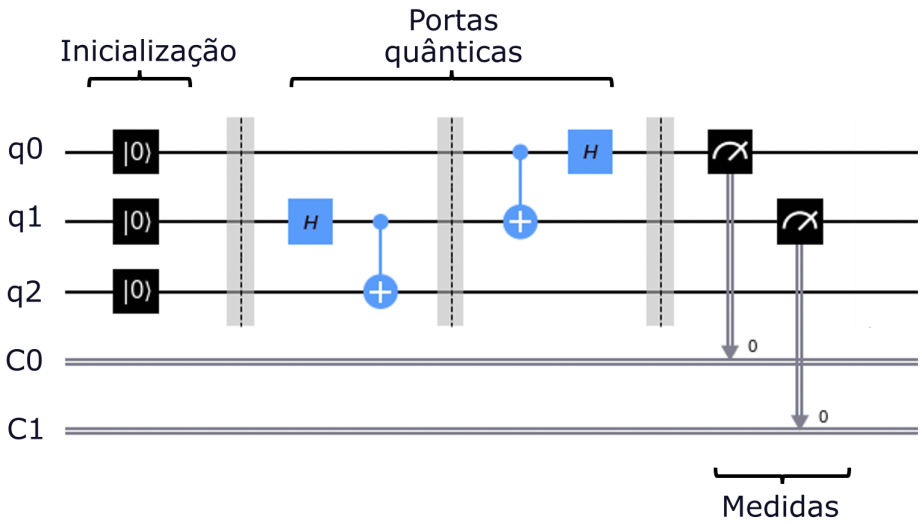
	NOME	NOTAÇÃO	REPRESENTAÇÃO MATRICIAL	ATUAÇÃO NA BASE COMPUTACIONAL
Portas de 1-qbit	Identidade	id	$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$	$\|0\rangle \to \|0\rangle$ $\|1\rangle \to \|1\rangle$
	Porta NOT	X	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$	$\|0\rangle \to \|1\rangle$ $\|1\rangle \to \|0\rangle$
	Porta Y	Y	$\begin{bmatrix}0 & -i\\i & 0\end{bmatrix}$	$\|0\rangle \to i\|1\rangle$ $\|1\rangle \to -i\|0\rangle$
	Porta Z	Z	$\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}$	$\|0\rangle \to \|0\rangle$ $\|1\rangle \to -\|1\rangle$
	Porta de Hadamard	H	$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$	$\|0\rangle \to \frac{\|0\rangle+\|1\rangle}{\sqrt{2}}$ $\|1\rangle \to \frac{\|0\rangle-\|1\rangle}{\sqrt{2}}$
	Porta de fase	S	$\begin{bmatrix}1 & 0\\0 & i\end{bmatrix}$	$\|0\rangle \to \|0\rangle$ $\|1\rangle \to i\|1\rangle$
		S†	$\begin{bmatrix}1 & 0\\0 & -i\end{bmatrix}$	$\|0\rangle \to \|0\rangle$ $\|1\rangle \to -i\|1\rangle$
	Porta de T	T	$\begin{bmatrix}1 & 0\\0 & e^{i\pi/4}\end{bmatrix}$	$\|0\rangle \to \|0\rangle$ $\|1\rangle \to e^{i\pi/4}\|1\rangle$
		T†	$\begin{bmatrix}1 & 0\\0 & e^{-i\pi/4}\end{bmatrix}$	$\|0\rangle \to \|0\rangle$ $\|1\rangle \to e^{-i\pi/4}\|1\rangle$
Porta de 2-qbits	Porta CNOT - Not-Controlado		$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix}$	$\|00\rangle \to \|00\rangle$ $\|01\rangle \to \|01\rangle$ $\|10\rangle \to \|11\rangle$ $\|11\rangle \to \|10\rangle$
Operação	Medida na base computacional padrão (Z)			
	Barreira de evolução		A barreira evita a evolução do q-bit através da linha	

Fonte: (RABELO; COSTA, 2018)

Os algoritmos quânticos são construídos por meio de circuitos e portas lógi-

cas quânticas. A Figura 2 apresenta o circuito do algoritmo de teletransporte quântico. Neste esquema, cada linha horizontal simboliza um *qubit* ($q0$, $q1$ e $q2$), sendo a leitura realizada da esquerda (dados quânticos iniciais) para a direita (dados quânticos finais), enquanto, as operações realizadas nesses *qubits* são representadas por caixas dispostas ao longo dos fios e ao final são realizadas medições para que o estado colapse e o resultado seja armazenado nos bits clássicos $c0$ e $c1$.

Figura 2 – Circuito de teletransporte quântico



Fonte: Adapatada de QISKIT CONTRIBUTORS (2023a)

2.2 Aprendizado de Máquina

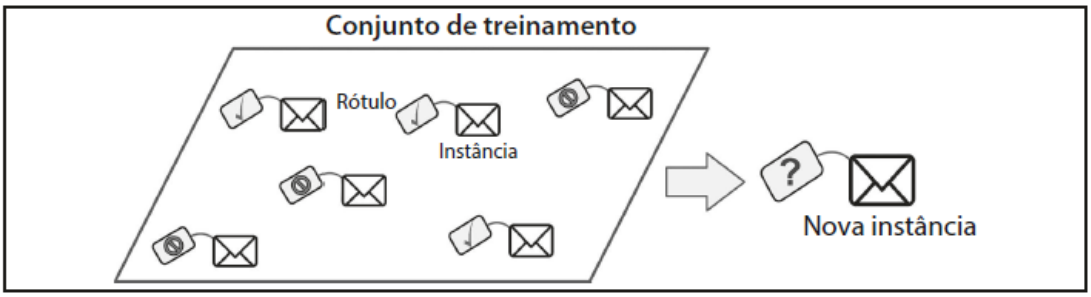
Aprendizado de máquina é a ciência da programação que possibilita os computadores aprenderem com os dados, sem serem explicitamente programados para isso. O cientista norte-americano Arthur Lee Samuel é reconhecido como um dos pioneiros neste campo. Em seu estudo, foi demonstrado que um computador programado para aprender a jogar damas superava a habilidade de um programa criado por humanos (SAMUEL, 1959). Esse trabalho impulsionou o desenvolvimento de outros algoritmos, expandindo significativamente as áreas de aplicação que poderiam se beneficiar com essa nova abordagem.

O autor Geron (2019) categoriza os sistemas de aprendizado em quatro principais vertentes, considerando o tipo de supervisão que receberam durante o treinamento. Essas categorias incluem o aprendizado supervisionado, não supervisionado, semissupervisionado e por reforço.

No âmbito do aprendizado supervisionado, os dados de treinamento são acompanhados de rótulos cujo propósito é indicar a classe à qual pertencem. Esse modelo

é comumente empregado em tarefas de classificação, como os filtros de spam utilizados em correios eletrônicos, que estão representados na Figura 3. Além disso, outra aplicação típica consiste na previsão de valores numéricos, como o preço de imóveis, a partir de um conjunto de características relevantes (tamanho do terreno, localização, qualidade, dentre outros atributos). Nesse contexto, existem diversos algoritmos amplamente reconhecidos que desempenham essas atividades, sendo alguns dos mais notáveis: regressão linear, regressão logística, máquinas de vetores de suporte, *K-Nearest Neighbors* (KNN, K-Vizinhos Mais Próximos), árvores de decisão e redes neurais.

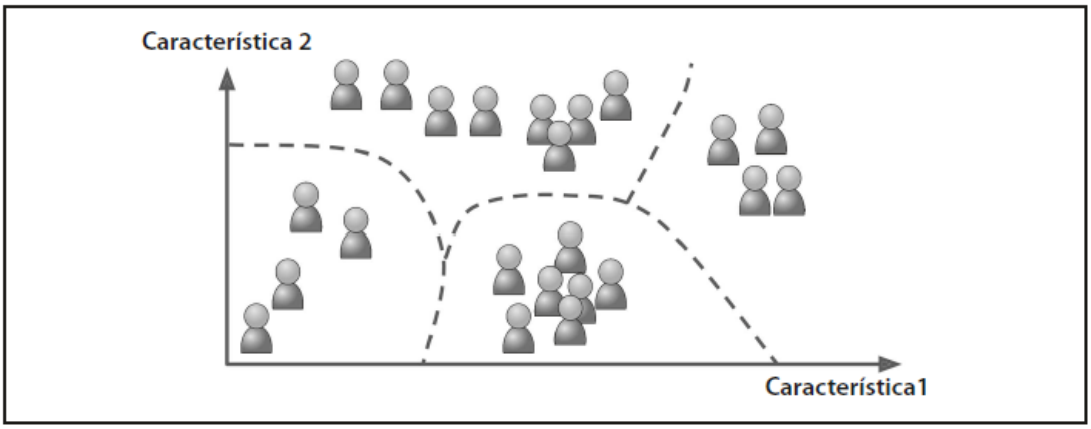
Figura 3 – Conjunto de treinamento rotulado para aprendizado supervisionado



Fonte: Geron (2019)

No modelo não supervisionado, os dados de treinamento não contém rótulos, uma vez que a responsabilidade de definir as classes recai sobre o próprio algoritmo. Uma aplicação prática desse tipo de abordagem ocorre em propagandas personalizadas, onde públicos com características semelhantes são agrupados para recomendar produtos específicos, utilizando uma técnica não supervisionada chamada clusterização, como mostrado na Figura 4. Além disso, o aprendizado não supervisionado é crucial na visualização de dados complexos em representações mais simples, facilitando sua interpretação. Essa abordagem também é utilizada na extração de características ao reduzir a dimensionalidade de imagens e na detecção de anomalias, como transações incomuns em cartões de crédito e peças defeituosas em processos de fabricação. Alguns dos principais algoritmos incluem o k-médias, o agrupamento baseado em densidade, a análise de componentes principais e a incorporação de vizinhos estocásticos distribuídos.

