Universidade Federal de Pernambuco

Centro de Informática

Curso de Ciência da Computação

Funções de Ativação Arbitrárias para Redes Neurais Quânticas

Análise e Desenvolvimento

Trabalho de Conclusão de Curso de Graduação

por

Matheus Hopper Jansen Costa

Orientador: Prof. Dr. Fernando Maciano

Recife, Maio / 2023

Matheus Hopper Jansen Costa

Funções de Ativação Arbitrárias para Redes Neurais Quânticas

Monografia apresentada ao Curso de Ciência da Computação, como requisito parcial para a obtenção do Título de Bacharel em Ciência da Computação, Centro de Informática da Universidade Federal de Pernambuco.

Orientador: Prof. Dr. Fernando Maciano

Recife

2023

Agradecimentos

Eu gostaria de agradeçer a todos vocês que me ajudaram durante esta jornada, especialmente para:

Professor Dr. Fernando Maciano por sua paciência e compreensão durante todo o desenvolvimento.

À minha mãe Rachel, meu pai Ricardo, minha namorada Ana Clara e meus amigos Igor e Ícaro por acreditarem em mim.

Ao meu avô que faleceu em 04/09/2022 e tinha o sonho de ver eu me formar.

Qualquer tecnologia suficientemente avançada é indistinguível de magia.

Arthur C. Clarke

RESUMO

Atualmente, é esperado que o campo das redes neurais artificiais se beneficie fortemente dos avanços na computação quântica. Em particular, a aprendizagem de máquina quântica, uma classe de algoritmos que utiliza conceitos da mecânica quântica para criar algoritmos inteligentes, proporcionará um aumento na velocidade de processamento para realizar tarefas como reconhecimento de padrões, agrupamento e aprendizado de máquina em geral. Já existem perceptrons quânticos com funções de ativação específicas e é possível reproduzir algumas funções de ativação experimentalmente, embora haja impacto de parâmetros de implementação não identificados durante a reprodução. Neste trabalho, objetiva-se adaptar os experimentos de um algoritmo quântico proposto em "Quantum Activation Function for Quantum Neural Network", utilizando a biblioteca Qiskit para gerar funções de ativação arbitrarias.

Palavras-chave: Computação Quântica, Inteligência Artificial, Funções de Ativação, Qiskit, Aprendizado de Máquina Quântico.

ABSTRACT

Currently, the field of artificial neural networks is expected to benefit greatly thanks to advances in quantum computing. In particular, quantum machine learning, a class of algorithms that use quantum mechanics concepts to create intelligent algorithms, will provide an increase in processing speed to perform tasks such as pattern recognition, clustering, and machine learning in general. Quantum perceptrons with specific activation functions already exist, and although some alternatives are already available to perform arbitrary activation functions in the quantum processor, they still have their implementation limitations. In this work, the objective is to replicate the experiments of a proposed quantum algorithm in "Quantum Activation Function for Quantum Neural Network"using the Qiskit to generate arbitrary activation functions.

Keywords: Quantum Computing, Artificial Intelligence, Activation Functions, Qiskit, Quantum Machine Learning.

LISTA DE FIGURAS

LISTA DE SIGLAS

$UFPE$	Universidade Federal de Pernambuco
AI	Inteligência Artificial
QML	Aprendizagem de Máquina Quântico
QNN	Rede Neural Quântica
RUS	Circuitos de Repetição até o Sucesso

SUMÁRIO

1 INTRODUÇÃO

Uma rede neural quântica (QNN - Quantum Neural Network) é codificada através de qubits no processador quântico. Na computação clássica um neurônio é implementado através de funções matemáticas. Por exemplo, o perceptron de Rosenblatt (1952) [1] é a rede neural mais simples que é constituída por uma camada de entrada com N neurônios e uma camada de saída com apenas um neurônio que se comporta como a função de ativação da rede. Perceptrons de múltiplas camadas [2] são arquiteturas um pouco mais complexas do que o perceptron, isso porque além das camadas de entrada e de saída, esses perceptrons também podem possuir uma ou mais camadas intermediárias de neurônios que agem como aproximadores universais de funções. Nessa camada intermediária, as funções comprimem os valores reais em valores normalizados, agindo como uma função de ativação [3].

Em princípio, computadores quânticos são adequados para realizar cálculos de produto tensorial [4,5]. De fato, os qubits podem ser arranjados em circuitos agindo como as camadas do análogo quântico de redes neurais. Se combinados com funções de ativações clássicas como a sigmóide e tangente hiperbólica, elas devem ser capazes de processar algoritmos de aprendizagem profunda como os usados para problemas de classificação, reconhecimento de padrões, clusterização e tomada de decisão. Como depois do processo de medição o qubit perde suas propriedades e se torna um bit clássico, implementar funções de ativação numa QNN é um desafio que requer uma abordagem mais imediata. Nosso objetivo é preservar o máximo de informação possível do qubit enquanto nos aproveitamos de cada computação da rede ao mesmo tempo. Então precisamos atrasar a medição até o fim do fluxo computacional, depois de ter processado a informação pelos neurônios com uma função de ativação adequada. No aprendizado de máquina quântica (QML- Quantum Machine Learning) [6,7], ignorando a implementação de redes neurais quânticas em computadores quânticos adiabáticos [8], existem duas maneiras de se codificar uma QNN num modelo quântico baseado em portas quânticas. A primeira consiste em definir uma QNN como um circuito quântico variacional, onde a não-linearidade é introduzida através de medições [9,10]. Essas QNNs não são baseadas em teoremas matemáticos mas em modelos heurísticos que são empíricos [11]. Além disso, esse tipo de modelos baseados em algoritmos quânticos variacionais sofrem de um problema, o desaparecimento de gradi-

entes exponenciais, conhecido como problema de barren plateau [12], que precisam de técnicas para mitigá-los [13,14]. Bem diferente, a segunda abordagem busca implementar um algoritmo verdadeiramente quântico para computação de rede neural e para realmente cumprir os requisitos de aproximação do teorema de Hornik [15], talvez ao custo de um circuito de maior profundidade. Tal abordagem diz respeito a modelos semi clássicos [16,17] ou totalmente quânticos [18,19] onde a função de ativação também é computada através da medição.

Além disso, existem várias formas de codificar os valores de entrada nos qubits. Já que um qubit representa uma superposição dos estados 0 e 1, alguns métodos de codificação são diferenciados pelo relacionamento entre o número de qubit e a capacidade máxima de codificação. A primeira é a opção 1-para-1 pela qual cada neurônio de entrada da rede corresponde a um qubit [20–22]. A implementação mais simples consiste em armazenar as informações numa sequência de bits atribuídos aos estados da base clássica do espaço quântico. Um método semelhante de 1-para-1 consiste em armazenar uma superposição de dados binários como uma série de bit em um estado de múltiplos qubits. Tais redes neurais quânticas são baseadas no conceito de memória associativa quântica [23, 24]. Uma outra maneira de usar 1-para-1 é dada pelo quron [25]. Um quron é um qubit onde os estados 0 e 1 representam a rede em repouso e ativada, respectivamente [25].

Alternativamente, outra maneira de codificar os qubit é fazendo a amplitude dos estados quânticos em superposição guardarem a informação do vetor de entrada [26–30]. A eficiência de codificação torna-se exponencial como um estado com n-qubits é um elemento de um espaço vetorial de dimensão n^2.

No entanto, codificar as entradas como coeficientes de uma superposição de estados quânticos requer um algoritmo genérico para preparações de estado quântico [31–33] ou, alternativamente, para alimentar diretamente dados quânticos para a rede [34]. Geralmente, para preparar um estado quântico arbitrário de n-qubits requer um número de portas quânticas que escalam exponencialmente com n. Mas a longo prazo, uma codificação de tipo n-para-2^n garante uma melhor aplicabilidade a problemas reais do que as opções 1-para-1. Além disso, tal método de codificação satisfaz os requisitos do teorema de Hornik para garantir a capacidade de aproximação da função universal [15]. Apesar de algumas restrições relativamente pesadas, como a codificação digital e o fato de que a função de ativação envolve medidas irreversíveis, exemplos para essa declaração foram

relatados em [27, 29, 30].

Definimos um método de codificação n-para-2^n que envolve entradas, pesos e um bias no intervalo de [-1, 1] em R. O modelo usa uma arquitetura baseada em portas quânticas para criar funções de ativação de aproximação arbitrárias usando apenas operações reversíveis. O algoritmo consiste em iterar todas as computações de produto interno até d-ésima ordem, onde d é um conjunto de qubits adicionado ao circuito que não faz parte do conjunto de entrada n. Consequentemente, a aproximação das funções de ativação mais comuns podem ser calculadas reconstruindo a série de Taylor na d-ésima ordem.

1.1 **Objetivos**

Este trabalho tem como objetivo adaptar o algoritmo quântico que gera funções de ativações arbitrárias e usar esse algoritmo para construir o perceptron quântico proposto em "Quantum Activation Function for Quantum Neural Network", modelo construído com métricas de desempenho de classificação.

1.2 **Organização do texto**

O algoritmo é implementado utilizando python e a biblioteca Qiskit [36] para construir um perceptron de uma única camada com 4 neurônios e diferentes funções de ativação como sigmóide e seno, truncadas pela sétima ordem. Os próximos capítulos deste trabalho estão organizados da seguinte forma: na seção 2, algumas definições e a abordagem geral são definidas; na seção 3 detalhamos o que são neurônios clássicos e quânticos, na seção 4 explicitamos os detalhes da implementação dos autores do artigo e deste trabalho, passando por cada algoritmo que foi construído; Na seção 5, descrevemos os resultados e comparamos os resultados obtidos por mim com os resultados dos autores; e por fim, concluo na seção 6 com o que podemos esperar no futuro.

2 CONCEITOS DA ÁREA

Nesta sessão abordo alguns conceitos importantes para o entendimento do trabalho

2.1 Inteligência Artificial

Historicamente, pesquisadores têm buscado várias versões diferentes de Inteligência Artificial (AI - Artificial Intelligence). Alguns definiram a inteligência em termos de fidelidade ao desempenho humano, enquanto outros preferem uma definição abstrata e formal de inteligência chamada racionalidade, falando livremente, fazendo a "coisa certa". O assunto em si também varia: alguns consideram a inteligência uma propriedade de processos internos de pensamento e raciocínio, enquanto outros se concentram no comportamento inteligente, uma caracterização externa. [35]

Aos olhos do público, às vezes há confusão entre os termos "inteligência artificial" e "aprendizado de máquina". O aprendizado de máquina é um subcampo de AI que estuda a capacidade de melhorar o desempenho com base na experiência. Alguns sistemas de AI usam métodos de aprendizado de máquina para alcançar competência, mas outros não. [35]

Na inteligência artificial, um agente inteligente é qualquer coisa que perceba seu ambiente, age de forma autônoma para atingir objetivos, e pode melhorar seu desempenho com o aprendizado ou pode utilizar o conhecimento . Eles podem ser simples ou complexos - um termostato é considerado um exemplo de agente inteligente, assim como um ser humano, como qualquer sistema que se enquadre na definição, como uma empresa, um estado ou um bioma.

2.2 Aprendizado de Máquina

Um agente está aprendendo se melhorar seu desempenho depois de fazer observações sobre o mundo. O aprendizado pode variar do trivial, como anotar uma lista de compras, ou profundo, como quando Albert Einstein inferiu uma nova teoria do universo. Quando o agente é um computador, chamamos isso de aprendizado de máquina: um computador observa alguns dados, constrói um modelo com base nos dados e usa o modelo como uma hipótese sobre o mundo e um software que pode resolver problemas.

Por que queremos que uma máquina aprenda? Por que não apenas programá-lo da maneira certa para começar? Há duas razões principais. Primeiro, os projetistas não podem prever todas as situações futuras possíveis. Por exemplo, um robô projetado para navegar em labirintos deve aprender o layout de cada novo labirinto que encontra; um programa para prever os preços do mercado de ações deve aprender a se adaptar quando as condições mudam de alta para queda. Em segundo lugar, às vezes os designers não têm ideia de como programar uma solução por conta própria.

Quando a saída do aprendizado é um conjunto finito de valores (como ensolarado/nublado/chuvoso ou verdadeiro/falso), o problema de aprendizado é chamado de classificação. Quando é um número (como a temperatura de amanhã, medida como um número inteiro ou um número real), o problema de aprendizado é uma regressão. Existem três tipos de feedback que podem acompanhar as entradas e que determinam os três principais tipos de aprendizagem:

No **aprendizado supervisionado**, o agente observa os pares de entrada-saída e aprende uma função que mapeia da entrada para a saída. Por exemplo, as entradas podem ser imagens de câmeras, cada uma acompanhada por uma saída dizendo "ônibus" ou "pedestre", etc. Uma saída como essa é chamada de rótulo. O agente aprende uma função que, ao receber uma nova imagem, prevê o rótulo apropriado.