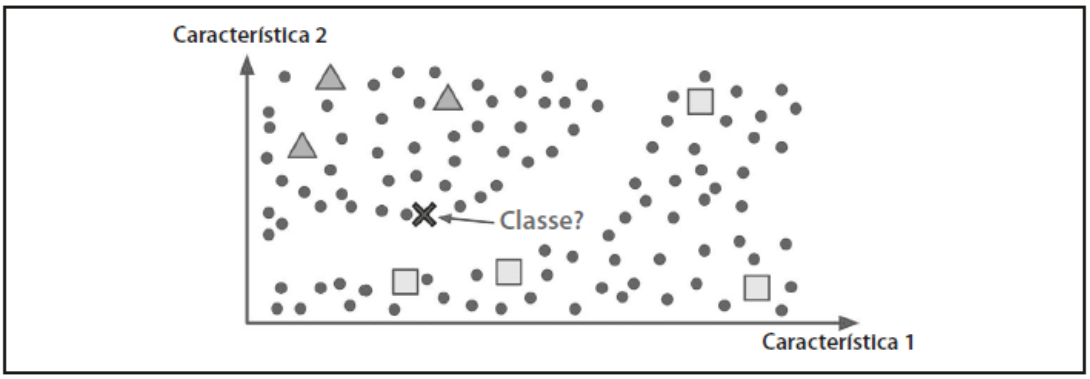
Figura 4 – Clusterização para detecção de grupos distintos



Fonte: Geron (2019)

O aprendizado semissupervisionado envolve dados que são parcialmente rotulados durante o treinamento. Isso encontra aplicação em diversos serviços, como hospedagem de fotos, em que, ao fazer o *upload* das imagens, o aplicativo pode automaticamente reconhecer rostos semelhantes e, posteriormente, solicitar o nome das pessoas. Com apenas um rótulo por pessoa, o sistema pode generalizar para todas as outras. A Figura 5 exemplifica esse cenário, onde existem duas regiões e algumas amostras têm rótulos, permitindo a futura generalização para todo o conjunto de dados. As redes neurais de crenças profundas e máquinas restritas de Boltzmann são alguns dos algoritmos mais clássicos para este sistema de aprendizado.

Figura 5 – Aprendizado semissupervisionado

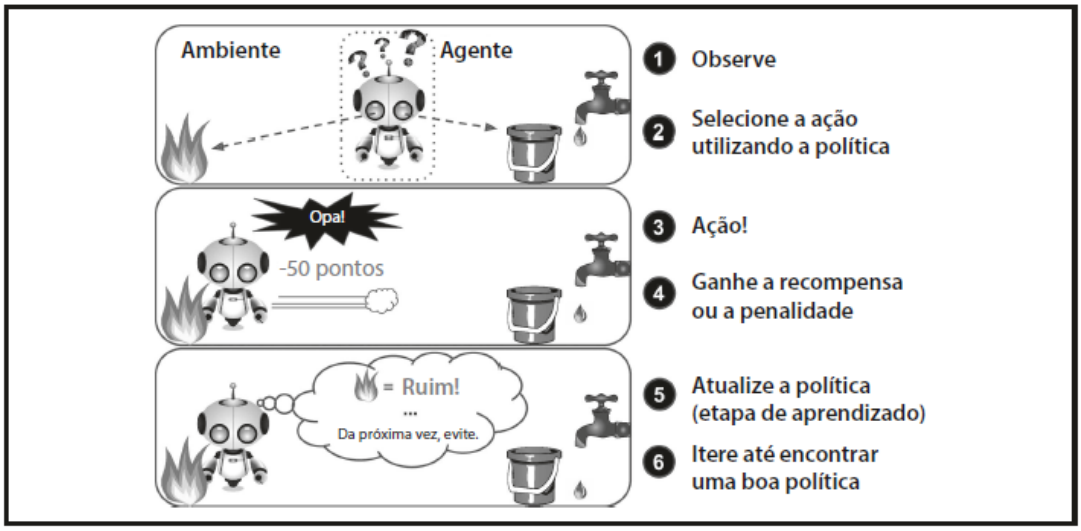


Fonte: Geron (2019)

Por último, aprendizado por reforço envolve agentes inteligentes que observam o ambiente em que estão inseridos e tomam ações para obter recompensas, ou, em casos inadequados, penalidades. Dessa forma, o agente aprende de forma autônoma a melhor política que maximiza o acúmulo de recompensas ao longo do tempo. Na Figura 6, é possível visualizar um exemplo em que um robô aprende a evitar o fogo,

uma vez que isso resulta em penalizações em sua pontuação. Essa abordagem é frequentemente encontrada tanto em aplicações de robótica quanto em jogos inteligentes, como no AlphaGo da DeepMind.

Figura 6 – Aprendizado por reforço



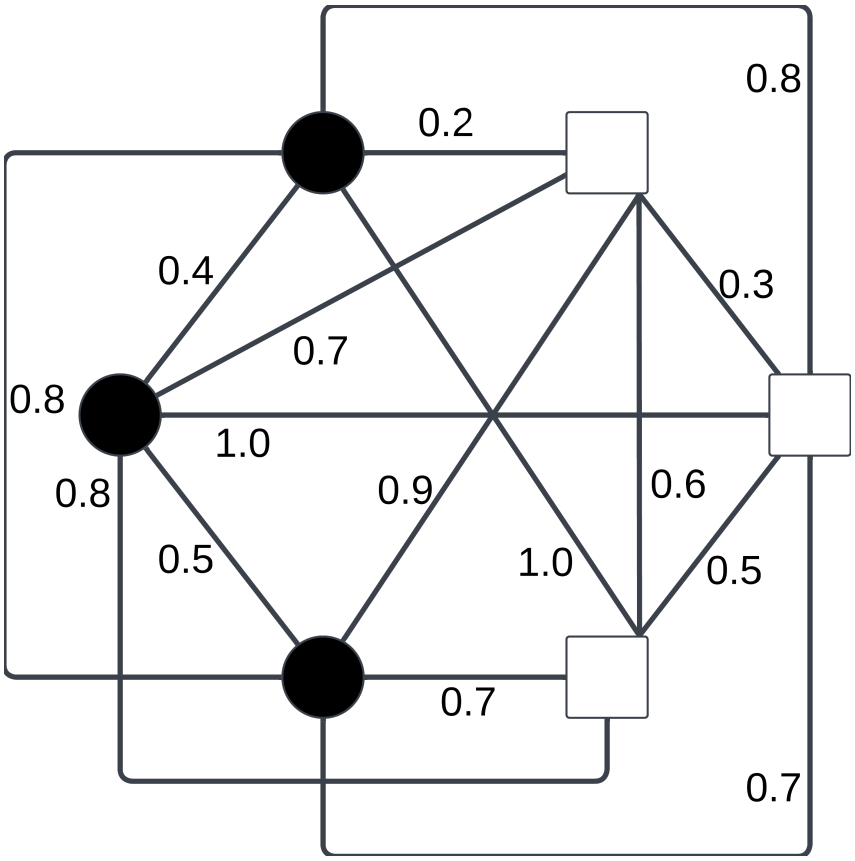
Fonte: Geron (2019)

2.2.1 Classificador Baseado em Floresta de Caminhos Ótimos

Existem diversos algoritmos de aprendizado de máquina, mas neste trabalho, será utilizado um em específico: o classificador *Optimum-Path Forest* (OPF, Floresta de Caminhos Ótimos). O classificador OPF modela o conjunto de treinamento como um grafo e, em seguida, realiza sua partição em uma floresta de caminhos mínimos. Esta técnica é aplicada tanto em cenários de treinamento não supervisionado, conforme proposto inicialmente por Rocha, Falcao e Meloni (2008), quanto em cenários supervisionados, como apresentada por Papa (2008), em que usa tanto grafos completos quanto grafos KNN. Uma análise mais aprofundada do aprendizado supervisionado OPF usando grafos completos é fundamental, já que este trabalho partirá desta técnica como base para a futura construção do módulo.

No início, é necessário representar as amostras do conjunto de treinamento como vértices de um grafo, onde a relação de adjacência é completa, ou seja, o grau destes vértices é máximo. As arestas são ponderadas usando alguma métrica de distância calculada a partir dos vetores de características. A Figura 7 ilustra um exemplo desse grafo completo ponderado criado a partir de um conjunto de treinamento específico que apresenta duas classes. Com o grafo inicial definido, será realizado o procedimento que visa escolher os dados mais representativos de cada classe e estes serão definidos como protótipos.

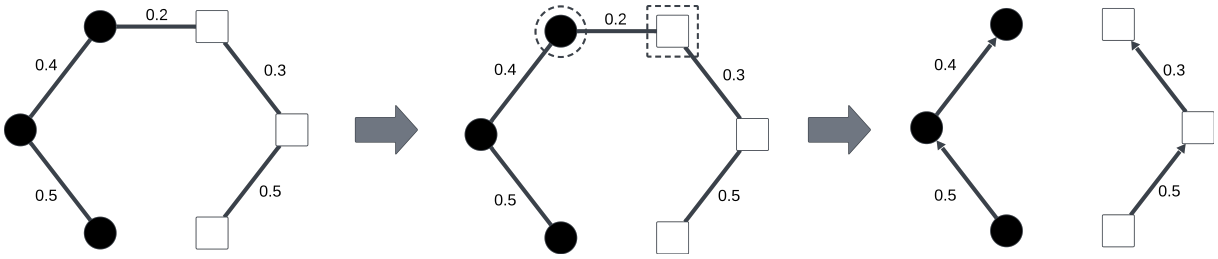
Figura 7 – Conjunto de treinamento modelado como um grafo



Fonte: Adaptado de Papa (2008)

Os protótipos são representados pelos dados situados na região de fronteira entre as classes. Para determiná-los, é empregado o algoritmo da *Minimum Spanning Tree* (MST, Árvore Geradora Mínima), no qual o resultado desse algoritmo é um grafo acíclico composto por arestas não direcionadas e ponderadas, de modo que a soma dessas ponderações seja a menor possível. Após a obtenção da MST, o grafo é percorrido, e os vértices adjacentes com rótulos diferentes são marcados como protótipos, isto é, são identificados como os elementos mais próximos entre classes distintas. A partir disso, estabelece-se a floresta de caminhos ótimos, na qual cada árvore possui como raiz um protótipo indicador de uma classe, marcando o início da fase de treinamento da OPF. Partindo do grafo apresentado na Figura 7, a aplicação do algoritmo MST resulta no primeiro grafo, conforme mostrado pela Figura 8. Em seguida, o percurso pelo grafo é realizado a fim da identificação dos protótipos, selecionando-se um protótipo para cada classe, totalizando duas marcações. Posteriormente, a floresta é estabelecida, cuja representação é visualizado no último grafo desta mesma figura.

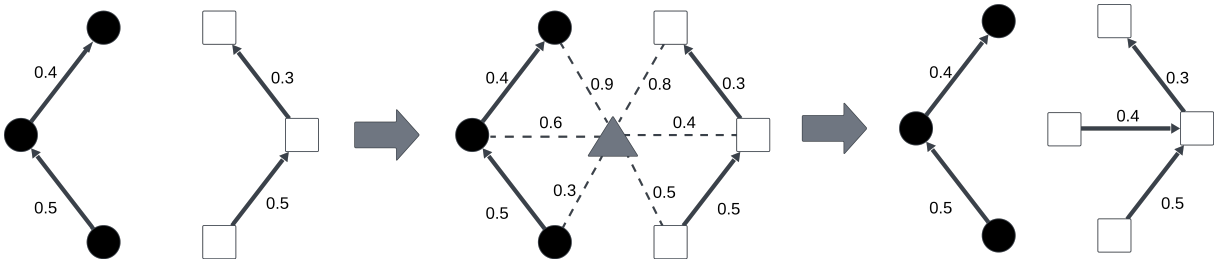
Figura 8 – Etapa de treinamento



Fonte: Adaptado de Papa (2008)

Após a fase de treinamento, procede-se à etapa de classificação utilizando um conjunto de teste distinto ao inicial. Para cada amostra pertencente a este conjunto, calculam-se as distâncias entre ela e as do treino. Em seguida, determina-se o custo de todos os caminhos possíveis que ligam a amostra de teste às raízes das árvores criadas pelo OPF. Identifica-se o caminho ótimo, ou seja, aquele com o menor custo, conectando a amostra de teste à raiz. A amostra de teste é então rotulada com o mesmo rótulo do protótipo associado à raiz do caminho ótimo encontrado. A Figura 9 ilustra o processo de classificação de uma amostra representada por um triângulo. Neste exemplo, as distâncias entre esta amostra e todos os dados de treinamento são calculadas, e o caminho ótimo até os protótipos é estabelecido. Observe que mesmo a aresta $0,3$ seja a menor, foi escolhida a de peso $0,4$ pois esta apresenta menor distância até a raiz da árvore ($0,3+0,5+0,4 > 0,4+0,3$). Portanto, a amostra triângulo é classificada como quadrado, uma vez que ela está mais próxima desta classe.

Figura 9 – Etapa de classificação



Fonte: Adaptado de Papa (2008)

Essa técnica de classificação encontra aplicação em diversas áreas, incluindo a descrição de texturas em imagens (MONTOYA-ZEGARRA et al., 2008), o diagnóstico de doenças laríngeas (PAPA et al., 2008) e a classificação de impressões digitais (MONTOYA-ZEGARRA et al., 2009). Além disso, existem duas bibliotecas amplamente reconhecidas que implementam essa técnica nas linguagens C e Python, as quais serão discutidas em detalhes no próximo capítulo.

2.3 Aprendizado de Máquina Quântico

A computação quântica pode ser empregada no contexto do aprendizado de máquina para resolver problemas complexos, tendo o potencial de melhorar tanto o tempo de treinamento quanto a precisão da predição dos dados. Conforme bem exposto pelos pesquisadores Schuld e Petruccione (2018), existem quatro abordagens para combinar essas duas vertentes computacionais, dependendo da origem dos dados e dos dispositivos que realizam o processamento de informações, que podem ser clássicos ou quânticos. Esta distinção pode ser visualizada na Figura 10.

Figura 10 – Abordagens do aprendizado de máquina quântico

Fonte: Adaptado de Schuld e Petruccione (2018)

A primeira abordagem refere-se aos dados clássicos processados de forma clássica (clássica-clássica), o que corresponde à abordagem convencional do aprendizado de máquina, embora baseada em métodos da informação quântica. A segunda, quântica-clássica, lida com dados gerados em sistemas quânticos, porém processados de maneira clássica, onde os cenários incluem o uso do aprendizado de máquina para auxiliar na computação quântica. A terceira utiliza o processamento quântico nos dados clássicos, clássica-quântica, voltado para mineração de dados e otimização de problemas. Já a última envolve a análise de dados quânticos processados em máquinas quânticas (quântica-quântica).

Considerando a abordagem clássica-quântica, alguns dos problemas de maior relevância são aqueles que envolvem otimização. Entre os protocolos, destacam-se a

computação adiabática e o *quantum annealing* como métodos principais. O *quantum annealing* é um processo que se inicia com a preparação de um estado fundamental composto por n *qubits*, e a partir disso, busca-se a configuração do estado que representa a menor energia do sistema. Essa técnica frequentemente utiliza um conjunto de dados descritos por meio de um hamiltoniano, uma equação matemática que descreve a energia total de um sistema físico, amplamente empregada na mecânica quântica.

Os hamiltonianos e o *quantum annealing* são especialmente aplicáveis em contextos de otimização, permitindo a descrição de problemas como hamiltonianos para, posteriormente, empregar a técnica de *quantum annealing* na busca pela configuração que minimiza o problema modelado. As primeiras implementações desta técnica foram realizadas nos computadores desenvolvidos pela empresa canadense D-Wave. Nestes sistemas, o hamiltoniano é formulado no formato de um problema *Quadratic Unconstrained Binary Optimization* (QUBO, Otimização Binária Quadrática Irrestrita), cuja solução é uma sequência binária de $x_1, \cdots, x_n$ que minimiza a função de energia

$$\sum_{i \leq j=1}^{n} w_{ij} x_i x_j, \tag{2.7}$$

com os coeficientes ou pesos w_{ij}. Vários algoritmos de aprendizado de máquina estão fundamentados em problemas de otimização, especialmente aqueles que se estruturam em grafos, como as redes Bayesianas. Nessas estruturas, é viável representar cada aresta como presente ou ausente. Isso permite a utilização dos *qubits* $|0\rangle$ e $|1\rangle$ para expressar a conectividade por meio de uma sequência binária. Além destas redes, problemas como corte máximo, programação linear binária, coloração de grafos, problema da mochila e do caixeiro viajante podem ser modelados utilizando essa abordagem.

As técnicas mais comumente utilizadas para resolver o problema QUBO incluem o *Quantum Approximate Optimization Algorithm* (QAOA, Algoritmo de Otimização Aproximada), desenvolvido por Farhi, Goldstone e Gutmann (2014), e o *Feedback-Based Quantum Optmization* (FALQON, Algoritmo Quântico Baseado em Feedback), proposto por Magann et al. (2022). Neste projeto, o foco será o algoritmo FALQON, pois este apresenta uma vantagem sobre o primeiro algoritmo, o qual é a garantia de convergência da solução ao evitar possíveis travamentos em mínimos locais.

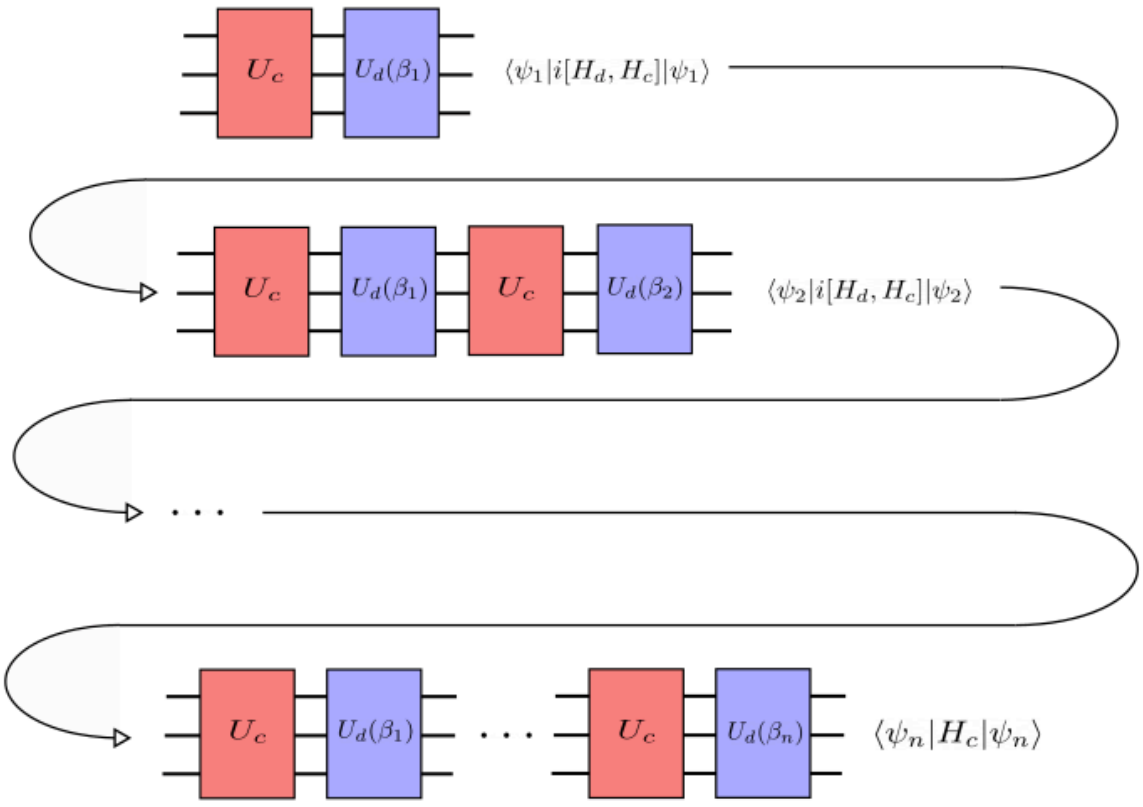
Conforme a pesquisa realizada por Magann et al. (2022), o algoritmo FALQON é empregado na resolução de problemas de otimização. Este algoritmo quântico é baseado na teoria ótima de caminho e no teorema de Trotter-Suzuki, com sua implementação feita por meio de circuitos quânticos variacionais, ou seja, circuitos parametrizados. Considere um sistema descrito por um hamiltoniano H_c, cuja representação matricial é diagonal e nela contém todas as configurações que este sistema pode assumir. O teorema de Trotter-Suzuki afirma a evolução de um sistema quântico pode ser apro-

ximada por uma sequência de aproximações quânticas mais simples. Sendo assim é possível reescrever o hamiltoniano inicial em $H_c = H_1 + H_2$ pois a evolução do sistema será dada pelas transformações temporais feitas em H_1 e H_2. Dado um estado inicial aleatório $|\psi_0\rangle$, o operador de evolução temporal $U_c = e^{-iH_c\Delta t}$, o momento angular total na direção x como $H_d = \sum \sigma_x^i$, e a dinâmica temporal em H_d como $U_d(\beta_k) = e^{-iH_d\beta_k\Delta t}$, em que β_k é calculado iterativamente, tem-se o seguinte algoritmo recursivo para cada passo k :

1. Preparar o estado $|\psi_k\rangle = U_d(\beta_k)U_c \cdots U_d(\beta_1)U_c|\psi_0\rangle$;
2. Medir o valor esperado $A_k = i\langle\psi_k|H_dH_c - H_cH_d|\psi_k\rangle$; e
3. Calcular $\beta_{k+1} = -A_k$

A Figura 11 ilustra o processo recursivo do algoritmo FALQON. Após a execução das k iterações, a saída deste algoritmo será a aproximação do estado fundamental $|\psi_k\rangle$ com a menor configuração de energia.