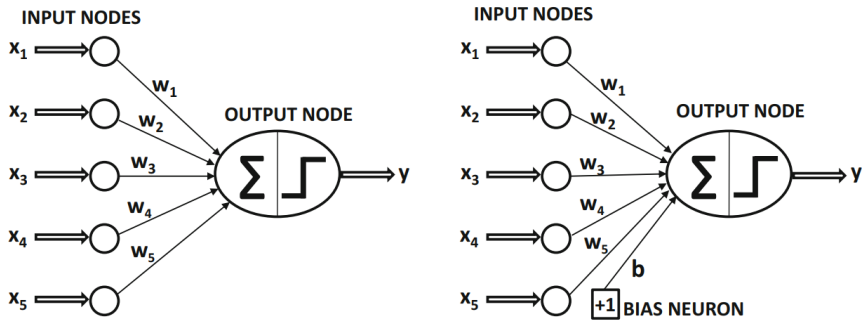
No **aprendizado não-supervisionado**, o agente aprende padrões na entrada sem nenhum feedback explícito. A tarefa de aprendizado não supervisionada mais comum é o agrupamento: detectar agrupamentos potencialmente úteis de exemplos de entrada.

Na **aprendizado por reforço** o agente aprende a partir de uma série de reforços: recompensas e punições. Cabe ao agente decidir quais das ações anteriores ao reforço foram as maiores responsáveis por ele, e alterar suas ações visando mais recompensas no futuro. [35]

2.3 Redes Neurais

Redes Neurais Artificiais são técnicas populares de aprendizado de máquina que simulam o mecanismo de aprendizagem biológica. Numa rede de camada única, um conjunto de entradas é mapeado diretamente para uma saída usando uma variação generalizada de uma função linear. Essa simples instanciação de uma rede neural também é chamada de perceptron. Em redes neurais multicamadas, os neurônios são organizados

em camadas, em que as camadas de entrada e saída são separadas por um grupo de camadas ocultas. Essa arquitetura de camada da rede neural também é chamada de rede feed-forward.



(a) Perceptron without bias (b) Perceptron with bias

Figura 1: Estrutura de um neurônio clássico, Perceptron

A rede neural mais simples é chamada de perceptron. A arquitetura básica do perceptron é mostrada na Figura 1(a). Esta rede neural contém uma única camada de entrada e um nó de saída. Considere uma situação em que cada instância de treinamento é da forma (X,y), onde cada X $= [x_1, ..., x_d]$ contém d variáveis e y $\in \{-1, +1\}$ contém o valor observado do variável de classe binária. Por "valor observado" nos referimos ao fato de que ele nos é dado como parte dos dados de treinamento, e nosso objetivo é prever a variável de classe para casos em que ela não é observada.

A camada de entrada contém d nós que transmitem as d características $\vec{X} = [x_1, ..., x_d]$ com arestas de peso $\vec{W} = [w_1, ..., w_d]$ para um nó de saída. A camada de entrada não executa nenhum cálculo por si só. A função linear $\vec{W} \cdot \vec{X} = \sum_{i=1}^{d} w_i * x_i$ é calculada no nó de saída. Posteriormente, o sinal desse valor real é usado para prever a variável dependente de $\vec{X}$. Portanto, a previsão $\hat{y}$ é calculada da seguinte forma:

$$\hat{y} = sign\{\vec{W}\vec{X}\} = sign\{\sum_{i=1}^{d} w_i * x_i\} \tag{2.1}$$

A função sign mapeia um valor real para +1 ou −1, que é apropriado para classificação binária. Observe o circunflexo no topo da variável y para indicar que é um valor previsto em vez de um valor observado. A função sign cumpre o papel de uma função de ativação. Diferentes opções de funções de ativação podem ser usadas para simular diferentes tipos de modelos usados em aprendizado de máquina, como regressão de mínimos

quadrados com alvos numéricos, a máquina de vetores de suporte ou um classificador de regressão logística. A maioria dos modelos básicos de aprendizado de máquina pode ser facilmente representada como arquiteturas simples de rede neural. É um exercício útil modelar técnicas tradicionais de aprendizado de máquina como arquiteturas neurais, pois fornece uma imagem mais clara de como o aprendizado profundo generaliza o aprendizado de máquina tradicional.

Em muitas configurações, há uma parte invariável da previsão, que é chamada de viés. Por exemplo, considere uma configuração em que as variáveis são centradas na média, mas a média da previsão da classe binária de $\{-1, +1\}$ não é 0. Isso tenderá a ocorrer em situações em que a distribuição da classe binária é altamente desequilibrada. Nesse caso, a abordagem acima mencionada não é suficiente para a previsão. Precisamos incorporar uma variável de viés adicional b que captura essa parte invariável da previsão:

$$\hat{y} = sign\{\vec{W}\vec{X} + b\} = sign\{\sum_{i=1}^{d} w_i * x_i + b\} \tag{2.2}$$

O viés pode ser incorporada como o peso de uma aresta usando um neurônio com viés. Isso é conseguido adicionando um neurônio que sempre transmite um valor de 1 para o nó de saída. O peso da aresta que conecta o neurônio ao nó de saída fornece o viés. Um exemplo de um neurônio com viés é mostrado na Figura 1(b). [36]

2.4 **Computação Quântica**

A computação quântica é uma fusão da física quântica com a ciência da computação. Ele incorpora algumas das ideias da física do século XX para uma maneira inteiramente nova de pensar a computação. A unidade básica da computação quântica é o qubit. Um bit clássico é 0 ou 1. Se for 0 e medirmos, obteremos 0. Se for 1 e medirmos 1, obteremos 1. Em ambos os casos, o bit permanece inalterado. A situação é totalmente diferente para qubits. Um qubit pode estar em um número infinito de estados - uma superposição de 0 e 1, mas quando o medimos, como no caso clássico, obtemos apenas um de dois valores, 0 ou 1. O ato de medir muda o qubit. Um modelo matemático simples descreve tudo isso com precisão. Qubits também podem ser emaranhados. Quando fazemos uma medição de um deles, isso afeta o estado do outro.

Essas três coisas – superposição, medição e emaranhamento – são as ideias-chave

da mecânica quântica. Uma vez que sabemos o que eles significam, podemos ver como eles podem ser usados na computação.

Definimos um qubit como qualquer unidade ket em $\Re^2$. Normalmente, dado um qubit, queremos medi-lo. Se vamos medi-lo, também precisamos incluir uma direção de medição. Isso é feito introduzindo uma base ortonormal ordenada $(|b_0\rangle, |b_1\rangle)$ O qubit pode ser escrito como uma combinação linear - muitas vezes chamada de superposição linear - dos vetores de base. Em geral, terá a forma $d_0|b_0\rangle + d_1|b_1\rangle$. Depois de medirmos, seu estado saltará para $|b_0\rangle$ ou $|b_1\rangle$. A probabilidade de ser $|b_0\rangle$ é d_0^2; a probabilidade de $|b_1\rangle$ é d_1^2. Este é exatamente o mesmo modelo que temos usado, mas agora conectamos os bits clássicos 0 e 1 aos vetores de base. Vamos associar o vetor de base $|b_0\rangle$ com o bit 0 e o vetor de base $|b_1\rangle$ com o bit 1. Assim, quando medirmos o qubit $d_0|b_0\rangle + d_1|b_1\rangle$ obteremos 0 com probabilidade d_0^2 e 1 com probabilidade d_1^2.

Na mecânica quântica, para realizar alguma medição, temos que interagir com o sistema. Essas interações vão perturbar nosso sistema. Como veremos, nosso modelo não leva em consideração a "força" da medida. Em vez disso, é o processo real de fazer a medição, como quer que seja feito, que afeta o sistema. Cada vez que uma medição é feita, veremos que o sistema é alterado de certas maneiras prescritas; essas formas prescritas dependem do tipo de medição que está sendo feita, mas não da força da medição. Incorporar medições na teoria é uma das diferenças entre a mecânica clássica e a quântica. [37]

O emaranhamento quântico descreve quando pares ou grupos de partículas são gerados ou interagem de maneira que o estado quântico de cada partícula não pode ser descrito independentemente e o estado quântico é um sistema completo. No sistema emaranhado, as medidas de propriedades físicas como posição, momento, spin e polarização são adequadamente correlacionadas. Se um par de partículas é gerado de forma tão emaranhada que seu spin total é zero. Uma partícula de um par emaranhado sabe qual medição foi realizada na outra, que no momento da medição pode estar separada por distâncias arbitrariamente grandes. [38]

3 NEURÔNIOS

Explicar um pouco da relação de neurônios artificiais com os neurônios humanos

3.1 Neurônio Clássico

Um neurônio clássico é uma função que leva n variáveis $x_1, x_2, ..., x_n$ e mapeia-os para o valor de saída $a = \sigma(w_1x_1 + w_2x_2 + ... + w_nx_n + b)$ com w_i e b sendo os pesos e viés sinápticos, respectivamente (Figura 1). A função $\phi = w_1x_1 + ... + w_nx_n + b$ é chamado de sinal de entrada para o neurônio. A função de ativação σ é uma função não linear.

Um neurônio pode estar em um estado ativo ou em um estado de repouso, representado por 1 ou 0 em notação matemática. A Figura 1 mostra a estrutura de um neurônio: entrada $(x_1, x_2, ..., x_n)$ e seus pesos correspondentes $(w_1, w_2, ..., w_n)$ e viés b e a função de ativação f que combina todos os componentes da entrada para gerar uma saída. Note que sem f, a saída é uma transformação linear direta da entrada. Um neurônio é uma simples unidade de deslocamento com muitas entradas e uma saída.

3.2 Neurônio Quântico

A fim de imitar, usando um circuito quântico, a função do neurônio clássico onde as entradas $x_1, \ldots, x_n \in \{0, 1\}$ são combinadas linearmente para formar uma entrada $\theta = w_1x_1 + \cdots + w_nx_n + b$, pode-se simplesmente usar o estado $|x\rangle = |x_1 \ldots x_n\rangle$ como um estado de controle e aplicar $R_y(2w_i)$ em um qubit ancilla condicionado no iésimo-qubit, seguido por $R_y(2b)$ no qubit ancilla. Isso equivale a aplicar o $R_y(2\theta)$ no ancilla qubit condicionado ao estado $|x\rangle$ dos neurônios de entrada (Figura 2 **b** e **c**). O segundo passo é realizar uma rotação por $R_y(2\sigma(\theta))$ onde σ é uma função não linear (sigmóide ou função de limiar). Aproximamos essa rotação por uma classe de circuitos chamados circuitos de repetição até o sucesso (RUS - Repeat Until Succes) [39]. A Figura 2-**c** mostra um circuito que implementa $R_y(2q(\theta))$ onde $q(\theta) = \arctan(\tan^2\theta)$ é uma função não linear do tipo sigmóide (Figura 2-**d**). A ação de um circuito RUS no qubit de saída depende do resultado da medição do qubit ancilla. Se a medição retornar $|0\rangle$, isso indica que a rotação por $2q^{ok}(\theta)$ foi aplicado com sucesso ao qubit de saída. Caso contrário, se o qubit ancilla medir $|1\rangle$, isso indica que o circuito implementou uma rotação $R_y(\pi/2)$ no qubit ancilla.

Neste caso corrigimos a operação aplicando $R_y(-\pi/2)$ e depois repetimos o circuito até que $|0\rangle$ seja medido no ancilla qubit, daí o nome repetição até sucesso.