Figura 11 – Circuito recursivo FALQON



Fonte: (WAKEHAM; CERONI, 2021)

3 Metodologia

O presente capítulo abordará o processo de construção do módulo quântico, detalhando os recursos, técnicas e o procedimento utilizado para desenvolvê-lo.

3.1 Ferramentas

Os materiais empregados no desenvolvimento do módulo abrangem tanto o aspecto do software quanto o do hardware. No que diz respeito ao primeiro, foram considerados a linguagem de programação, o ambiente de desenvolvimento, o sistema operacional e o gerenciador de ambientes virtuais. No segundo, os aspectos físicos da máquina utilizada para o desenvolvimento e experimentação. Considerando este contexto, escolheram-se as seguintes ferramentas:

- ***Software***
  - Sistema Operacional: Windows 11 Home;
  - Ambiente de Desenvolvimento Integrado: Visual Studio Code;
  - Gerenciador de Ambiente Virtual: Pipenv;
  - Linguagem de Programação: Python na versão 3.11;
  - Bibliotecas principais: OPFython na versão 1.0.12, Qiskit na versão 0.43.0 e Qutip na versão 4.7.1; e
  - Base de dados: Boat (KUNCHEVA, 2005).
- ***Hardware***
  - Notebook pessoal Acer Aspire 3 cujas especificações principais estão indicadas no quadro a seguir:

Quadro 1 – Especificações Técnicas do Acer Aspire 3

Marca	Acer
Modelo	Aspire A315-42G
Processador	AMD Ryzen 5 3500U 2,10 GHz
Armazenamento Interno	SSD de 480 GB e HD de 1 TB
Memória RAM	16,0 GB
Placa de vídeo integrada	Radeon Vega 8 Graphics
Placa de vídeo dedicada	Radeon 540X

Fonte: Elaborado pela autora.

A seguir, descreveremos os principais recursos, como o gerenciador de ambiente virtual e as bibliotecas (OPFython, Qiskit e Qutip). Além disso, é fundamental destacar o conjunto de dados utilizada nos processos de treinamento e teste deste projeto.

3.1.1 Pipenv

O Pipenv é uma ferramenta de código aberto desenvolvida por Reitz (2017) com o propósito de gerenciar ambientes virtuais para projetos em Python. Além de criar e administrar esses ambientes virtuais, o Pipenv também permite o controle das dependências do projeto. Esta ferramenta consolida as funcionalidades do *pip* (gerenciador de pacotes) e do *virtualenv* (gerenciador de ambientes virtuais) em uma única interface de linha de comando, simplificando substancialmente o processo de desenvolvimento.

A implementação do Pipenv demonstrou sua relevância ao possibilitar o isolamento das dependências, garantindo que não haja conflitos com outros projetos. Essa abordagem é particularmente crucial, considerando que a linguagem Python e suas bibliotecas são instaladas globalmente ao nível de cada usuário nos computadores. Adicionalmente, o uso do Pipenv simplifica a replicação do projeto em diferentes máquinas, uma vez que fornece um arquivo contendo todas as dependências necessárias para seu funcionamento. Isso torna a colaboração e a execução em diferentes ambientes mais eficazes e reduz possíveis incompatibilidades, promovendo a reprodutibilidade e a escalabilidade do desenvolvimento.

3.1.2 OPFython

O OPFython, desenvolvido por Rosa e Papa (2021), é uma biblioteca de código aberto com o propósito de simplificar a implementação do algoritmo OPF para tarefas de classificação de dados. Além de oferecer uma implementação otimizada do algoritmo, esta biblioteca destaca-se pela sua facilidade de uso e versatilidade na manipulação de diferentes tipos de dados. Compatível com Python a partir da versão 3.6, o OPFython é composto por seis estruturas fundamentais, que incluem:

- *core*: núcleo da biblioteca, em que é estabelecido tanto a estrutura fundamental do algoritmo quanto ao fornecimento das ferramentas que facilitarão a construção de outros componentes;
- *math*: implementação matemática em baixo nível, incluindo cálculos de distância e funções de randomização;
- *stream*: responsabiliza pela manipulação dos dados, desde o carregamento inicial até a separação destes em conjuntos de treinamento e testes;

- *subgraphs*: implementação de algumas variações de modelos de grafos, como o completo e o KNN;
- *models*: abrange os modelos de classificação, incluindo o supervisionado, o semi-supervisionado e o não supervisionado; e
- *util*: funções utilitárias em comum a todos os módulos da biblioteca a fim de evitar repetições desnecessárias.

Vale ressaltar que esta biblioteca foi construída com base da LibOPF, desenvolvida por Papa (2015) na linguagem de programação C.

3.1.3 Qiskit

O Qiskit é um conjunto de desenvolvimento de software desenvolvida pela empresa IBM em 2017 para programação em computadores quânticos (QISKIT CONTRIBUTORS, 2023a). Escrito em Python, esta ferramenta pode ser executada com uso de primitivas nativas em simuladores locais, simuladores em nuvem e hardware quântico real. Atualmente encontra-se disponível na versão 0.44.0.

O Qiskit consistia inicialmente em quatro elementos: Terra, Aer, Ignis e Aqua. O Qiskit Terra forma a base da biblioteca, permitindo o desenvolvimento e otimização de programas quânticos, bem como a administração de execuções em lote de experimentos por acesso remoto. Por sua vez, o Qiskit Aer oferece três simuladores de alto desempenho (*QasmSimulator*, *StatevectorSimulator*, *UnitarySimulator*). O Qiskit Ignis é responsável pela construção de códigos de correção de erros. Por fim, no Qiskit Aqua, estão contidos os algoritmos quânticos para a construção de aplicações quânticas.

Devido ao crescimento e evolução da biblioteca, o pacote Qiskit Aqua foi fragmentado em *qiskit-optimization*, *qiskit-nature*, *qiskit-machine-learning* e *qiskit-finance*. Além disso, o Qiskit Ignis passou a ser chamado de Qiskit Experiments, com o propósito de definir e monitorar experimentos com base nas funcionalidades principais do Qiskit Terra (QISKIT CONTRIBUTORS, 2023b).

3.1.4 QuTiP

Os pesquisadores Johansson, Nation e Nori (2012) projetaram uma ferramenta de código aberto para simulações numéricas de sistemas de mecânica quântica de pequeno e médio porte. Essa biblioteca oferece uma série de ferramentas para simular a evolução temporal de sistemas quânticos abertos e fechados, bem como a manipulação de operadores quânticos e funções de onda.

Amplamente utilizada em pesquisas e aplicações acadêmicas que envolvem sistemas quânticos, incluindo física quântica, óptica quântica, computação quântica

e outras áreas afins. Escrita em Python, a biblioteca combina recursos dos pacotes NumPy, SciPy e Matplotlib, permitindo o trabalho com matrizes esparsas e a representação visual de dados. Isso torna o desenvolvimento e a análise de sistemas quânticos mais acessíveis, claros e eficazes para pesquisadores e estudantes.

3.1.5 Base de Dados

Nos projetos de aprendizado de máquina, os conjuntos de dados desempenham um papel crucial ao permitir o treinamento e a validação dos modelos desenvolvidos. Através da disponibilidade de um conjunto de dados que não apenas descreve o problema a ser abordado, mas também representa uma situação real ou simulada, torna-se viável a construção de modelos preditivos e analíticos. No contexto deste projeto, foi adotado o conjunto de dados conhecido como Boat, elaborado por Kuncheva (2005), composto por 100 amostras distribuídas em três classes distintas, cada uma caracterizada por duas variáveis. A seleção desse conjunto foi motivada pela sua facilidade de manipulação e visualização, além de ter sido utilizada por Rosa e Papa (2021) como parte dos testes para a biblioteca OPFython. Portanto, a disponibilidade, estrutura e aplicabilidade prévia na avaliação de ferramentas similares tornaram-no uma escolha conveniente para as necessidades do projeto.

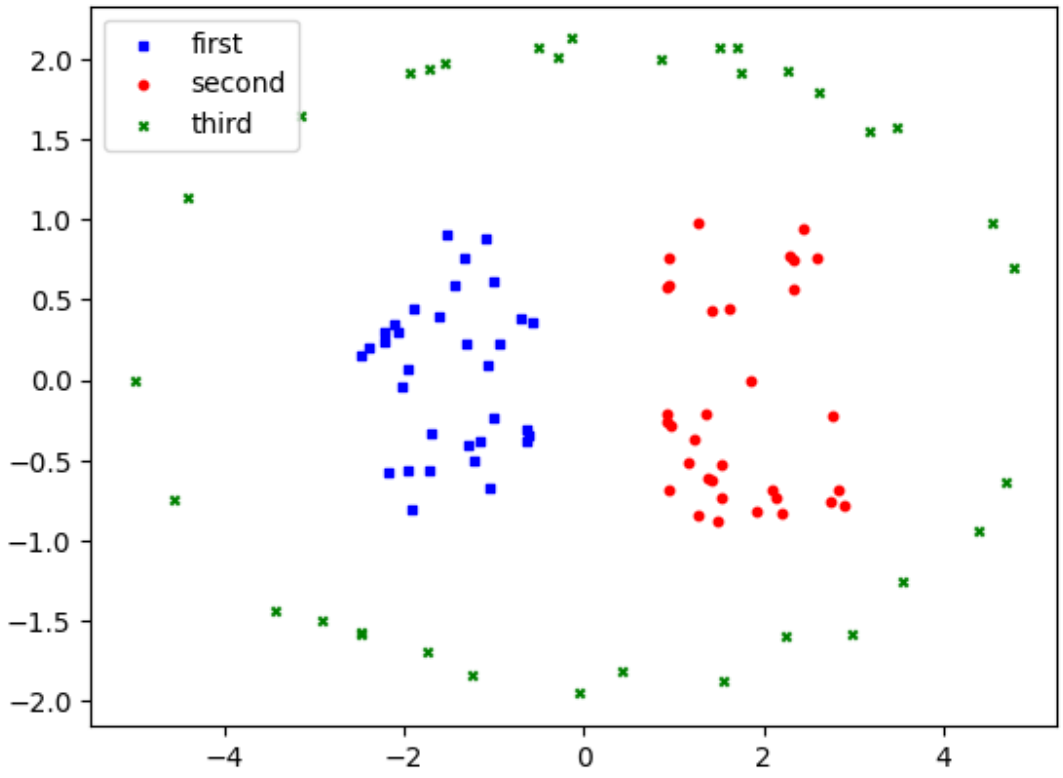
Tabela 1 – Conjunto de dados

#	Classe	Característica 1	Característica 2
0.0000000000000000e+00	0.0000000000000000e+00	-1.528821945190429688e+00	9.044460058212280273e-01
1.0000000000000000e+00	0.0000000000000000e+00	-2.036158084869384766e+00	-4.268300160765647888e-02

Fonte: Adaptado de Kuncheva (2005).