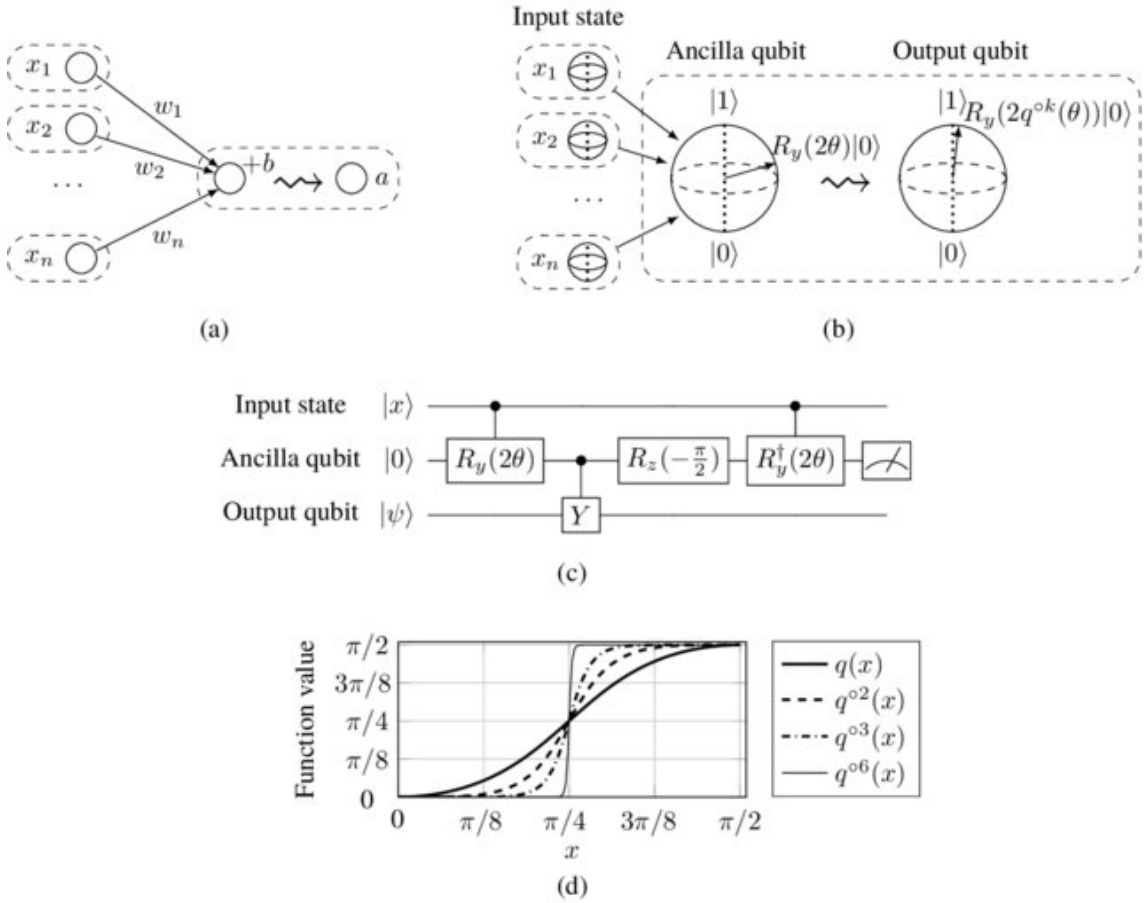


Figura 2: Configuração básica do modelo de neurônio quântico. **a)** O neurônio clássico (marcado com caixas tracejadas). As entradas $x_1, ..., x_n$ são combinados com pesos específicos w_i , e tendenciosos por b para formar $\phi = w_1x_1 + ... + w_nx_n + b$. A ativação de saída é a $= \sigma(\phi)$, com σ sendo uma função sigmoide ou passo. **b)** O neurônio quântico (marcado por caixas tracejadas). Visualização da esfera bloch do estado de qubit de saída antes e depois da RUS, correspondendo à função de ativação linear e não linear, respectivamente. A função q é mostrada na subtrama . **d**. A notação representa a atualização interna do neurônio correspondente à função de ativação. Na prática, é a capacidade de usar rotações por 2ϕ para implementar rotações por 2q·k (ϕ) através de circuitos repetidos até o sucesso. Supõe-se que o estado de entrada seja preparado por algum método externo, possivelmente controlado por outros neurônios quânticos. **c)** Circuito repetido até o sucesso (RUS) para realizar rotação com um ângulo q(ϕ) = arctan(tan 2 ϕ). Aqui nós usamos a convenção R_p (ϕ) = exp (-iP ϕ/2) onde p $\in \{x, y, z\}$ rotula operadores Pauli P $\in \{X, Y, Z\}$. **d)** Função não linear q(ϕ) = arctan (tan 2 ϕ) e sua autocomposição q·k (ϕ) = arctan (tan 2 k ϕ).

4 ALGORITMO

4.1 Abordagem Geral

A fim de definir uma versão quântica do perceptron com parâmetros contínuos e função de ativação analítica arbitrária, vamos considerar um perceptron de uma camada. Um perceptron de uma camada é composto de neurônios de entrada N_{in} e um neurônio de saída equipado de uma função de ativação $f : R \to I$ onde I é um conjunto compacto. O neurônio de saída calcula o produto interno entre o vetor dos valores de entrada $\vec{x} = (x_0, x_1, ..., x_{N_{in}-1}) \in R^{N_{in}}$ e o vetor de pesos $\vec{w} = (w_0, w_1, ..., w_{N_{in}-1})$ mais um viés b. Tal valor escalar é tomado como argumento de uma função de ativação. O valor real de saída $\hat{y} \in$ I do perceptron é definido como $\hat{y} \equiv f(w \cdot x + b)$.

Aqui, desenvolvemos um circuito quântico que computa uma aproximação de $\hat{y}$. O algoritmo começa calculando o produto interno $\vec{W} \cdot \vec{X}$ mais o valor de viés b. Depois ele avalia a saída $\hat{y}$ calculando uma aproximação da função de ativação f. Em um computador quântico, uma operação de medição aparentemente representa a implementação mais simples de uma função de ativação não linear, como feito por exemplo em [27] para resolver um problema de classificação binária em um perceptron quântico. Essa abordagem, no entanto, não pode ser generalizada para construir uma rede neural baseada em qubits de várias camadas.

Em primeiro lugar, as operações de medição quebram o algoritmo quântico e impõem a inicialização dos qubits camada por camada, impedindo assim uma única execução quântica de um rede neural de várias camadas. Em segundo lugar, outras funções de ativação — além das implícitas pelas operações de medição, são mais adequadas para resolver problemas genéricos de aprendizado de máquina.

Este trabalho evita esses dois problemas com um novo algoritmo quântico, que é baseado em dois teoremas como detalhado abaixo. O algoritmo quântico é composto de dois passos. Primeiro, as potências de $\vec{W}\vec{X} + b$ são armazenados como amplitudes de um estado quântico multi-qubit. Em seguida, a função de ativação escolhida é aproximada construindo sua expansão da série polinomial de Taylor através de rotações do estado quântico. Os ângulos de rotação são determinados pelos coeficientes da série polinomial da função de ativação escolhida. Eles podem ser explicitamente computados pelo algoritmo quântico proposto.

4.2 Notações

Primeiro resumimos a notação usada ao longo do texto. Sendo H o espaço bi-dimensional de Hilbert associado a um qubit. Em seguida, o espaço de Hilbert 2^n-dimensional associado a um registrador q de n qubits é escrito como $H_q^{\otimes n} \equiv H_{qn-1} \otimes H_{qn-2} \otimes ... \otimes H_{q0}$. Se denotarmos por $\{|0\rangle, |1\rangle\}$ a base computacional em H, então a base computacional em $H_q^{\otimes n}$ se lê $\{|s_{n-1}, s_{n-2}, ..., s_0\rangle, s_k \in \{0,1\}, k = \{0,1,...,n-1\}$. Um elemento $|s_{n-1}, s_{n-2}, ..., s_0\rangle$ desta base computacional pode ser escrito alternativamente como $|i\rangle$ onde $i \in \{0,1,...,2_{n-1}\}$ é o número decimal inteiro que corresponde à sequência de bit $s_{n-1}, s_{n-2}, ..., s_0$. Em particular, se $N = 2^n$, então $|N-1\rangle \equiv |2^n - 1\rangle \equiv |11...1\rangle \equiv |1\rangle^{\otimes n}$. Nesta notação, o número de qubits de um registrador é indicado com uma letra minúscula, como n e d, enquanto a dimensão do espaço Hilbert associado é indicada pela letra maiúscula correspondente, como $N = 2^n$ e $D = 2^d$.

A expressão ${U_q}^n = U_{q_{n-1}} \otimes U_{q_{n-2}} \otimes ... \otimes U_{q_0}$ representa uma transformação unitária separável construída com transformações de um qubit U_{q_j} atuando em cada qubit do registrador q. Uma transformação unitária de múltiplos qubits não separável é geralmente escrita como U_q e, em alguns casos, simplesmente U. Dois registradores a e q, respectivamente com d e n qubits, podem ser compostos em um único registrador suportando o $N + D$ espaço de Hilbert ${H_a}^{\otimes d} \otimes {H_q}^{\otimes n}$ com base computacional $\{|i\rangle_a |j\rangle_q$, com $i = 1, ..., D$ e $j = 1, ..., N$ $\}$. Por brevidade, usaremos a notação compacta 0_q para $1_a \otimes 0_q$ e 0_a para $1_a \otimes 0_q$ para operadores O atuando em apenas um dos dois registradores. Em particular, escrevemos

$$|i\rangle\langle i| \equiv 1_a \otimes |i\rangle_{qq}\langle i| \tag{4.1}$$

para a projeção D-dimensional no estado $|i\rangle$ do registrador q.

Casos particulares de operadores unitários implementáveis em um computador quântico de modelo de circuito são as portas controladas. Deixe $C_{q_i}U_{q_j}$ representar uma transformação U controlada: o operador U é aplicado no qubit q_j (chamado qubit alvo) se q_i estiver no estado $|1\rangle$ (chamado qubit de controle). A transformação $\bar{C}_{q_i}U_{q_j}$é uma transformação controlada onde a porta U é aplicada no qubit q_j se o qubit q_i estiver no estado $|0\rangle$. Portanto, $\bar{C}_{q_i}U_{q_j} = X_{q_i}C_{q_i}U_{q_j}X_{q_j}$onde, nesse caso, a é o conjunto de qubits de controle enquanto q_j é o qubit alvo.

A seguir, dois registradores de qubit q e a de n e d qubits, respectivamente, são atribuídos.

4.3 Computação da Série Polinomial

Como dito acima, o objetivo é construir um estado quântico de $(n+d)$ qubits contendo a expansão de Taylor de f(z) na ordem d, onde $z \equiv \vec{w} \cdot \vec{x} + b$ até um fator de normalização. O número n de qubits necessários, além de d, é determinado pela dimensão do vetor de entrada. Primeiro precisamos codificar as potências $(1, z, z^2, ..., z^d)$ nos qubits (n + d). O seguinte Lema fornece o ponto de partida:

Lema 1: Dados dois vetores $\vec{x}$, $\vec{w} \in [-1, 1]^{N_{in}}$ e um número b ∈ [-1, 1], e dado um registrador com n qubits tal que $N = 2^n \geq N_{in} + 3$, então existe um circuito quântico realizando uma transformação unitária tal que:

$$\langle N-1|U_z(\vec{x}, \vec{w}, b)|0\rangle = \frac{\vec{w}\vec{x} + b}{N_{in} + 1} \equiv z \tag{4.2}$$

onde $|0\rangle \equiv |0\rangle^{\otimes n}$ e $|N-1\rangle \equiv |1\rangle^{\otimes n}$.

No Lema 1 um operador unitário U_z de n qubits é definido pelo requesito que a equação 4.2 detém, onde $b \equiv [-1, 1]$, $\vec{x} = (x_0, x_1, \ldots, x_{N_{in-1}})$ e $\vec{w} = (w_0, w_1, \ldots, w_{N_{in-1}})$, onde $N_{in} \leq 2^n - 3$ e x_i, w_i ∈ [-1, 1]. A existência de um número infinito desses operadores é trivialmente óbvia do ponto de vista puramente matemático. O problema é fornecer uma realização explícita em termos de portas quânticas realistas.

Prova: Vamos definir dois vetores em $\Re^N$: $\vec{v}_x = (\vec{x}, 1, A_x, 0)$ e $\vec{v}_{w,b} = (\vec{w}, b, 0, A_{w,b})$ onde $N \equiv 2^n$. Em tais vetores $N - N_{in} - 3$ coeficientes são sempre nulos enquanto os valores A_x e $A_{w,b}$ são constantes definidas de tal forma que $\vec{v}_x\vec{v}_x = \vec{v}_{w,b}\vec{v}_{w,b} = N_{in} - 1$. Segue-se então que $\vec{v}_{w,b}^T\vec{v}_x = \vec{w}\vec{x} + b \in [-N_{in} - 1, N_{in} + 1]$. Agora definimos dois estados quânticos de n-qubits $|\psi_x\rangle$ e $|\psi_{w,b}\rangle$ de seguinte modo

$$|\Psi_x\rangle = \sum_{i=0}^{N-1} \frac{\vec{v}_{x,i}}{\sqrt{N_{in}+1}}|i\rangle, |\Psi_{w,b}\rangle = \sum_{i=0}^{N-1} \frac{\vec{v}_{w,b,i}}{\sqrt{N_{in}+1}}|i\rangle \tag{4.3}$$

Então, por construção

$$\langle \Psi_x | \Psi_{w,b} \rangle = \frac{\vec{w}\vec{x} + b}{N_{in} + 1} \equiv z \tag{4.4}$$

O algoritmo de inicialização mencionado acima nos permite considerar transformações unitárias $U_x = U(\vec{v}_x)$ e $U_{w,b} = X^{\otimes n} U^\top(\vec{v}_{w,b})$, onde X é uma porta quântica equivalente a porta NOT na computação clássica, tal que $U_x|0\rangle = |\Psi_x\rangle$ e $U_{w,b}|\Psi_{w,b}\rangle = |1\rangle$. Segue que

$$\langle \Psi_x | \Psi_{w,b} \rangle = \langle \Psi_{w,b} | U_{w,b}^\top U_{w,b} | \Psi_x \rangle = \langle N-1 | U_{w,b} | \Psi_x \rangle = \langle N-1 | U_{w,b} U_x | 0 \rangle^{\otimes n} \tag{4.5}$$

Comparando com as equações 4.3 vemos que $U_z(\vec{x}, \vec{w}, b) = X^{\otimes n} U^\top(\vec{v}_{w,b}) U(\vec{v}_x)$.

Como as amplitudes dos estados $|\Psi_x\rangle$ e $|\Psi_{w,b}\rangle$ são reais, as fases são 0 ou π e não é mais necessário aplicar uma série de portas R_z multicontroladas para defini-los. Uma única transformação diagonal é suficiente, com 1 ou -1 na diagonal. Para isso, os estados hipergráficos se mostram eficazes [40]. Graças a esse tipo de estado, um pequeno número de portas Z, CZ e Z multicontroladas são necessárias para alcançar a transformação $U(\vec{v})$. As transformações, que introduzem as fases das amplitudes de um estado quântico de n-qubits, são resumidas por um operador chamado $Ph_n^\top$ na figura Figura 3.

Existem muitas alternativas para os estados $|\Psi_x\rangle$ e $|\Psi_{w,b}\rangle$ que dão o mesmo produto interno $\langle \Psi_{w,b} | \Psi_x \rangle$ = z. Definindo dois vetores

$$\vec{v}_x = (A_x, x_0, \ldots, x_{N_{in}-1}, 1, 0, \ldots, 0, 0) \in \Re^N$$
$$\vec{v}_{w,b} = (0, w_0, \ldots, w_{N_{in}-1}, b, 0, \ldots, 0, A_{w,b}) \in \Re^N$$

então a transformação $U(\vec{v}_x)$ e $U(\vec{v}_{w,b})$ aplicada no estado $|0\rangle^{\otimes n}$ retorna dois estados, $|\Psi_x\rangle$ e $|\Psi_{w,b}\rangle$ respectivamente, tal que $\langle \Psi_{w,b} | \Psi_x \rangle$ = z. O motivo da escolha mostrada acima se deve às fases a serem adicionadas. Como os valores $A_{w,b}$ e A_x não aparecem no produto interno, suas fases não são relevantes. Protanto, tais estados $|\Psi_x\rangle$ e $|\Psi_{w,b}\rangle$ tornam desnecessaria a aplicação de (n-1) portas Z controladas para ajustar as fases das amplitudes associadas aos estados $|0\rangle$ e $|N-1\rangle$. Na Figura 3 a composição das transformações U_x e $U_{w,b}$ são mostradas no caso de um perceptron de uma camada com $N_{in} = 4$ neurônios. Nesse caso, o número de neurônios de entrada é 4. Como $n = \log_2 N$

e $N \geq N_{in}+3$ então, dado N_{in} nurônios de entrada o número mínimo de qubits necessário é $n = \lceil \log_2(N_{in}+3) \rceil$. Portanto, com $N_{in} = 4$, n=3 qubits são necessários para armazenar z em um estado quântico.

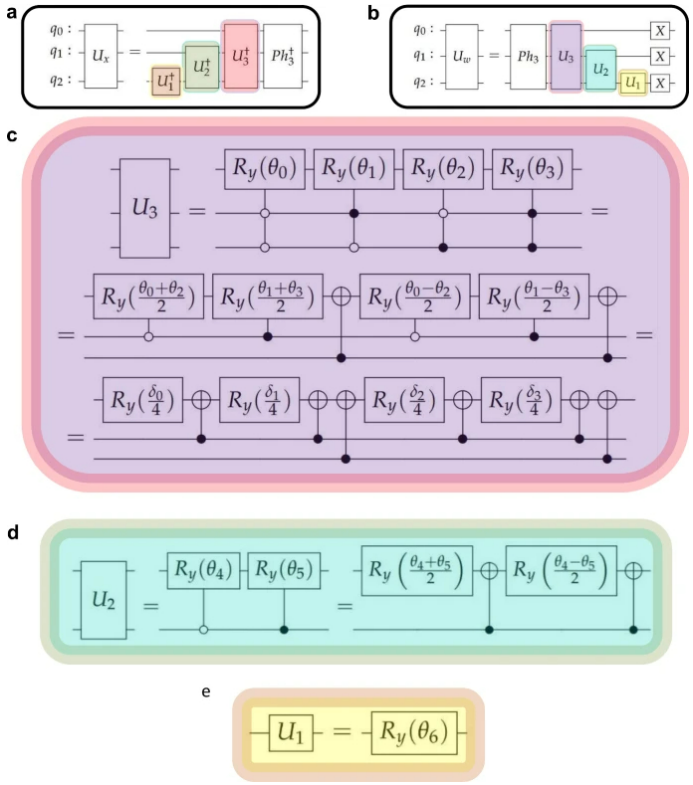


Figura 3: Composição fundamental dos portões das transformações $U_{w,b}$ e U_x. As transformações $U(v_x)$ e $U(v_{w,b})$ codificam, respectivamente, os coeficientes dos vetores v_x e $v_{w,b}$ em um estado quântico de superposição. Eles são compostos pelo inverso dos operadores U_i onde i=1,...,n e Ph3, que introduz as fases das amplitudes de probabilidade. Os operadores U3(c), U2(d) e U1(e) são mostrados como composição de rotações multicontroladas Ry que são iguais a uma composição de portões Ry e portões Not controlados [40].

A variável z generaliza em dois aspectos o produto interno da Ref. [27] onde as entradas e os pesos assumem apenas valores binários $\{-1, 1\}$ e nenhum viés está envolvido.

A transformação $U_z(\vec{x}, \vec{w}, b)$ é um bloco de construção chave do algoritmo do perceptron quântico. De fato, no circuito quântico, tal transformação é repetida várias vezes no espaço de Hilbert ampliado para $H_a^{\otimes d} \otimes H_q^{\otimes n}$ pela adição de outro registrador a de d qubits. A existência desse circuito é garantida pelo Teorema 1 em **??**

4.4 Teoremas

Teorema 1: Seja z um valor real no intervalo [-1, 1] assumido por $(\vec{x}\cdot\vec{w}+b)/(N_{in}-1)$ onde x, w $\in [-1,1]^{N_{in}}$ e b $\in$ [-1, 1]. Sejam q e a dois registradores de n e d qubits, respectivamente, com N $= 2^n \geq N_{in}+3$. Em seguida, existe um circuito quântico que transforma os dois resgistradores do estado inicial $|0\rangle_a\ |0\rangle_q$ em um estado emaranhado $|\varphi^d\rangle$ com (n + d)-qubits da forma

$$|\varphi_z^d\rangle = |\varphi_z^d\rangle_\perp + \frac{1}{2^{\frac{d}{2}}}|z\rangle_a^{\otimes d}|N-1\rangle_q \tag{4.6}$$

onde

$$|N-1\rangle\langle N-1|_q|\varphi_z^d\rangle_\perp = \mathbf{0} \tag{4.7}$$

onde $\mathbf{0}$ é o elemento nulo do espaço de Hilbert e

$$|z\rangle \equiv |0\rangle + z|1\rangle \tag{4.8}$$

O circuito é expresso por $S_V X_q^{\otimes n}$ onde X é o portão quântico NOT e

$$S_V = V_{d-1}, ..., V_1, ..., V_0 \tag{4.9}$$

com

$$V_m = C_{a_m}U_z(\vec{x},\vec{w},b)_q C_{a_m} X_q^{\otimes n} C_q^n H_{a_m}, m = 0, 1, ..., d-1. \tag{4.10}$$

Prova: A tese do teorema é a existência de uma transformação que, atuando em dois registradores q e a com n e d qubits, respectivamente, retorna um estado $|\psi_z^m\rangle \in H_a^{\otimes n} \otimes H_q^{\otimes n}$ como definido na Equação 4.6. A demonstração consiste na construção de tal circuito. Para tal propósito, vamos definir os d estados $|\psi_z^m\rangle \in H_{a_{m-1}} \otimes H_{a_{m-2}} \otimes ... \otimes H_{a_0} \otimes H_q^{\otimes n}$, onde $m = 0, ..., d-1$.

$$|\psi_z^m\rangle = |\psi_z^m\rangle_\perp + |\psi_z^m\rangle_{||} = |\psi_z^m\rangle_\perp + \frac{1}{2^{m/2}}|z\rangle^{\otimes m}|N-1\rangle \tag{4.11}$$

onde $|N-1\rangle\langle N-1|_q|\psi_z^m\rangle_\perp = 0\ \forall\ m$. A partir de tal definição, segue-se que os estados $|\psi_z^m\rangle$ são estados de $(n+m)$-qubits e $|\psi_z^0\rangle \equiv |N-1\rangle_q$ é um estado de n-qubits.

A prova do teorema é, portanto, reduzida a demonstrar a existência de uma sequência

de transformações V_m, onde $m = 0, ..., d-1$, de tal forma que $V_m|0\rangle_{a_m}|\psi_z^m\rangle = |\psi_z^{m+1}\rangle$ onde a_m é o m-ésimo qubit no registador a. Portanto, V_m é uma transformação unitária definida sobre o espaço $H_{a_m} \otimes H_{a_{m-1}} \otimes ... \otimes H_{a_0} \otimes H_q^{\otimes n}$. Vamos considerar o seguinte ansatz para a transformação V_m:

$$V_m = C_{a_m} U_z(\vec{x}, \vec{w}, b)_q C_{a_m} X_q^{\otimes n} C_q^n H_{a_m} \tag{4.12}$$

cuja representação gráfica é dada na Figura 3-**a**.

Vamos aplicar V_m, como definido na Equação 4.12, no estado $|0\rangle_{a_m}|\psi_z^m\rangle$.

$$\begin{aligned} &C_{a_m} U_z(\vec{x}, \vec{w}, b)_q C_{a_m} X_q^{\otimes n} C_q^n H_{a_m} |0\rangle_{a_m}|\psi_z^m\rangle = C_{a_m} U_z(\vec{x}, \vec{w}, b)_q C_{a_m} = \\ &= X_q^{\otimes n}[|0\rangle_{a_m}|\psi_z^m\rangle_\perp + 1/\sqrt{2}(|0\rangle_{a_m} + |1\rangle_{a_m})|\psi_z^m\rangle_{||}] \end{aligned} \tag{4.13}$$

A transformação $C_{a_m} U_z(\vec{x}, \vec{w}, b)_q C_{a_m} X_q^{\otimes n}$ consiste na aplicação de $U_z(\vec{x}, \vec{w}, b) X^{\otimes n}$ nos qubits q, essa operação é controlada pelo quibit a_m que significa que a transformação age somente em $|\phi_z^m\rangle_{||}$ focando em seus subespaços que resulta em:

$$U_z(\vec{x}, \vec{w}, b)_q X_q^{\otimes n}|\psi_z^m\rangle_{||} = 1/2^{m/2}|z\rangle^{\otimes m} U_z(\vec{x}, \vec{w}, b)|0\rangle_q^{\otimes n} \tag{4.14}$$

Portanto, a transformação V_m aplicada em $|0\rangle_{a_m}|\psi_z^m\rangle$ retorna o seguinte estado

$$\begin{aligned} &V_m|0\rangle_{a_m}|\psi_z^m\rangle = |0\rangle_{a_m}|\psi_z^m\rangle_\perp + 1/\sqrt{2}(|0\rangle_{a_m} 1/2^{m/2}|z\rangle^{\otimes m}|N-1\rangle_q + \\ &+ |1\rangle_{a_m} 1/2^{m/2}|z\rangle^{\otimes m} U_z(\vec{x}, \vec{w}, b)|0\rangle_q^{\otimes m}) \end{aligned} \tag{4.15}$$

Para demonstrar que o estado apenas obtido é $|\psi_z^{m+1}\rangle$, a projeção sobre o estado $|N-1\rangle_q$ deve retornar $\frac{1}{\sqrt{2^{m+1}}}|z\rangle^{\otimes(m+1)}|N-1\rangle_q$ a partir da definição do estado $|\psi_z^m\rangle$. Vamos aplicar a projeção $|N-1\rangle\langle N-1|_q$ no estado resultante na Equação 4.15. Como $|N-1\rangle\langle N-1|_q|\Psi_z^m\rangle_\perp = 0$ por definição, $\langle N-1|U(\vec{x}, \vec{w}, b)|0\rangle_q^{\otimes n} = z$ a partir do Lema 1, o resultado da projeção $|N-1\rangle\langle N-1|_q$ é a seguinte.

$$\frac{1}{\sqrt{2}}(|0\rangle_{a_m}\frac{1}{\sqrt{2}}|z\rangle^{\otimes m}|N-1\rangle_q + z|1\rangle_{a_m}\frac{1}{2^{\frac{m}{2}}}|z\rangle^{\otimes m}|N-1\rangle_q) = \frac{1}{2^{m+1}}|z\rangle^{\otimes(m+1)}|N-1\rangle_q = |\Psi_z^{m+1}\rangle_{||} \tag{4.16}$$

Tendo demonstrado que $V_m|0\rangle_{a_m}|\Psi_z^m\rangle = |\Psi^{m+1}\rangle$, a prova da existencia de tal tansformação que retorn $|\Psi_z^d\rangle$ aplicada em $|0\rangle_a^{\otimes d}|0\rangle_q^{\otimes n}$ procede por recursão. De fato, aplicando

$V_{d-1}\dots V_1V_0$ no estado $|0\rangle_a^{\otimes d}|\Psi_z^0\rangle = |0\rangle_a^{\otimes d}|N-1\rangle_q$ o resultado será o estado $|\Psi_z^d\rangle$.

Para resumir, o circuito quântico do algoritmo perceptron quântico começa expressando o operador unitário que inicializa os registradores q e a partindo do estado $|0\rangle_a|0\rangle_q$ para o estado $|\Psi_z^d\rangle$. Esse operador unitário é expresso por $S_vX^{\otimes n}$ onde S_v é a sub-rotina do circuito quântico que atinge o objetivo da primeira etapa do algoritmo de perceptron quântico, ou seja, codificar as potências de z até d em um estado quântico a partir do seguinte Corolário.

Colorário 1: O estado $|\Psi_z^d\rangle$ guarda as probabilidades das amplitudes para todas as potências em z^k, para $k = 0, 1, \dots, d$, até um fator trivial. De fato, na equação 4.6 do teorema 1 em implica

$$_q\langle N-1|_a\langle 2^k-1|\Psi_z^d\rangle = 2^{\frac{-d}{2}}z^k, k = 0, 1, \dots, d. \tag{4.17}$$

O primeiro passo do algoritmo do perceptron quântico consiste no armazenamento de todas as potências de $z \equiv \frac{(\vec{x},\vec{w},b)}{N_{in}+1}$ até a d-ésima ordem em um estado de (n+d)-qubits. A prova do Teorema 1 implica que o primeiro passo do algoritmo é o circuito quântico mostrado na Figura 4-a, consistindo em uma sub-rotina composta por uma porta Pauli X aplicada em cada qubit no registrador q e uma transformação $S_v = V_{d-1}\dots V_0$. De fato, a partir do Corolário 1, o estado $|\Psi_z^d\rangle = S_vX_q^{\otimes n}|0\rangle_a^{\otimes d}|0\rangle_q^{\otimes n}$ armazena como amplitudes de probabilidade todas as potências de z até a d-ésima ordem menos um fator $2^{\frac{-d}{2}}$. A prova do Colorário 1 segue que.