No conjunto de dados apresentando na Tabela 1, é notável que os dados estão dispostos em um formato numérico, onde a primeira coluna refere-se ao número da característica, indicando um aspecto de numeração das linhas. A segunda coluna corresponde à classe, como o rótulo que identifica a categoria a que cada instância pertence. As colunas subsequentes, por sua vez, representam as características associadas a cada instância de dados. Esse formato tabular escrito em um arquivo de texto é organizado permitindo a clara estruturação dos dados. Desta forma, é possível visualizar a base de dados com foco nas classes conforme é expresso na Figura 12.

Figura 12 – Base de dados Boat com suas três classes em evidência no espaço



Fonte: Elaborada pela autora.

3.2 Abordagem Proposta

O algoritmo OPF classifica os dados por meio de uma floresta de caminhos ótimos. Como mencionado anteriormente (subseção 2.2.1), a etapa inicial do treinamento consiste em identificar as amostras representantes de cada classe por meio do grafo obtido pelo problema da MST. Esse problema pode ser interpretado como uma questão de otimização em um grafo, viabilizando o uso de técnicas de otimização quântica. Entretanto, diante do desafio de adaptar o problema da MST ao contexto quântico, uma abordagem alternativa foi adotada: a criação de um ciclo hamiltoniano fechado, exemplificado pelo *Traveling Salesman Problem* (TSP, Problema do Caixeiro Viajante).

Na otimização quântica, é necessário formular os problemas como hamiltonianos usando a técnica QUBO para na sequência serem resolvidos por algoritmos específicos, como o QAOA ou o FALQON; diante disso, o módulo deste projeto implementou estas técnicas. Esta seção abordará a modelagem do problema TSP para um problema QUBO e discutirá a implementação do módulo quântico dentro da OPF.

3.2.1 Modelagem do problema do caixeiro viajante

Suponha um conjunto de cidades com suas respectivas distâncias. Dada uma cidade de origem e outra de destino, busca-se encontrar a rota de menor custo que visite todas as cidades exatamente uma vez e que retorne a origem. Esse desafio é conhecido na ciência da computação como TSP e é classificado como um problema NP-difícil, o que significa não haver um algoritmo conhecido que o resolva de maneira ótima em tempo polinomial. O TSP é considerado um problema de otimização binária, o que o torna passível de modelagem no contexto quântico.

Considere um grafo completo no qual as arestas w_{ij} representam as distâncias entre os vértices i e j. O TSP pode ser descrito por meio de uma sequência binária que indica se a aresta está presente ($x = 1$) ou não ($x = 0$) na solução do grafo. Assim, é possível formular matematicamente este problema como a soma dos pesos das arestas presentes, representada por $\sum_i \sum_{j>i} w_{ij}x_{ij}$, e suas restrições:

1. Número de arestas deve se igualar ao de vértices n, $\left(\sum_i \sum_{j>i} x_{ij} - n\right)^2$; e
2. Cada vértices deverá conter duas arestas, ou seja, $\sum_i \left(\sum_{j\neq i} x_{ij} - 2\right)^2$.

Diante disso, é possível formular a Equação abaixo como a função custo que descreve o problema em questão,

$$C(x) = \sum_i \sum_{j>i} w_{ij}x_{ij} + \left(\sum_i \sum_{j>i} x_{ij} - n\right)^2 + \left[\sum_i \left(\sum_{j\neq i} x_{ij} - 2\right)^2\right]. \tag{3.1}$$

Para representar a equação acima como um hamiltoniano, é necessário reescrever a variável x_{ij} como $\tilde{Z}_{ij}$, em que $\tilde{Z}_{ij} = \frac{1-Z}{2}$. Nesse contexto, Z é um operador quântico que introduz uma fase negativa quando o *qubit* é igual a $|1\rangle$. A transformação aplicada pelo operador $\tilde{Z}_{ij}$ nos estados $|0\rangle$ e $|1\rangle$ é definida abaixo:

$$\begin{aligned} \tilde{Z}_{ij}|0\rangle &= 0|0\rangle, \\ \tilde{Z}_{ij}|1\rangle &= 1|1\rangle. \end{aligned} \tag{3.2}$$

Portanto, a codificação da função custo para o hamiltoniano que descreve a dinâmica do problema TSP é dada por:

$$H_c = \sum_i \sum_{j>i} w_{ij}\tilde{Z}_{ij} + P_1\left(\sum_i \sum_{j>i} \tilde{Z}_{ij} - n\right)^2 + P_2\left[\sum_i \left(\sum_{j\neq i} \tilde{Z}_{ij} - 2\right)^2\right]. \tag{3.3}$$

O hamiltoniano H_c apresentado acima descreverá todas as configurações de grafo e aquelas que não atenderem as restrições mencionadas anteriormente serão penalizadas para que estes não sejam soluções na hora de procurar o estado de menor energia (JORDAN, 2018).

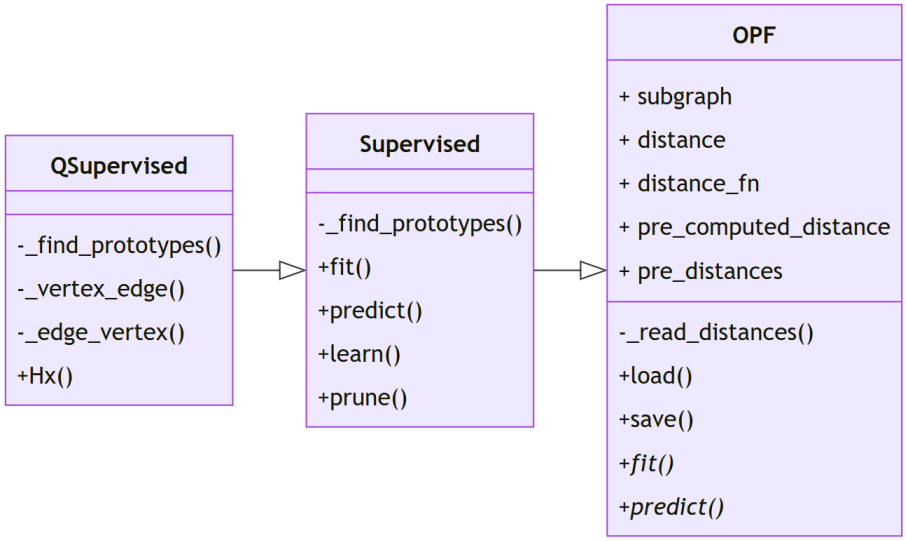
3.2.2 Arquitetura Geral

O termo “arquitetura de software” refere-se à estrutura geral de uma aplicação, englobando a organização de seus componentes, suas interações e a estrutura de dados. Essa arquitetura é crucial para visualizar e interpretar as funcionalidades do programa, além de direcionar o desenvolvimento e estabelecer padrões para garantir clareza e coesão (PRESSMAN; MAXIM, 2021).

O objetivo do módulo quântico desenvolvido é realizar a seleção dos protótipos durante a etapa de treinamento, utilizando os princípios da computação quântica. Para alcançar essa meta, foi necessário adaptar a biblioteca OPFython, construída seguindo os conceitos de programação orientada a objetos, para inserir este novo módulo. Os modelos de aprendizado são um componente importante desta biblioteca e cada um tem um treinamento diferente, conforme listado na seção 2.2. Diante disso, este projeto trabalhou sobre o modelo supervisionado com grafo completo.

A fim de integrar esse módulo quântico à estrutura da biblioteca OPFython, foi adicionada uma nova classe que herda os métodos do modelo supervisionado, estendendo a classe *SupervisedOPF*. A Figura 13 ilustra a representação da classe recém-criada, chamada *QSupervisedOPF*, destinada a sobrescrever o método de busca dos protótipos *_find_prototypes* presente na classe *SupervisedOPF*. Em outros termos, o diagrama de classes representado na Figura 13 destaca de forma mais clara a arquitetura da aplicação.

Figura 13 – Diagrama de classes simplificado da aplicação



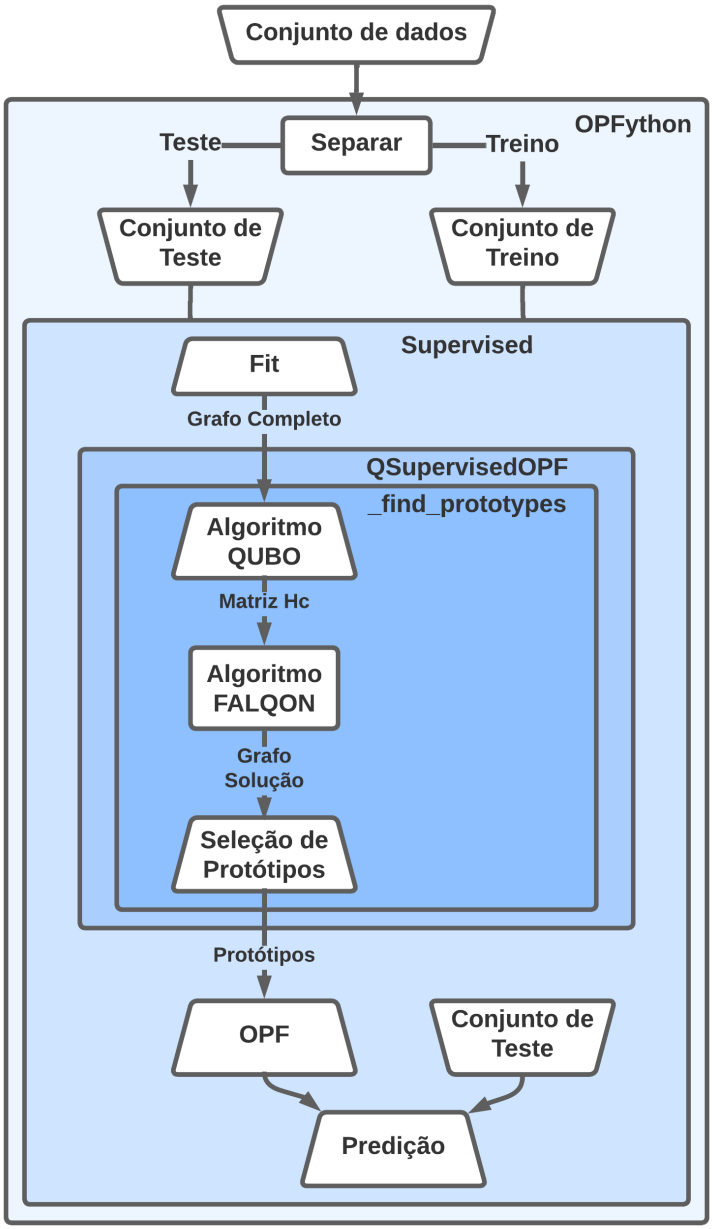
Fonte: Elaborada pela autora.