Prova: Como mostrado acima, o estado $|\Psi_z^d\rangle$ pode ser escrito como $|\Psi_z^d\rangle = |\Psi_z^d\rangle_\perp + |\Psi_z^d\rangle_\parallel$ onde $|N-1\rangle\langle N-1|_q|\Psi_z^d\rangle_\perp = 0$, portanto, $_q\langle N-1|_a\langle 2^k-1|\Psi_z^d\rangle =_q \langle N-1|_a\langle 2^k-1|\Psi_z^d\rangle_\perp$.

Como $|\Psi_z^d\rangle_\parallel = \frac{1}{\sqrt{2^d}}|z\rangle_a^{\otimes d}|N-1\rangle_q$ então

$$_q\langle N-1|_a\langle 2^k-1|\Psi_z^d\rangle_\parallel = \frac{1}{\sqrt{2^d}}\langle 2^k-1|z\rangle_a^{\otimes d}\langle N-1|N-1\rangle_q \tag{4.18}$$

Vamos reescrever $|2^k-1\rangle$ de forma binária $|s_{d-1}s_{d-2}\dots s_0\rangle$ onde $s_j = 1$ de $j = 0$ até $j = d-1$ e $s_j = 0$ caso contrário.

$$\frac{1}{\sqrt{2^d}}\langle s_0, s_1, \dots, s_{d-1}|z\rangle_a^{\otimes d} = \frac{1}{\sqrt{2^d}}\prod_{i=1}^{d}\langle s_{d-i}|z\rangle_{a_{d-1}} = 2^{d/2}z^k \tag{4.19}$$

Este último é válido porque, $\forall j = 0, \dots, d-1$,

$$\langle s_j|z\rangle_{a_j} = \langle s_j|0\rangle_{a_j} + z\langle s_j|1\rangle_{a_j} = z^{s_{aj}} \tag{4.20}$$

Portanto, ${}_q\langle N-1|_a\langle 2^k-1|\Psi_z^d\rangle \equiv 2^{-d/2}z^k$

O próximo passo do algoritmo consiste em transformar o estado $|\Psi_z^d\rangle$ de modo a obter um polinômio especial de d graus recursivamente definido em z Tal passo é identificável com a sub-rotina S_u do circuito do perceptron quântico, ver Figura 4**a**

Pela equação 4.6 no Teorema 1 deve existir um operador unitário S_u que age como a identidade em H_q e retorna, quando aplicado à $|z\rangle_a^{\otimes d}$, um novo estado que guarda o ponomio. Na verdade, detém o seguinte. [41]

Teorema 2: Seja $\{f_k, k = 1, \ldots, d\}$ a família de polinomios em z definida pela seguinte lei recursiva

$$f_k(z) = f_{k-1}(z)\cos\vartheta_{k-1} - z^k \sin\vartheta_{k-1}, k = 1, \ldots, d, \tag{4.21}$$

com $f_0(z) = 1$ e $\vartheta_k \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ para qualquer $k = 0, \ldots, d-1$.

Então existe uma família $\{U_k, k = 1, \ldots, d\}$ de operadores unitários tal que

$${}_a\langle 0|U_k|z\rangle_a^{\otimes d} = f_k(z) \tag{4.22}$$

Esses operadores unitários são definidos pela lei recursiva

$$U_k = C_{a0}X_{ak}\bar{C}_{ak}R_y(-2\vartheta_{k-1})_{a0}U_{k-1}, k = 1, \ldots, d, \tag{4.23}$$

com $U_0 = 1$

A sub-rotina S_u demonstrada na Figura 4**a** corresponde a U_d. A prova do teorema 2 segue.

Prova: A prova desse teorema segue dois passos. O primeiro passo consistem em demosntrar que

$${}_a\langle 0|U_1|z\rangle_a^{\otimes d} = \cos\theta_0 - z\sin\theta_0 = f_1(z) \tag{4.24}$$

com $U_1 = C_{a0}X_{a1}\bar{C}_{a1}R_y(-2\vartheta_0)_{a0}$ como definido na equação 4.23. No segundo passo, a prova procede para $U_k \forall_k = 1, 2, \ldots, d$ recursivamente.

O objetivo é provar que ${}_a\langle 0|U_k|z\rangle_a^{\otimes d} = f_k(z)$ assumindo que ${}_a\langle 0|U_{k-1}|z\rangle_a^{\otimes d} = f_{k-1}$ onde

$$U_k = C_{a0}X_{ak}\bar{C}_{ak}R_y(-2\vartheta_{k-1})_{a0}U_{k-1}$$

como definido na equação 4.23. Vamos considerar o estado $U_k|z\rangle_a^{\otimes d}$, onde $d \geq k \geq 1$. O estado $|z\rangle_a^{\otimes d}$ é considerado no caso com $k = 0$. Depois, focamos no subespaço $H_a^{\otimes d}$ definido como $H_{\{0,1\}} = \{|0\rangle^{\otimes d}, |0\rangle^{\otimes(d-1)}|1\rangle\}$. O operador que realiza a projeção dos elementos de $H_a^{\otimes d}$ em $H_{\{0,1\}}$ é $P_{\{0,1\}} = |0\rangle\langle 0|_a + |1\rangle\langle 1|_a$.

Vamos agora para a primeira etapa da demonstração. A primeira operação consiste em aplicar U_1 ao estado $|z\rangle_a^{\otimes d}$. Por conta da definição do estado $|z\rangle_a^{\otimes d}$, segue que $\langle 2\hat{i}-1|z\rangle_a^{\otimes d} = z^i$ onde $i = 0, \ldots, d$ (Colorário 1), portanto, a projeção sobre $H_{\{0,1\}}$ do estado $|z\rangle_a^{\otimes d}$ é

$$P_{\{0,1\}}|z\rangle_a^{\otimes d} = |0\rangle^{\otimes d} + z|0\rangle^{\otimes(d-1)}|1\rangle \tag{4.25}$$

o operador $\bar{C}_{a_1}R_y(-2\vartheta_0)_{a_0} = X_{a_1}C_{a_1}R_y(-2\vartheta_0)X_{a_0}$ rotaciona o qubit a_0 pelo eixo y da esfera de Bloch com angulo $-2\vartheta_0$, somente se o qubit a_1 estiver no estado $|0\rangle$, portanto, tal operador age no subespaço $H_{\{0,1\}}$. A projesão nesse subespaço do estado $\bar{C}_{a1}R_y(-2\vartheta_0)_{a0}|z\rangle_a^{\otimes d}$ é

$$(\cos\theta_0 - z\sin\theta_0)|0\rangle^{\otimes d} + (\sin\theta_0 + z\cos\theta_0)|0\rangle^{\otimes(d-1)}|1\rangle = f_1(z)|0\rangle^{\otimes d} + (sin\theta_0 + z\cos\theta_0)|0\rangle^{\otimes(d-1)}|1\rangle \tag{4.26}$$

como $C_{a0}X_{a1}$ é uma porta NOT controlada que age somente se o qubit a_0 estiver no estado $|1\rangle$ então

$${}_a\langle 0|U_1|z\rangle_a^{\otimes d} =_a \langle 0|C_{a0}X_{a1}\bar{C}_{a1}R_y(-2\vartheta_0)|z\rangle = \cos\theta_0 - z\sin\theta_0 = f_1(z) \tag{4.27}$$

o que completa o primeiro passo da demonstração. Agora passamos para o passo recursivo.

Aqui, a única hipótese é que ${}_a\langle 0|U_{k-1}|z\rangle_a^{\otimes d} = f_{k-1}(z)$, então, diferete do primeiro passo onde a projeção de $|z\rangle_a^{\otimes d}$ no subespaço $H_{\{0,1\}}$ era conhecido, dessa vez a projeção de $U_{k-1}|z\rangle_a^{\otimes d}$ é igual a

$$f_{k-1}(z)|0\rangle^{\otimes d} + B_{k-1}|0\rangle^{\otimes d}|1\rangle \tag{4.28}$$

onde B_{k-1} é um valor real desconhecido. Vamos aplicar $C_{a0}X_{ak}\bar{C}_{ak}R_y(-2\vartheta_{k-1})_{a0}$ no estado $U_{k-1}|z\rangle_a^{\otimes d}$ para obter o estado $U_k|z\rangle_a^{\otimes d}$. Da Equação 4.28:

$$_a\langle 0|U_k|z\rangle_a^{\otimes d} = f_{k-1}(z)\cos\vartheta_{k-1} - B_{k-1}\sin\vartheta_{k-1} \tag{4.29}$$

Para provar o Teorema, B_{k-1} tem que ser igual a z^k já que $f_k(z) = f_{k-1}(z)\cos\vartheta_{k-1} - z^k\sin\vartheta_{k-1}$. O propósito do segundo passo da prova pode ser alcançado provando somente que $B_{k-1} = z^k \forall k = 1,\ldots,d$. Isso já está provado para $k = 0$ porque $\langle 2^i - 1|z\rangle_d^{\otimes d} = z^i$ como demonstrado acima. Agora vamos provar que $B_{k-1} = z^k$ para $k = 1$ enquanto que para $k > 1$ a prova segue recursivamente.

O estado $|2^i - 1\rangle$ é um estado da base computacional $H_a^{\otimes d}$. Escrevendo este estado na forma binária ficamos com $|S_{d-1}S_{d-2}\ldots S_0\rangle$ onde $S_j = 1$ de $j = 0$ até $j = i - 1$ e 0 caso contrário. Como dito anteriormente o operador $\bar{c}_{a_1}R_y(-2\vartheta_0)_{a_0}$ age no estado $|S_{d-1}S_{d-2}\ldots S_0\rangle$ onde $S_1 = 0$, portanto, não age no estado $|2^i - 1\rangle \forall i > 1$. Ao inv´s disso, o operador $C_{a_0}X_{a_1}$ age no estado $|S_{d-1}S_{d-2}\ldots S_0\rangle$ onde $s_0 = 1$ e aplica uma operação NOT no bit s_1. Isso significa que o estado $|2^i - 1\rangle$ se torna $|2^i - 1 - 2\rangle \forall i > 1$. Então, já que $\langle 2^i - 1|z\rangle = z^i$, graças ao $U_1(\vartheta_0) = C_{a_0}X_{a_1}\bar{C}_{a_1}R_y(-2\vartheta_0)_{a_0}$ então $\langle 2^i - 1 - 2|U_1|z\rangle_a \otimes d = z^2$ e $B_1 = z^2$. Agora procedemos para a parte recursiva $k > 1$ assumindo que

$$\langle 2^i - 1 - \sum_{h=1}^{k-1} 2^h |U_{k-1}(\vec{\vartheta_{k-2}})|z\rangle = z^i \tag{4.30}$$

onde $d > i \geq k$. O estado $|2^i - 1 - \sum_{h=1}^{k-1} 2^h\rangle$ escrito de forma binária é $|S_{d-1}S_{d-2}\ldots S_0\rangle$ onde $S_j = 1$ de $j = 0$ até $j = i - 1$ e 0 caso contrário. Em particular, para $i = k$, $|2^i - 1 - \sum_{h=1}^{k-1} 2^h\rangle = |0\ldots 01\rangle$ e portanto $_a\langle 1|U_{k-1}|z\rangle_a^{\otimes d} = z^i$ que significa que $B_{k-1} = z^k$. A recursão consiste em provar que $\langle 2^i - 1 - \sum_{h=1}^{k-1} 2^h|U_{k-1}|z\rangle_a^{\otimes d} = z^i$ a partir da suposição na Equação 4.30. Começando pelo estado $U_{k-1}|z\rangle_a^{\otimes d}$ e aplicamos na transformação $C_{a0}X_{ak}\bar{C}_{ak}R_y(-2\vartheta_{k-1})_{a0}$. A transformação $\bar{C}_{ak}R_y(-2\vartheta_{k-1})_{a0}$ age no estado $|S_{d-1}S_{d-2}\ldots S_0\rangle$ onde $S_k = 0$, portanto, não age no estado $|2^i - 1 - \sum_{h=1}^{k-1} 2^h\rangle \forall i > k$. Em vez disso $C_{a0}X_{ak}$ é uma transformação bit-flip que atua no estado $|S_{d-1}S_{d-2}\ldots S_0\rangle$ somente se $S_0 = 1$ e aplica operações NOT no qubit S_k, então

$$C_{a0}X_{ak}|2^i - 1 - \sum_{h=1}^{k-1} 2^h\rangle = |2^i - 1 - \sum_{h=1}^{k-1} 2^h\rangle \tag{4.31}$$

isso significa que

$$\langle 2^i - 1 - \sum_{h=1}^{k-1} 2^h|U_k(\vec{\vartheta_{k-1}})|z\rangle = z^i \tag{4.32}$$

para $d > i > k$ e, em particular, para $i = k{+}1$,$_a\langle 1|U_{k-1}|z\rangle_a^{\otimes d} = z^k$ portanto $B_k = k^{k+1}$. Este resultado final prova que $B_k = z^{k+1} \forall k = 0, \ldots, d-1$, então

$$_a\langle 1|U_{k-1}|z\rangle_a^{\otimes d} = f_{k-1}(z)\cos\vartheta_{k-1} - B_{k-1}\sin\vartheta_{k-1} = f_k(z) \tag{4.33}$$

A segunda etapa está do teorema está concluída e, assim, a prova do teorema.

Na Figura 4-**a**, S_u é a sub-rotina que atinge o segundo passo do algoritmo perceptron, a composição da expansão polinomial em z, e é igual a $U_d(\vec{\vartheta_{d-1}})$. [41]

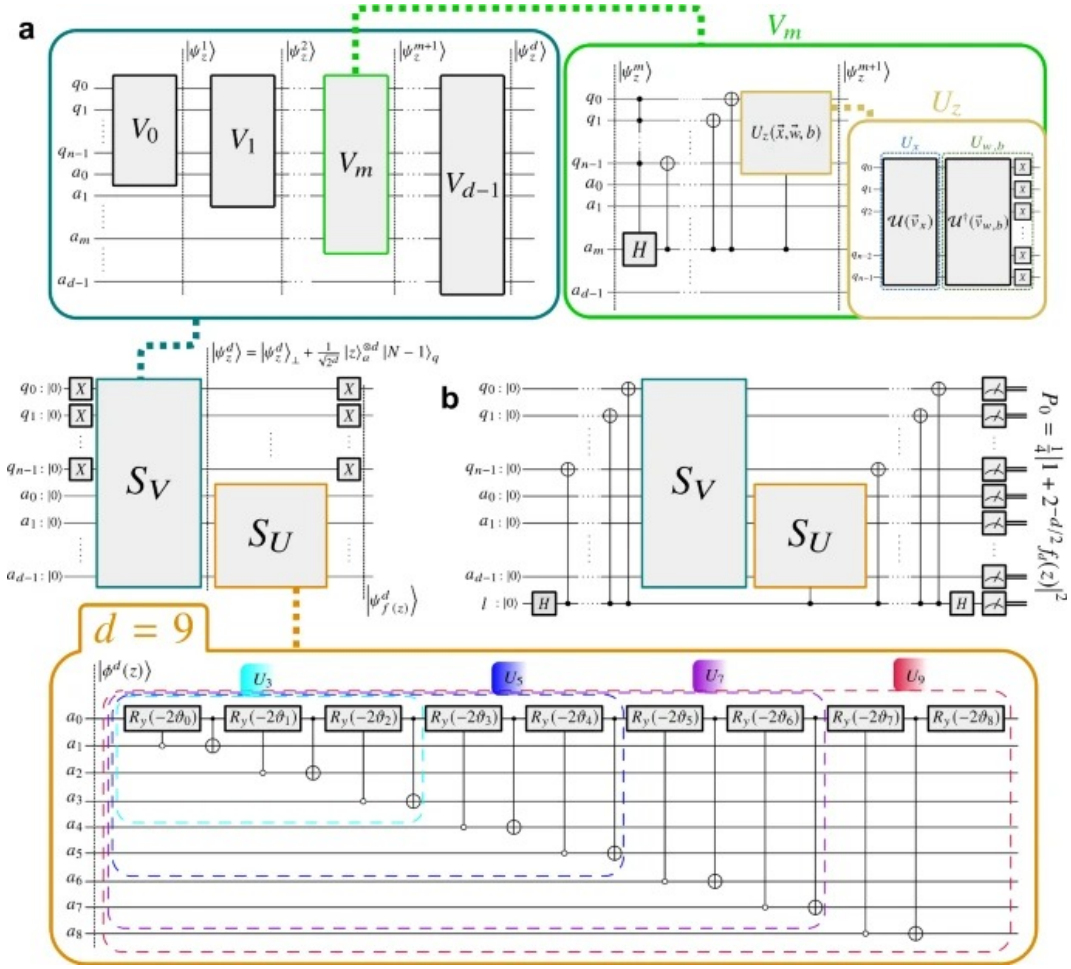


Figura 4: Circuito quântico de um perceptron de uma camada baseado em qubits em dois casos diferentes. **(a)** O circuito quântico que retorna o estado $|\Psi^d_{f(x)}\rangle$ codifica o valor $f_d(z)$ como indicado pelos Teoremas 1 e 2 . Em um contexto mais geral, $f_d(z)$ é o valor de saída de um neurônio de uma camada oculta de rede neural profunda, pois nenhuma medição é necessária quando a informação é enviada para a próxima camada. As sub-rotinas S_v e S_u são descritos nas caixas superiores, onde o caso geral de S_v é mostrado, e na caixa inferior, respectivamente. Neste último, mostra-se um caso particular em d=9. Tal valor corresponde à ordem máxima de d=9 da expansão polinomial utilizada para aproximar as funções de ativação tanh, sigmoid e seno. **(b)** Um circuito quântico completo de um perceptron de uma camada baseado em qubit é mostrado. Nesse caso, realizando as medições de todos os qubits e calculando a média após muitas repetições do circuito, é possível estimar a saída y_q do perceptron de uma camada

A transformação $S_u = U_d \otimes X^{\otimes n}$ aplicada a $|\psi^d_z\rangle$ retorna um estado com probabilidade de amplitude associada a $|0\rangle_a|0\rangle_q$ igual a $2^{\frac{-d}{2}} f_d(z)$. Vamos denotar tal estado quântico final de $(n+d)$-qubits como $|\Psi^d_{f(z)}\rangle$. A Equação 4.21 define $f_d(z)$ como um polinomio de grau d com coeficientes dependendo dos d ângulos $\theta_k, k = 0, \ldots, d-1$. A partir de Teorema 2, segue-se um Corolário que mostra como definir tais ângulos a fim de aproximar uma função de ativação analítica arbitrária $f(z)$ por $f_d(z)$.

Colorário 2 Seja f uma função analítica real durante um intervalo compacto I. Se fd é o membro superior da família de polinômias definida no Teorema 2, na Equação 4.21 então os ângulos $\theta_k, k = 0, \ldots, d-1$ podem ser escolhidos de tal forma que $f_d(z)$ coincide com a d-ésima ordem da expansão de Taylor $f(z)$ em torno de $z = 0$, até um fator constante C_d que depende de f e da ordem d como

$$C_d = a_k \prod_{j=k}^{d-1} (\cos\theta_j)^{-1}, \tag{4.34}$$

onde $a_k = \frac{1}{k!} f^{(k)}(0)$ é o primeiro coeficiente não-zero da expansão e $f^{(k)}$ o k-ésimo derivativo de f.

Prova: Vamos denotar com $T_d(z)$ a expansão truncada da série polinomial de uma função analítica f na ordem d expressa por $T_d(z) = \sum_{i=0}^{d} a_i z^i$.

Seja f uma função analítica então existe um k, onde $0 \leq k \leq d$ e $a_i = 0 \forall i < k$, de tal forma que, fatorando a_k, $T_d(z)$ resultada em

$$T_d(z) = a_k [z^k + \sum_{i=k}^{d-1} \frac{a_i + 1}{a_k} z^{i+1}] \tag{4.35}$$

Vamos considerar o valor $f_d(z)$, definido pela Equação 4.21. Se $\cos\theta_i \neq 0 \forall i = k, \ldots, d-1$ e $\theta_i = -\frac{\pi}{2}$ caso contrário, então, fatorizando qualquer $\cos\theta_i \neq 0$, pode ser expresso por

$$f_d(z) = A_{dk} [z^k + \sum_{i=k}^{d-1} \frac{\tan\theta_i}{A_{ik}} z^{i+1}] \tag{4.36}$$

onde $A_{ik} = \prod_{j=k}^{i-1} \cos\theta_j$. Vamos escolher os ângulos θ_i para satisfazer a equidade

$$T_d(z) = \frac{a_k}{A_{dk}} f_d(z) \tag{4.37}$$

Portanto, equalizando termo por termo em poderes de z, a equação resultante para $i \geq k$ é

$$\theta_i = \arctan(-\frac{a_{i+1}}{a_k} A_{dk}) \tag{4.38}$$

Como os valores A_{ik} dependem dos ângulos $\theta_k, \ldots, \theta_{i-1}$ então os ângulos θ_i por sua vez, dependem deles. Significa que o cálculo de todos os ângulos deve ser ordenado de θ_k

a θ_{d-1}. A partir dessa definição dos ângulos θ_i, onde $i = 0, \ldots, d-1$, a Equação 4.42 está satisfeita. Portanto $f_d(z)$ é igual à expansão de série de Taylor de $f(z)$ na ordem d menos do que um fator constante C_d

$$C_d = \frac{a_k}{A_{dk}} = \frac{a_k}{\prod_{j=k}^{d-1} \cos\theta_j} \tag{4.39}$$

Seu valor é constante enquanto z muda, e depende dos coeficientes a_i da expansão Taylor da função f onde $i = k, \ldots, d$. [41]

4.5 **Cálculo Inicial**

Para realizar o desenvolvimento do algoritmo proposto, começamos com o cálculo das variáveis A_x e $A_{w,b}$ descritas no Lema 1. Pela definição dada na prova do Lema, temos que $\vec{v}_x \dot{\vec{v}}_x = \vec{v}_{w,b} \dot{\vec{v}}_{w,b} = N-1$, substituindo $\vec{v}_x$ e $\vec{v}_{w,b}$ por $(\vec{x}, 1, A_x, 0)$ e $(\vec{w}, b, 0, A_{w,b})$, respectivamente, temos

$$\begin{aligned} &{A_x}^2 + \{x_0, \ldots, x_{N_{in}-1}\}^2 + A_x(2\{x_0, \ldots, x_{N_{in}-1}\} + 2) + \\ &+ 2\{x_0, \ldots, x_{N_{in}-1}\} + 1 = N_{in} - 1 \end{aligned} \tag{4.40}$$

$$\begin{aligned} &{A_{w,b}}^2 + \{w_0, \ldots, w_{N_{in}-1}\}^2 + A_{w,b}(2\{w_0, \ldots, w_{N_{in}-1}\} + 2) + \\ &+ b^2 + 2b(\{w_0, \ldots, w_{N_{in}-1}\}) = N_{in} - 1 \end{aligned} \tag{4.41}$$

podemos ver que são duas equações de segundo grau e podemos usar a formula de Bhaskara para achar as raízes dessas equações e calcular os valores de A_x e $A_{w,b}$. Reorganizando as equações, obtemos os termos a, b e c da formula de Bhaskara para cada uma delas, $a_x = 1$, $b_x = 2(x_0, \ldots, x_{N_{in}}) + 2$ e $c_x = (x_0, \ldots, x_{N_{in}})^2 + 2(x_0, \ldots, x_{N_{in}}) + 2 - N_{in}$ e $a_{w,b} = 1$, $b_{w,b} = 2(w_0, \ldots, w_{N_{in}}) + 2b$ e $c_{w,b} = b^2 + (x_0, \ldots, x_{N_{in}})^2 + 2b(x_0, \ldots, x_{N_{in}}) + 1 - N_{in}$. Podemos ver que as variáveis A_x e $A_{w,b}$ só dependem dos valores dos vetores $\vec{x}, \vec{w}$ e do viés b.

Com o valor das variáveis podemos construir nossos vetores de entrada $\vec{v}_x = (\vec{x}, 1, A_x, 0)$ e peso $\vec{v}_{w,b} = (\vec{w}, b, 0, A_{w,b})$ onde o 0 representa a quantidade de $0's$ para que o tamanho do vetor seja igual a N onde $N = 2^n$ com $N \geq N_{in} + 3$ e n sendo a quantidade de qubits do circuito.

Tendo definido os vetores de entradas e pesos, o próximo passo é a construção do circuito quântico para obter os estados iniciais $|\Psi_x\rangle$ e $|\Psi_{w,b}\rangle$. O algoritmo realmente começa

a partir do estado alvo $|\Psi_v\rangle$ e constrói uma transformação unitária que o evolui de volta para $|0\rangle$. O circuito de inicialização é, portanto, o inverso disso. Para transformar $|\Psi_v\rangle$ em $|0\rangle$, pode-se proceder recursivamente desembaraçando um qubit de cada vez. Considere primeiro o qubit q_0 menos significativo no registrador. Exploramos o seguinte teorema em [31]

Teorema 3: Dado um registrador q de n qubits num estado arbitrário $|\Psi\rangle \in H_q^{\otimes n}$, existe um conjunto de ângulos $\{\theta_j, \psi_j, j = 0, 1, \ldots, N, N = 2^{n-1}\}$, tal que o bloco-diagonal da matriz unitária

$$U_n(\{\theta_j\}, \{\psi_j\}) = \oplus_{j=0}^{N-1} R_y(-\theta_j) R_z(-\psi_j) \tag{4.42}$$

trasnforma $|\psi\rangle$ no produto direto $|\psi'\rangle \otimes |0\rangle$, onde $|\psi'\rangle \in H_q^{\otimes n}$. Fica claro que existe outro conjunto de ângulos $\{\theta'_j, \psi'_j, j = 0, 1, \ldots, N', N' = 2^{n-1}\}$ e transformação unitária $U_{n-1}(\{\theta'_j\}, \{\psi'_j\})$ que podem ser encontrados para desemaranhar q_1 e resultar num estado de forma $|\psi''\rangle \otimes |00\rangle$. O procedimento é então repetido até que todos os qubits sejam desembaraçados e o estado do qubit q_n mais significativo seja girado para $|0\rangle$, eventualmente gerando o estado $|0\rangle^{\otimes n}$. Ao todo, precisamos de n conjuntos de ângulos da forma

$$\{\theta_{j,m}, \psi_{j,m}, j = 0, \ldots, 2^{n-m-1} - 1, m = 0, 1, \ldots, n-1\} \tag{4.43}$$

com os operadores associados

$$U_{n,m}(\{\theta_{j,m}, \psi_{j,m}\}) \equiv U_{n-m}(\{\theta_{j,m}, \psi_{j,m}\}) \otimes 1^{\otimes m} \tag{4.44}$$

onde a definição da transformação U_{n-m} é dada pela Equação 4.42.