A Figura 14 ilustra o fluxograma da aplicação e enfatiza os procedimentos que o método sobrescrito *_find_prototypes* realiza, os quais podem ser simplificados em

quatro etapas que esta biblioteca realiza, sendo elas:

1. Calcular e normalizar o peso das arestas com base na distância euclidiana;
2. Montar hamiltoniano que descreva o problema TSP;
3. Obter o grafo de menor custo que resolva o problema TSP usando o algoritmo FALQON; e
4. Percorrer o grafo solução e marcar como protótipos as amostras adjacentes de classes diferentes.

Figura 14 – Fluxograma da aplicação usando o módulo desenvolvido



Fonte: Elaborada pela autora.

3.2.3 Desenvolvimento da aplicação

O processo inicial consistiu na configuração do ambiente virtual Pipenv, visando isolar a aplicação e prevenir potenciais conflitos entre as diversas versões de bibliotecas utilizadas. Posteriormente, foi instalada a biblioteca OPFython, seguida pela criação de dois arquivos: o primeiro correspondente ao programa principal e o segundo, ao módulo de seleção de protótipos, implementado de forma quântica.

O programa principal carrega o conjunto de dados e realiza a divisão em subconjuntos de treino e teste. Em seguida, aplica a abordagem clássica de seleção de protótipos para treinamento, realiza a previsão com o conjunto de testes e calcula a acurácia. Por fim, repete o mesmo procedimento, porém por meio da abordagem quântica para a seleção de protótipos. Este processo é exibido na Figura 15.

Figura 15 – Fluxo de controle do programa principal

```
# Dividir os dados em conjunto de treinamento e conjunto de testes
X_treino, X_teste, Y_treino, Y_teste = s.split(X, Y, percentage=0.5, random_state=1)

# Criar uma instância de SupervisedOPF
opf = SupervisedOPF(distance="log_squared_euclidean", pre_computed_distance=None)

opf.fit(X_treino, Y_treino)                # Treino
preds = opf.predict(X_teste)               # Predição
acc = g.opf_accuracy(Y_teste, preds)       # Acurácia
print(f"Accuracy: {acc}")

# Criar uma instância de QSupervisedOPF
q_opf = QSupervisedOPF(distance="log_squared_euclidean", pre_computed_distance=None)

q_opf.fit(X_treino, Y_treino)              # Treino
preds_q = q_opf.predict(X_teste)           # Predição
acc_q = g.opf_accuracy(Y_teste, preds_q)   # Acurácia
print(f"Accuracy Q: {acc_q}")
```

Fonte: Elaborada pela autora.

No desenvolvimento da nova abordagem de treinamento, o processo inicia-se com a depuração da biblioteca OPFython para compreender sua estrutura e integrar o módulo de forma mais adequada, visando aproveitar as funcionalidades já presentes. Durante esta etapa, foi constatado que a abordagem mais eficaz envolvia a criação de uma classe denominada *QSupervisedOPF*, a qual sobrescreveu o método *_find_prototypes*. Esse procedimento baseado nos conceitos de programação orientada a objetos permitiu herdar a estrutura do grafo de treinamento, facilitando a escolha das amostras e o retorno ao método de construção da floresta de caminhos ótimos.

Conforme mencionado anteriormente, a primeira etapa compreende o cálculo dos pesos das arestas. Esse método seguiu o procedimento padrão da OPF clássica, o

qual calcula os pesos das arestas com base na distância euclidiana. Com o vetor de pesos gerados, foi realizada a normalização, a fim de facilitar a convergência na obtenção do estado de menor energia. Em seguida, procedeu-se à montagem do hamiltoniano que descreve o TSP, por meio de operações matriciais e funções provenientes da biblioteca Qutip. O trecho de código que executa esse procedimento está representado na Figura 16. Os pesos $P1$ e $P2$ foram escolhidos manualmente e penalizam da melhor forma as restrições do problema a fim de encontrar o estado do problema.

Figura 16 – Código hamiltoniano

```
# Construção do Hamiltoniano Hc

# Somatório dos pesos das arestas para cada configuração de grafo
for i in range(0, n_edge):
    if i == 0:
        Hc = weights[i]*zI[i]
    else:
        Hc = Hc + weights[i]*zI[i]

# Restrição 1: Número de arestas deve ser igual ao de vértices
P1 = 9
R1 = 0
for k in range(0, n_edge):
    R1 += zI[k]

R1 = P1*(R1 - n_nodes*Id)**2

# Restrição 2: Duas arestas por vértice
P2 = 7
R2 = 0
for k in range(1, n_nodes+1):
    somatorio = 0
    for l in range(1, n_nodes+1):
        if k != l:
            edge = self._vertex_edge(k,l,n_nodes)
            somatorio += zI[edge]
    somatorio -= 2*Id
    R2 += P2 * (somatorio**2)

# Adicionar as restrições na função Hc
Hc += R1 + R2
```

Fonte: Elaborada pela autora.

Após a montagem do hamiltoniano, a implementação da técnica FALQON para otimizar problemas quânticos tornou-se necessária. A abordagem adotada para este algoritmo foi direta e está intimamente ligada ao procedimento mencionado seção 2.3. O trecho principal, ilustrado na Figura 17, destaca a obtenção da configuração final

do sistema após 10.000 interações a partir da matriz hamiltoniana. Antes de atingir o estado de menor energia, é calculada a probabilidade de ocorrência para cada configuração do grafo, e aquela com a maior probabilidade indica o modelo do grafo de menor energia.

Figura 17 – Código FALQON

```
# Resolver o problema QUBO com o algoritmo FALQON
dt=0.002
Psi = np.ones((2**n_edge,1))/np.sqrt(2**n_edge) # Estado |+>
lam=0.0
Sx = self.Hx(lam, n_edge)
beta = -1j*np.conjugate(Psi).T@(Sx@Hc-Hc@Sx)@Psi
resp=[]
for i in range(0,10000):
    Ux = la.expm(-1j*beta*Sx*dt)
    Uc = la.expm(-1j*Hc*dt)
    Psi = Ux@Uc@Psi
    beta = -1j*np.conjugate(Psi).T@(Sx@Hc-Hc@Sx)@Psi
    aux = np.conjugate(Psi).T@Hc@Psi
    resp.append(aux[0][0].real)

# Cálculo das probabilidades associadas a cada estado do grafo
probs = [abs(i)**2 for i in qi.Statevector(Psi)]
```

Fonte: Elaborada pela autora.

O último procedimento deste método envolve a seleção dos protótipos a partir do ciclo hamiltoniano fechado obtido pela etapa anterior. O procedimento de seleção de protótipos segue a mesma abordagem adotada no modelo original. Em essência, o algoritmo percorre cada vértice do grafo e escolhe como protótipos aqueles que pertencem a classes distintas. Uma vez que a seleção dos protótipos é concluída, o algoritmo avança para a próxima fase do treinamento, que consiste na montagem da floresta de caminhos mínimos. Essa etapa é fundamental para a construção do modelo, tendo sido realizada com base nos protótipos previamente selecionados. Após a determinação dessa floresta, o processo de treinamento do algoritmo é encerrado, e o modelo está pronto para ser testado com o conjunto de dados teste.

3.2.4 Dificuldade no desenvolvimento

A elaboração do módulo enfrentou desafios distintos. O primeiro obstáculo concentrou-se na manipulação de matrizes de grande escala, representando um desafio técnico significativo. Já a segunda dificuldade esteve relacionada à simulação do algoritmo

FALQON utilizando a biblioteca Qiskit, demandando esforços adicionais para manusear este recurso.

Durante a construção do hamiltoniano utilizando a formulação QUBO para o problema TSP, os *qubits* são encarregados de representar as arestas do grafo. Isso implica que um grafo com n arestas necessita de n qubits para uma representação adequada no contexto quântico. Esta abordagem traz desafios de ordem computacional, especialmente ao determinar as dimensões das matrizes que descrevem os operadores utilizados no referido hamiltoniano. Ao representar esses operadores, são requeridas n matrizes de ordem 2^n, onde n representa o número de arestas presentes no grafo. Por exemplo, considerando um conjunto de treinamento com quatro amostras que utiliza o modelo de grafo completo, há seis arestas, resultando em um operador com dimensões de $6 \times 64 \times 64$ (seis matrizes quadráticas de ordem 64). Além disso, conforme o número de arestas aumenta, a necessidade de memória para armazenar essas matrizes cresce exponencialmente, dificultando o treinamento de modelos com conjuntos de dados extensos.

O segundo desafio relaciona-se às simulações quânticas utilizando os simuladores oferecidos pela biblioteca Qiskit. Foi observado que o algoritmo FALQON é iterativo, no qual cada iteração k depende da medição realizada em A_k, que contribui para a determinação do valor de β_{k+1}. A descrição dessa medição requer uma série de métodos numéricos complexos, exigindo um conhecimento mais aprofundado no tema, além de dificultar a implementação necessária para os simuladores poderem processar essa etapa do algoritmo de maneira adequada. Desta forma, foi implementado o FALQON simplificado.

4 Experimentação e Análise dos Resultados

Neste capítulo, serão abordados os experimentos conduzidos, os resultados obtidos e uma análise com relação ao problema descrito.

4.1 Experimentos e Resultados Obtidos

Com o intuito de avaliar a eficácia do módulo quântico, foram conduzidos dois experimentos, cada um utilizando conjuntos distintos de treinamento contendo diferentes quantidades de amostras. O foco principal desses experimentos foi verificar se o algoritmo FALQON convergia para o estado de menor energia, e se este estado representava adequadamente o grafo solução do problema TSP.

Partindo da base de dados Boat, originalmente elaborada por Kuncheva (2005), a experimentação deste projeto envolveu a fragmentação desse conjunto de 100 amostras em dois cenários distintos. Cada cenário de experimentação apresentou uma quantidade x de amostras que posteriormente foram divididas em dois subconjuntos distintos, sendo a primeira metade destinada ao treinamento e a segunda metade aos testes. Devido às limitações de memória RAM e à manipulação de grandes matrizes, os conjuntos de dados utilizados na experimentação foram de tamanho reduzido.

A comparação entre os dois modelos, quântico e tradicional, foram baseadas na acurácia que a própria biblioteca OPFython implementa. Esta acurácia, varia entre zero e um, considerando não apenas as previsões corretas, mas também a distribuição de erros por classe, considerando a distribuição desigual de exemplos por classe.

A seguir, são apresentados os experimentos, juntamente com seus respectivos resultados obtidos.