Portanto, o inverso do produto do operador

$$\mu_n(\{\theta_{j,m}, \psi_{j,m}\}) \equiv \prod_{m=1}^{n} U_{n,n-m}(\{\theta_{j,m}, \psi_{j,m}\}) \tag{4.45}$$

inicializa um registrador de n qubits para o estado $|\psi_v\rangle$, uma vez que uma fase de pré-processamento é executada para calcular os ângulos $(\{\theta_{j,m}, \psi_{j,m}\})$ a partir do vetor $\vec{v}$. Para o caso de dados quânticos, os ângulos das esferas de Bloch dos qubits já são fornecidos, portanto, não é necessária uma fase de pré-processamento para calculá-los.

Se $\vec{v} = (v_0, v_1, \ldots, v_{N-1})$ é um vetor real como no caso dos vetores $\vec{v}_x$ e $\vec{v}_{w,b}$ descritos no Teorema 1 em **??**, então a matriz unitária $\mu(\vec{v})$ que transforma $|0\rangle^{\otimes n}$ em $|\psi_v\rangle$ pode ser fatorada em duas matrizes de acordo com

$$\mu(\vec{v}) = Ph_n(\vec{v})\mu_n^\dagger(\{\theta_{j,m}, \psi_{j,m} = 0\}) \tag{4.46}$$

onde, como $\mu_n^\dagger(\{\theta_{j,m}, \psi_{j,m} = 0\})|0\rangle^{\otimes n} = |\psi_v\rangle$,

$$\mu_n^\dagger(\{\theta_{j,m}, \psi_{j,m} = 0\})|0\rangle^{\otimes n} = |\psi_{v+}\rangle \equiv \sum_{i=0}^{N-1} |v_i\rangle|i\rangle \tag{4.47}$$

e

$$Ph_n(\vec{v})|\psi_{v+}\rangle = \sum_{i=0}^{N-1} \frac{v_i}{|v_i\rangle} v_i|i\rangle = |\psi_v\rangle \tag{4.48}$$

é a transformação diagonal que introduz os sinais negativos corretos dos coeficientes v_i.

A transformação $\mu_n^\dagger(\{\theta_{j,m}, \psi_{j,m} = 0\})$ depende somente no conjunto de ângulos $\{\theta_{j,m}\}$ e, considerando a Equação 4.45 com $\psi_{j,m} = 0$ para todos os pares j, m.

$$\mu_n^\dagger(\{\theta_{j,m}, \psi_{j,m} = 0\}) = \prod_{m=0}^{n-1} [\oplus_{j_m=0}^{2^m-1} R_y(\theta_{j,m}) \otimes 1^{\otimes(n-m)}] \tag{4.49}$$

Vamos ver como computamos esses ângulos. Reescrevendo o estado $|\psi_{\vec{v}_+}\rangle$ como:

$$|\psi_{\vec{v}_+}\rangle \equiv \sum_{i=0}^{\frac{N}{2}-1} |i\rangle \otimes (|v_{2i}||0\rangle + |v_{2i+1}||1\rangle = \sum_{i=0}^{\frac{N}{2}-1} |i\rangle \otimes (|v_{i0}||0\rangle + |v_{i1}||1\rangle \tag{4.50}$$

Resumidamente, os índices $2i$ e $2i+1$ são substituidos por i_0 e $i_1 \forall i = 0, 1, \ldots, \frac{N}{2} - 1$. Vamos mostrar como os ângulos $\{\theta_{j,m}\}$ são calculados pelos coeficientes $|v_i|$.

Como primeiro passo, vamos encontrar o relacionamento entre $\{\theta_{j,m}\}$ e os coeficientes $|v_i|$. Pelo Teorema 3 em **??**, como os coeficientes $|v_i|$ possuem uma fase nula, um tem o operador

$$U_n = \oplus_{i=0}^{N-1} R_y(-\theta_i) \tag{4.51}$$

transforma o estado $|\psi_{\vec{v}}\rangle$ num estado separado. Definindo um estado

$$|p\rangle = \sum_{i=0}^{\frac{N}{2}-1} p_i |i\rangle \in H^{\otimes(n-1)} \tag{4.52}$$

tal que $U_n|\psi_{\vec{v}+}\rangle = |p\rangle|0\rangle$ então, como cada rotação $R_y(-\theta_i)$ é definida no subespaço $H_i = \{|i\rangle \otimes |0\rangle, |i\rangle|1\rangle$, obtemos $\frac{N}{2}$ equações para os ângulos $\{\theta_{i,0}\}$

$$\overbrace{\begin{pmatrix} \cos\frac{\theta_i}{2} & \sin\frac{\theta_i}{2} \\ -\sin\frac{\theta_i}{2} & \cos\frac{\theta_i}{2} \end{pmatrix}}^{R_y(-\theta_i)} \begin{pmatrix} |v_{i_0}| \\ |v_{i_1}| \end{pmatrix} = \rho_i \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \forall i$$

O sistema de equações correspondente é:

$$\begin{aligned} |v_{i0}| \cos\frac{\theta_i}{2} + |v_{i1}| \sin\frac{\theta_i}{2} &= p_i \\ -|v_{i0}| \sin\frac{\theta_i}{2} + |v_{i1}| \cos\frac{\theta_i}{2} &= 0 \end{aligned}$$

Usando a segunda equação do sistema é possível obter os ângulos θ_i no caso de $v_{i0} \neq 0$.

$$\begin{aligned} -|v_{i0}| \sin\frac{\theta_i}{2} + |v_{i1}| \cos\frac{\theta_i}{2} &= 0 \\ \Rightarrow \frac{|v_{i1}|}{|v_{i0}|} &= \tan\frac{\theta_i}{2} \\ \Rightarrow \theta_i &= 2\arctan\frac{|v_{i1}|}{|v_{i0}|} \end{aligned}$$

Se $v_{i0} = 0$ podemos facilmente tomar $\theta_i = \pi$ porque desta forma a matriz R_y permite transformar o estado $|1\rangle$ em $|0\rangle$.

Uma vez obtidos os ângulos como acima, temos que calcular explicitamente p_i para iterar o algoritmo para separar o próximo qubit no registrador. Para fazer isso, a primeira equação do sistema acima é usada:

$$p_i = (|v_{i0}|^2 \cos^2\frac{\theta_i}{2} + |v_{i1}|^2 \sin^2\frac{\theta_i}{2})^{\frac{1}{2}} \tag{4.53}$$

Portanto, os valores p_i são definidos positivamente.

Agora precisamos de um operador U_{n-1} para separar o qubit menos significativo no estado de $(n-1)$-qubits $|p\rangle$. Para encontrar os ângulos apropriados para construir o operador U_{n-1} repetimos o cálculo da equação matricial acima com o coeficiente do estado $|p\rangle$ tomado dois a dois como o coeficiente v_{i0} e v_{i1}.

$$|p\rangle = \sum_{i=0}^{\frac{N}{4}-1} |p\rangle \otimes (p_{i0}|0\rangle + p_{i1}|1\rangle) \tag{4.54}$$

Definindo um estado $|\alpha\rangle \in H^{\otimes(n-2)}$, tal que $U_{n-1}(\{\theta_{i,1}\})|p\rangle = |\alpha\rangle|0\rangle$, é possível encontrar a relação entre $\theta_{i,1}$ e os coeficientes p_{i0} e p_{i1} e itere o cálculo para encontrar todos os ângulos $\{\theta_{i,m}\}\forall i, m$.

Na construção da transformação que codifica os valores reais das entradas e pesos nos estados iniciais do circuito $|\Psi_v\rangle$ e $|\Psi_{w,b}\rangle$, precisamos primeiro calcular quantos serão os ângulos necessários para a codificação e quais seus valores. No caso da inicialização de um estado com 3 qubits, 7 ângulos vão ser necessários. De fato, um estado quântico arbitrário de 3 qubits é uma superposição de 2^3 autoestados, então 8 é o número das amplitudes, mas como o estado deve respeitar a restrição de unitaridade, precisamos apenas de 7 parâmetros independentes para defini-la. Vamos considerar o estado arbitrário de 3 qubits

$$|\psi_{\vec{v}}\rangle = \sum_{i=0}^{3} |i\rangle \otimes (v_{i0}|0\rangle + v_{i1}|1\rangle) \tag{4.55}$$

$$\vec{v} = (v_{00}, v_{01}, v_{10}, v_{11}, v_{20}, v_{21}, v_{30}, v_{31}) \in \Re^8$$

com $\vec{v}$ sendo um vetor real, a transformação $\mu(\vec{v})$, tal que $\mu(\vec{v})|0\rangle^{\otimes 3} = |\psi_{\vec{v}}\rangle$ é

$$\mu(\vec{v}) = Ph_3^\dagger(\vec{v})\mu_3^\dagger(\{\{\theta_{j,m}\}, \{\varphi_{j.m} = 0\}\}) = Ph_3^\dagger(\vec{v}) \prod_{m=0}^{3} \mu_{3,m}^\dagger(\{\theta_{j,m}\}, \{0\}) \tag{4.56}$$

por fim a equação destrinchada fica igual a

$$\mu(\vec{v}) = Ph_3^\dagger(\vec{v})\mu_3^\dagger(\{\theta_{j,0}\})\mu_2^\dagger(\{\theta_{j,1}\})\mu_1^\dagger(\{\theta_{j,2}\}) \tag{4.57}$$

Como temos 4 pares (v_{i0}, v_{i1}) o operador U_3 depende de 4 ângulos, U_2 depende de 2

ângulos e U_1 de um único ângulo. Para simplificar a notação, vamos reescrever o conjunto de ângulos $\{\theta_{j,0}\}$ como $\{\theta_0, \theta_1, \theta_2, \theta_3\}$, o conjunto $\{\theta_{j,1}\}$ como $\{\theta_4, \theta_5\}$ e $\{\theta_{j,2}\}$ como $\{\theta_6\}$.

4.6 Algoritmo desenvolvido

A implementação do algortimo para a inicialização dos estados segue:

Algorithm 1 Inicialização do estado quântico

```
n ← len(reg_q)
deltas_list ← lista de ângulos θ_i
init_circuit ← circuito quântico
init_reg ← registrador quântico(n)
for deltas in deltas_list do
    for j = 0 to len(deltas) do
        if len(deltas) == 1 then
            init_circuit ← R_y(d[j], init_reg[0])
        else if len(deltas) == 2 then
            init_circuit ← R_y(d[j]/2, init_reg[0])
            init_circuit ← CX(init_reg[1], init_reg[0])
        else
            init_circuit ← R_y(d[j]/4, init_reg[0])
            if j  mod 2 == 0 then
                init_circuit ← CX(init_reg[1], init_reg[0])
            else
                init_circuit ← CX(init_reg[1], init_reg[0])
                init_circuit ← CX(init_reg[2], init_reg[0])
            end if
        end if
    end for
end for
return circuit
```

Para desenvolver a transformação $Ph_n^\dagger(\vec{v})$ descrita na Equação 4.48, fizemos uma adaptação do algoritmo descrito pelos autores do artigo com a finalidade de simplificar sua implementação. Como mostrado, $Ph_n^\dagger(\vec{v})$ é uma matriz diagonal com termos 1 e -1. Os autores utilizaram um método baseado em hipergrafos quânticos [40], enquanto nosso desenvolvimento se baseia na transformação de uma matriz unitária gerada a partir dos vetores de enradas e pesos classicamente com a ajuda de um método da biblioteca qiskit, que cria uma porta lógica quântica a partir de uma matriz unitária qualquer. A implementação dos autores considera um estado de n qubits $|+\rangle^{\otimes n}$, onde $\sqrt{2}|+\rangle = |0\rangle + |1\rangle$, e o estado do hipergrafo quântico que corresponde ao hipergrafo $G \leq n = \{V, E\}$, onde

V é o conjunto de n vertices e E é o conjunto de hiper-arestar para qualquer ordem de k = 1 até k = n.

Algorithm 2 Implementação do operador $Ph_n^\dagger(\vec{v})$

```
v ← vetor que será a matriz diagonal
q_input ← qubits do registrador q
circuit ← circuito quântico
vector_mod ← lista vazia
Ph ← lista vazia
for i = 0 to len(v) do
    if v[i] == 0 then
        inserir 1 em vector_mod
    else
        inserir v[i]/|v[i]| em vector_mod
    end if
end for
Ph ← numpy.diag(vector_mod)
circuit ← circuit.unitary(Ph, q_input)
retrun circuit
```

No Algoritmo 2, criamos uma matriz diagonal com termos $\{-1, 1\}$ a partir dos valores nos vetores de entradas e pesos, de tal forma que preservamos somente os sinais desses valores.