4.1.1 Experimento 01

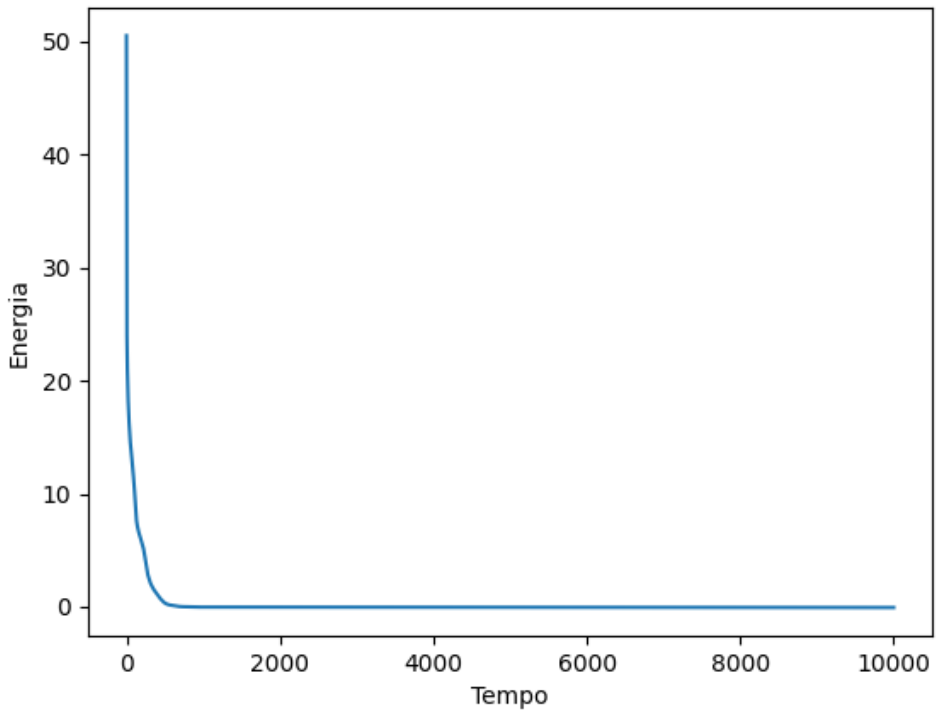
No primeiro cenário, o conjunto de dados foi composto por oito amostras, como representado na Tabela 2. Durante o treinamento, um grafo com quatro vértices foi construído, tendo sido selecionados dois protótipos, uma vez que o conjunto de dados envolvia duas classes distintas.

Tabela 2 – Base de dados com oito amostras

Identificação	Classe	Característica 1	Característica 2
1	0	2.898	-7.795
2	0	2.145	-7.348
3	0	1.520	-7.304
4	0	1.346	-2.069
5	1	-1.721	1.940
6	1	-2.483	-1.567
7	1	-3.155	1.641
8	1	-1.744	-1.697

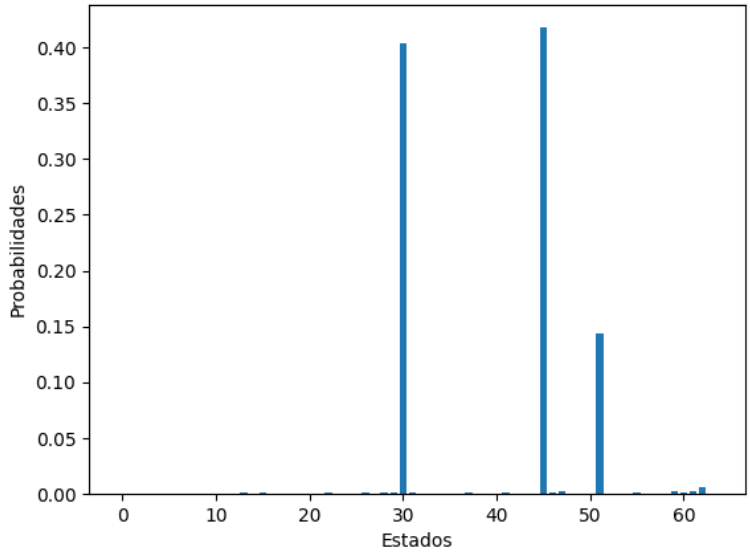
Após realizar o primeiro cenário de experimentação, aquele que apresenta quatro amostras de treinamento, foi possível observar que a solução do problema TSP foi obtida, uma vez que a energia do sistema decai ao longo das iterações (tempo), como demostra o gráfico da Figura 18. Além disso, a probabilidade associada a cada modelo de grafo é indicado na Figura 19, com o estado que apresenta a maior probabilidade associada representando o grafo de custo mínimo que atende as restrições do problema. Vale destacar que tanto o modelo clássico quanto o modelo quântico obtiveram uma acurácia de 1,0 (ou 100%).

Figura 18 – Gráfico de energia para o experimento 1



Fonte: Elaborada pela autora.

Figura 19 – Probabilidade associada a cada estado para o experimento 1



Fonte: Elaborada pela autora.

4.1.2 Experimento 02

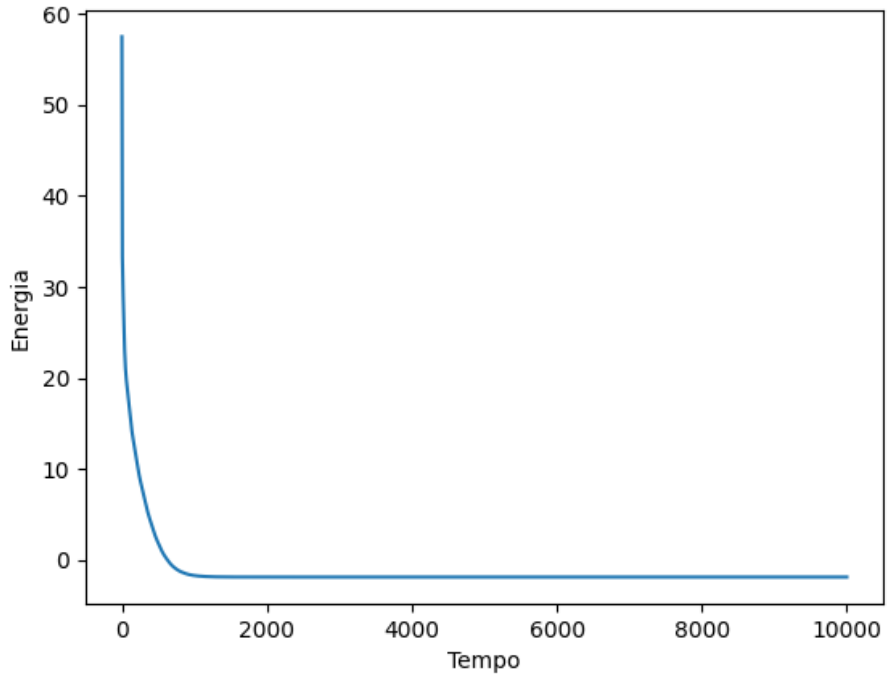
Já no segundo cenário, o conjunto de dados era constituído por dez amostras, conforme mostrado na Tabela 3.Durante o treinamento, um grafo com cinco vértices foi criado e novamente dois protótipos foram escolhidos, devido à presença de duas classes distintas no conjunto de dados.

Tabela 3 – Base de dados com dez amostras

Identificação	Classe	Característica 1	Característica 2
1	0	-1.091	8.868
2	0	-1.336	7.603
3	0	-1.000	-2.407
4	0	-5.829	3.597
5	0	-1.713	-5.692
6	1	-1.224	-4.974
7	1	-3.002	2.008
8	1	1.559	-1.873
9	1	-1.248	-1.839
10	1	8.488	2.003

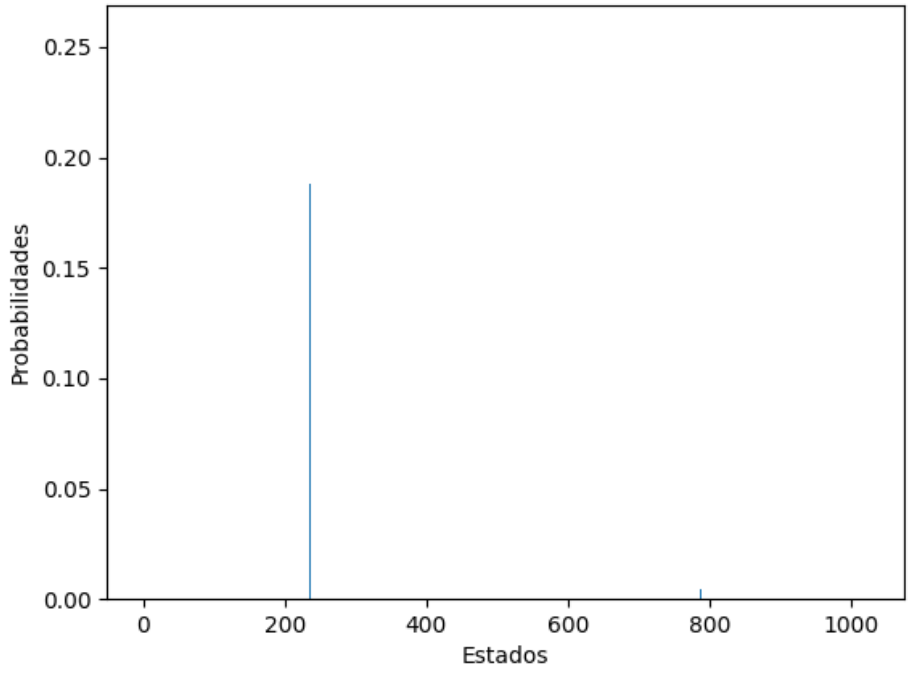
Já os resultados obtidos para o segundo cenário de testes são observados nas Figuras 20 (gráfico de energia) e 21 (gráfico de probabilidades). Quanto a acurácia para este cenário, nos dois modelos, clássico e quântico, foi de 0,5 (ou 50%).

Figura 20 – Gráfico de energia para o experimento 2



Fonte: Elaborada pela autora.

Figura 21 – Probabilidade associada a cada estado para o experimento 2



Fonte: Elaborada pela autora.

4.2 Discussões

Com base nos dois experimentos conduzidos, observou-se que em ambos os cenários de teste foi alcançado o estado de menor energia conforme as figuras 19 e 21. Desta forma, foi comprovado a eficácia do algoritmo FALQON na obtenção do grafo solução para o TSP. Ainda que no segundo experimento, descrito em subseção 4.1.2, a acurácia não tenha atingido o valor máximo (100 %), tanto os modelos clássico quanto quântico apresentaram resultados idênticos, sugerindo que ambos os modelos exibiram comportamento semelhante na seleção de protótipos, mesmo quando aplicados a grafos diferentes. Essa consistência nos resultas fortalece a confiabilidade da capacidade do algoritmo FALQON de encontrar soluções ótimas para o TSP independente das nuances do cenário de teste.

Além disso, a consistência dos resultados obtidos, conforme discutido anteriormente, também comprova a viabilidade e aplicabilidade dos módulos quânticos em problemas de otimização, à medida que a tecnologia quântica se torna cada vez mais acessível. Este trabalho indica que nas atuais limitações tecnológicas de simulação, a abordagem quântica demostrou potencial e, à medida que as tecnologias quânticas evoluem, a aplicação de algoritmos como o FALQON tornam-se ainda mais promissores, tendo em vista a possibilidade de implementá-lo completamente, ao invés do método simplificado.

5 Considerações Finais

Através deste projeto, foi possível adquirir uma compreensão aprofundada dos fundamentos da computação quântica e do classificador OPF. Além disso, demonstrou-se a viabilidade da integração de elementos de implementação quântica em modelos de aprendizado de máquina. Embora o acesso a computadores quânticos permaneça limitado, a exploração da adaptação desses modelos para o contexto quântico tem o potencial de aprimorar significativamente o desempenho e abrir um caminho para futuras aplicações promissoras, impulsionando a pesquisa na interseção entre computação quântica e aprendizado de máquina.

Embora ainda haja espaço para a realização de estudos e experimentos adicionais, este trabalho cumpriu seu propósito ao explorar os fundamentos da computação quântica e da otimização quântica, bem como ao trabalhar com o classificador OPF, que possui diversas aplicações. Os resultados obtidos constituem uma base sólida para pesquisas futuras e evidenciam o potencial das abordagens quânticas na ampliação dos modelos de aprendizado de máquina tradicionais. Isso indica, em outras palavras, que é viável adotar uma abordagem quântica para aprimorar o classificador OPF, abrindo portas para avanços significativos na interseção entre computação quântica e aprendizado de máquina clássico.

5.1 Trabalhos Futuros

Com base nos resultados e conclusões alcançados neste estudo, surgem diversas oportunidades para pesquisas futuras que podem aprimorar a eficiência do algoritmo abordado. Inicialmente, é essencial aprofundar a pesquisa na área de Otimização Quântica, visando aprimorar o algoritmo FALQON para possibilitar a realização de simulações quânticas através da biblioteca Qiskit, além de realizar experimentos usando hardware quântico real.

Adicionalmente, a investigação de técnicas de manipulação de grandes matrizes e a modelagem de grafos quânticos pode ampliar o conjunto de treinamento, possibilitando a análise da vantagem da solução quântica em bases de dados volumosas em comparação com modelos clássicos, enriquecendo ainda mais este projeto. Estas direções futuras têm o potencial de abrir novas perspectivas para produzir um modelo quântico eficiente do classificador.

Referências

DIRAC, P. A. M. **The principles of quantum mechanics**. Fourth edition. Oxford, New York: Oxford University Press, 1982. (International Series of Monographs on Physics). ISBN 978-0-19-852011-5.

FARHI, E.; GOLDSTONE, J.; GUTMANN, S. **A quantum approximate optimization algorithm**. 2014. Disponível em: https://arxiv.org/abs/1411.4028. Acesso em: 02 nov. 2023.

FEYNMAN, R. P. **Simulating physics with computers**. **International Journal of Theoretical Physics**, v. 21, n. 6, p. 467–488, jun. 1982. ISSN 1572-9575. Disponível em: https://doi.org/10.1007/BF02650179. Acesso em: 02 nov. 2023.

GERON, A. **Mãos à obra: aprendizado de máquina com Scikit-Learn e TensorFlow**. 1. ed. [S.l.]: Alta Books, 2019. ISBN 9788550803814.

IBM. **IBM quantum computing**. 2023. Disponível em: https://www.ibm.com/topics/quantum-computing. Acesso em: 02 nov. 2023.

JOHANSSON, J. R.; NATION, P. D.; NORI, F. **QuTiP: An open-source Python framework for the dynamics of open quantum systems**. **Computer Physics Communications**, v. 183, n. 8, p. 1760–1772, ago. 2012. ISSN 0010-4655. Disponível em: https://doi.org/10.1016/j.cpc.2012.02.021. Acesso em: 02 nov. 2023.

JORDAN, S. **Traveling Santa problem**. 2018. Disponível em: http://quantumalgorithmzoo.org/traveling_santa/. Acesso em: 02 nov. 2023.

KUNCHEVA, L. **Artificial data sets**. 2005. Disponível em: https://lucykuncheva.co.uk/activities/artificial_data.htm. Acesso em: 02 nov. 2023.

LOREDO, R. **A década quântica está se movendo muito mais rapidamente que o esperado**. 2021. Disponível em: https://www.ibm.com/blogs/ibm-comunica/decada-quantica/. Acesso em: 02 nov. 2023.

MAGANN, A. B.; RUDINGER, K. M.; GRACE, M. D.; SAROVAR, M. **Feedback-based quantum optimization**. **Physical Review Letters**, v. 129, n. 25, p. 250502, dez. 2022. ISSN 0031-9007, 1079-7114. Disponível em: http://arxiv.org/abs/2103.08619. Acesso em: 02 nov. 2023.

MCCULLOCH, W. S.; PITTS, W. **A logical calculus of the ideas immanent in nervous activity**. 1943. 115–133 p. Disponível em: https://doi.org/10.1007/BF02478259. Acesso em: 02 nov. 2023.

MIANO, M. G. V.; OLIVEIRA, A. S. de. **Desempenho de algoritmos quânticos e clássicos em treinamento de machine learning supervisionado**. **REVISTA TECNOLÓGICA DA FATEC AMERICANA**, v. 09, n. 02, p. 81–99, dez. 2021. ISSN 24467049. Disponível em: https://fatec.edu.br/revista/index.php/RTecFatecAM. Acesso em: 02 nov. 2023.

MONTOYA-ZEGARRA, J. A.; PAPA, J. P.; LEITE, N. J.; TORRES, R. da S.; FALCÃO, A. **Learning how to extract rotation-invariant and scale-invariant features from texture images**. **EURASIP journal on advances in signal processing**, Springer, v. 2008, p. 1–15, 2008. Disponível em: https://doi.org/10.1155/2008/691924. Acesso em: 02 nov. 2023.

MONTOYA-ZEGARRA, J. A.; PAPA, J. P.; LEITE, N. J.; TORRES, R. da S.; FALCAO, A. X. **Novel approaches for exclusive and continuous fingerprint classification**. In: SPRINGER. **Advances in Image and Video Technology: Third Pacific Rim Symposium, PSIVT 2009, Tokyo, Japan, January 13-16, 2009. Proceedings 3**. 2009. p. 386–397. Disponível em: https://doi.org/10.1007/978-3-540-92957-4_34. Acesso em: 02 nov. 2023.

MOORE, G. E. **Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp.114 ff. IEEE Solid-State Circuits Society Newsletter**, v. 11, n. 3, p. 33–35, 2006. Disponível em: https://ieeexplore.ieee.org/document/4785860. Acesso em: 02 nov. 2023.

NIELSEN, M. A.; CHUANG, I. L. **Quantum computation and quantum information**. 10th anniversary. ed. United States od America: Cambridge University Press, 2010. ISBN 978-1-107-00217-3.

PAPA, J. P. **Classificaçãoo supervisionada de padrões utilizando floresta de caminhos otimos**. Tese (Doutorado) — Universidade Estadual de Campinas, Instituto de Computação, 2008. Disponível em: https://hdl.handle.net/20.500.12733/1608859. Acesso em: 02 nov. 2023.

PAPA, J. P. **LibOPF**. 2015. Disponível em: https://github.com/jppbsi/LibOPF. Acesso em: 02 nov. 2023.

PAPA, J. P.; SPADOTTO, A. A.; FALCAO, A. X.; PEREIRA, J. C. **Optimum path forest classifier applied to laryngeal pathology detection**. In: IEEE. **2008 15th International Conference on Systems, Signals and Image Processing**. 2008. p. 249–252. Disponível em: https://ieeexplore.ieee.org/abstract/document/4604414. Acesso em: 02 nov. 2023.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de software: Uma abordagem profissional**. 9. ed. [S.l.]: AMGH, 2021. ISBN 978-6558040101.

QISKIT CONTRIBUTORS. **Qiskit: An open-source framework for quantum computing**. 2023. Disponível em: https://qiskit.org/. Acesso em: 02 nov. 2023.

QISKIT CONTRIBUTORS. **Release notes - Qiskit 0.44.3 documentation**. 2023. Disponível em: https://qiskit.org/documentation/release_notes.html#aqua-0-4. Acesso em: 02 nov. 2023.

RABELO, W. R. M.; COSTA, M. L. M. **Uma abordagem pedagógica no ensino da computação quântica com um processador quântico de 5-qbits**. **Revista Brasileira de Ensino de Física**, v. 40, p. e4306, maio 2018. ISSN 1806-1117, 1806-9126. Publisher: Sociedade Brasileira de Física. Disponível em: https://doi.org/10.1590/1806-9126-RBEF-2018-0038. Acesso em: 02 nov. 2023.

REITZ, K. **Pipenv**. 2017. Disponível em: https://github.com/pypa/pipenv. Acesso em: 02 nov. 2023.

ROCHA, L. M.; FALCAO, A. X.; MELONI, L. G. **A robust extension of the mean shift algorithm using optimum path forest**. In: **Proc. of the 12th Intl. Workshop on Combinatorial Image Analysis**. [s.n.], 2008. p. 29–38. Disponível em: https://www.researchgate.net/publication/256296614_A_Robust_Extension_of_the_Mean_Shift_Algorithm_using_Optimum_Path_Forest. Acesso em: 02 nov. 2023.

ROSA, G. H. de; PAPA, J. P. **OPFython: A Python implementation for Optimum-Path Forest**. **Software Impacts**, p. 100113, 2021. ISSN 2665-9638. Disponível em: https://doi.org/10.1016/j.simpa.2021.100113. Acesso em: 02 nov. 2023.

SAMUEL, A. L. **Some studies in machine learning using the game of checkers**. **IBM Journal of Research and Development**, v. 3, n. 3, p. 210–229, jul. 1959. ISSN 0018-8646. Conference Name: IBM Journal of Research and Development. Disponível em: https://ieeexplore.ieee.org/document/5392560. Acesso em: 02 nov. 2023.

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

WAKEHAM, D.; CERONI, J. **Feedback-based quantum optimization (FALQON)**. **PennyLane Demos**, maio 2021. Disponível em: https://pennylane.ai/qml/demos/tutorial_falqon/. Acesso em: 02 nov. 2023.