O próximo passo é a implementação da transformação S_v descrita na Equação 4.9. Primeiro aplicando uma função que constroi uma porta Pauli X multi-controlada, makeCH, onde os qubits de controle são os de entrada e o target varia pela quantidade de qubits no registrador auxiliar. Em seguida, aplicamos portas PAuli X controlada entre os qubits auxiliares e os qubits de entrada como targets. Depois acrescentamos no circuito as transformações de inicialização dos estados quânticos. Por fim, aplicamos portas Pauli X nos qubits de entrada. Esta transformação, S_v, leva o circuito do estado $|0\rangle_a^{\otimes d}|0\rangle_q^{\otimes n}$ para o estado $|\Psi_z^d\rangle$.

Agora precisamos do cálculo das aproximações da série de Taylor das funções de ativação que queremos testar e dos ângulos que vão carregar essa informação para os estados quânticos. Chegando neste ponto tivemos alguns problemas no desenvolvimento do algoritmo por conta de alguns cálculos dúbios durante a análise do artigo. Primeiro vamos destrinchar a Equação 4.38. No cálculo da variável A_{dk} existe uma inconsistencia do produtório onde os valores da iteração decrescem, por conta disso resolvi utilizar valores

Algorithm 3 Implementação da transformação S_v

```
circuit ← circuito quântico
q_input ← qubits do registrador q
q_aux ← qubits do registrador a
v_x ← vetor de entradas
v_w ← vetor de pesos
deltas_x ← ângulos para as entradas
deltas_w ← ângulos para os pesos
n ← len(q_input) − 1
m ← len(q_aux) − 1
u_input, u_weigth ← circuito depois da aplicação do algoritmo de inicialização nas entradas e pesos
cU_input = u_input.control(1)
cU_weigth = u_input.control(1)                    ▷ transformações controladas
for i = 0 to n + 1 do
    for j = 0 to m + 1 do
        makeCH(circuit, q_input, ch_aux, q_aux[j])        ▷ porta Hadammard controlada
        for k in q_input do
            circuit.cx(q_aux[j], k)
        end for
        circuit.append(cU_input, [q_aux[j],0,1,2])
        phx_gate, phw_gate = createPhGate(circuit, q_input, v_x, v_w)
        circuit.append(cU_weigth, [q_aux[j],0,1,2])
        for k in q_input do
            circuit.x(k)
        end for
    end for
end for
return circuit
```

fixos para d e k tal que, $k = d - 1$. A outra incosntância, encontrei na Equanção 4.38, que depende da variável A_{dk}. Para iniciarmos a recursão que define o ângulo θ_i da Equação 4.38 precisamos do case baso para θ_0 que não é explicitada no decorrer do artigo. Então eu adotei um range de valores de $\pi/8$ até 4π para avaliar os resultados e definir a melhor métrica. O algoritmo desenvolvido para essa parte foi o seguinte:

Algorithm 4 Implementação da transformação S_u

```
q_aux ← qubits do registrador a
angles ← ângulos para rotação dos qubits
n ← qunatidade de qubits do registrados a
qc ← circuito quântico temporário
for i = 1 to n do
    for j = 0 to len(angles) do
        if j == k − 1 then
            qc.cry((−2 ∗ angles[j]), q_aux[k], q_aux[0])
            qc.cx(q_aux[k], q_aux[0])
        end if
    end for
end for
qc ← qc.to_gate()
qc ← qc.control(1)
return qc
```

Juntando todos esse algoritmos anteriores, a criação do circuito completo é feita por uma função, create_circuit, que recebe como parâmetros o vetor de entradas, vetor de pesos, viés da função, função analítica e a ordem para aproximação. Dois vetores são definidos para guardar os ângulos de rotação necessárioa para realizar a inicialização dos estados. Definimos os valores dos vetores que serão passados como amplitude do estado quântico inicial. Criamos o circuito com todos os registradores que farão a computação da aproximação. Agora que temos os ângulos necessários e o circuito criado começamos a computação fazendo uma porta controlada x entre os qubits de entrada e o qubit do registrador 'l', controlado por 'l'. Aplicamos a transformação S_v ao circuito, que inclui, a incialização dos estados e o preparo do circuito, em seguida, aplicamos a transformação S_u com os ângulos de rotação calculádos a partir da expanção da Série de Taylor da função analítica escolhida, mais uma porta x controlada entres os qubits de entradas e 'l', dessa vez controlada pelos qubits de entrada. Por fim aplicamos uma medição em todos os qubits que particparam da computação para extrair a probabílidade de todos os qubits serem $|0\rangle$.

Algorithm 5 Criação do circuito quântico completo

input_vector, weight_vector, bias← vetores de entradas, pesos e bias da função
function ← função de ativação que queremos aproximar
order ← ordem da série de Taylor
delt_x, delt_w ← listas vazias para ângulos da inicialização dos estados
v_x, v_w = create_vectors(input_vector, weight_vector, bias)
n_in = int(math.log(len(v_x)+1, 2))
n_aux = order
$thetas\_x = split\_angles(generate\_thetas\_initial(v\_x, len(input\_vector))$
$thetas\_w = split\_angles(generate\_thetas\_initial(v\_w, len(weight\_vector))$
for $i = 0$ to $len(thetas\_x$ **do**
    $deltas\_x = generate\_deltas(thetas\_x[i], delt\_x)$
    $deltas\_w = generate\_deltas(thetas\_w[i], delt\_w)$
end for
$deltas\_x = split\_angles(deltas\_x, len(input\_vectgor))$
$deltas\_x = deltas_x[:: -1]$
$deltas\_w = split\_angles(deltas\_w, len(weight\_vectgor))$
$circuit = QuantumCricuit(name =' init')$
$number\_classical\_output = n\_in + n\_aux + 1$
$q\_input = QuantumRegister(n\_in,' q')$
$q\_aux = QuantumRegister(n\_aux,' a')$
$aux = QuantumRegister(n\_in + 1,' ch')$
$l = QuantumRegister(1,' l')$
$c\_output\_input = ClassicalRegister(n\_in,' c_input')$
$c\_output\_aux = ClassicalRegister(n\_aux,' c\_aux')$
$c\_output\_l = ClassicalRegister(1,' c\_l')$
$circuit.add\_register(register)$
$circuit.h(l)$
for i in q_input **do**
    circuit.cx(i, l)
end for
create_v_gate(circuit, q_input, q_aux, v_x, v_w, deltas_x, deltas_w)
maclaurin_angles = calculate_thetas_maclaurin(function, order)
circuit.append(u_gate(q_aux, maclaurin_amgções), [l[0], q_aux[:]])
for i in q_input **do**
    circuit.cx(l, i)
end for
circuit.h(l)
circuit.measure(q_input, c_output_input)
circuit.measure(q_aux, c_output_aux)
circuit.measure(l, c_output_l)
return circuit

4.7 Computação da Amplitude do Estado

Para resumir, o circuito quântico definido até agora emprega n + d qubits e executa duas transformações: a primeira envia o estado $|0\rangle_a^{\otimes d}|0\rangle_q^{\otimes n}$ para $|\Psi_z^d\rangle$, como consequência do Teorema 1, enquanto a segunda é $S_u \otimes X^{\otimes n}$ que devolve um estado tendo $2^{-d/2}f_d(z)$ como probabilidade da amplitude correspondente ao estado $|0\rangle_a|0\rangle_q$. Uma propriedade importante desse circuito quântico, que é mostrada na Figura 4**-a** é que ele codifica o valor $f_d(z)$, que é não linear em relação aos valores de entrada $\vec{x}$, em um estado quântico $|\Psi_{f(z)}^d\rangle$. De fato, em um contexto genérico, o circuito quântico na Figura 4**-a** pode ser integrado em um circuito para uma rede neural multicamada baseada em qubit. Nesse contexto, o valor $f_d(z)$ corresponde ao valor de saída de um neurônio oculto. A liberdade deixada pela função de ativação não ser destruida torna possível construir uma rede neural profunda baseada em qubit. Cada nova camada recebe estados quânticos, como o estado preparado para entrar na rede na primeira camada. Como último resultado, os autores mostram explicitamente como operar a última camada da rede, focando no caso de um perceptron de uma camada.

Enquanto o circuito da Figura 4**-a** retorna um estado que tem o valor $2^{-d/2}f_d(z)$ codificado como uma amplitude de probabilidade, o circuito da Figura 4**-b** permite estimar tal amplitude. Ele implementa uma versão baseada em qubit de um perceptron de uma camada. Qualquer algoritmo quântico termina extraindo informações do estado quântico dos qubits por meio de operações de medição. Aplicando as operações de medição aos qubits do circuito da Figura 3-a permite estimar apenas a probabilidade de medir um determinado estado quântico, mas a partir da probabilidade não é possível calcular a amplitude inerente. De fato, a probabilidade é o módulo quadrado da amplitude, portanto, não preserva a informação sobre o fator de fase (o sinal para valores reais) da amplitude. Para atingir tal objetivo um método simples consiste em definir um circuito quântico que retorna um estado quântico com um valor $\frac{1}{2}(1 + 2^{-d/2}f_d(z))$ armazenado como amplitude de probabilidade, tarefa operada pelo circuito da Figura 4**-b**.

Tal circuito quântico opera em um registrador l de um único qubit, além dos registradores q e a, e retorna a probabilidade $\frac{1}{4}|1+2^{-d/2}f_d(z)|^2$ para observar o estado $|0\rangle_l|0\rangle_a|0\rangle_q$. Vamos agora nos concentrar em tal circuito dedicado à estimativa da saída do perceptron. Vamos considerar um estado de n qubits $|\Psi\rangle$ com amplitudes reais e um operador unitário U tal que $U|0\rangle^{\otimes n} = |\Psi\rangle$. Para estimar a amplitude $\langle 0|\Psi\rangle$, onde $|0\rangle \equiv |0\rangle^{\otimes n}$, um algoritmo

de três etapas pode ser definido para atingir o objetivo. O algoritmo prevê o uso de $(n+1)$ qubits, n dos quais armazenar $|\Psi\rangle$ rotulado com q e um qubit adicional l. Os referidos três passos consistem numa porta Hadamard em l, a transformação U aplicada aos qubits do registardor q e controlada por l outra porta Hadamard em l De fato, a partir do estado, após o primeiro portão Hadamard, o estado $(n+1)$ qubits se torna $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle^{\otimes n}$.

Com a transformação controlada-U o estado se torna $\frac{1}{\sqrt{2}}(|0\rangle|1\rangle^{\otimes n} + |1\rangle|\Psi\rangle)$ e, com a última porta Hadamard, $\frac{1}{2}[|0\rangle(|0\rangle^{\otimes n}+|\Psi\rangle)|1\rangle(|0\rangle^{\otimes n}-|\Psi\rangle)]$. Após uma medição de $n+1$ qubits, a probabilidade de medir o estado $|0\rangle_l|0\rangle)q^{\otimes n}$ é $P_0 = \frac{1}{4}|1+\langle 0|\Psi\rangle|^2$. Após a estimativa de P_0 a amplitude $\langle 0|\Psi\rangle$ é alcançável invertendo a fórmula, portanto $\langle 0|\Psi\rangle = 2\sqrt{P_0} - 1$. O quadrado do módulo é invertível nesse caso porque $|\langle 0|\Psi\rangle \leq 1$. Vamos aplicar esse método de estimativa de amplitude no caso do perceptron quântico. O circuito quântico, exposto nas seções anteriores e mostrado na Figura 4-**a**, aplicado em dois registradores qubit q e a, cada um inicializado no estado $|0\rangle$, ele retorna um estado de $(n+d)$ qubits $|\Psi^d_{f(z)}\rangle$. O circuito é resumido como uma série de portas X aplicadas nos qubits no registrador q, a subrotina S_v aplicada nos dois registradores seguida por S_u aplicada no registrador a e outra série de portas X aplicadas nos qubits no registrador q. O circuito fica $X^{\otimes n}s_u S_v X^{\otimes n}|0\rangle^{\otimes d}|0\rangle^{\otimes n} = |\Psi^d_{f(z)}\rangle$. Portanto, para estimar ${}_q\langle -|_a\langle 0|\Psi^d_{f(z)}\rangle = 2^{-d/2}f_d(z)$, vamos aplicar o algoritmo de estimativa de amplitude descrito acima onde a transformação U é $X^{\otimes n}S_u S_v X^{\otimes n}$ e, por sua vez, o estado $|\Psi\rangle$ é o estado $|\Psi\rangle^d_{f_d(z)}$. Portanto, após uma medição em todos os qubits, a probabilidade de obter o estado $|0\rangle_l|0\rangle_a^{\otimes d}|0\rangle_q^{\otimes n}$ é

$$P_0 = \frac{1}{4}|1 + 2^{-d/2}f_d(z)|^2 \tag{4.58}$$

A partir da estimativa de P_0 é possível calcular uma estimativa de $2^{-d/2}f_d(z)$. Resumindo, o circuito, que permite estimar a amplitude $2^{-d/2}f_d(z)$, é $H_l C_l(X^{\otimes n}S_u S_v X^{\otimes n})H_l$ com uma medição para cada qubit nos registradores q, a e l respectivamente. Observe que tal circuito quântico é parcialmente diferente em relação ao circuito na Figura 4-**b**. A partir da Figura 4-**b**, a subrotina S_v não é controlada por l. De fato, a subrotina S_v é construída tal que $S_v|0\rangle^{\otimes d}|0\rangle^{\otimes n} = |0\rangle^{\otimes d}|0\rangle^{\otimes n}$. Portanto, o circuito $H_l C_l(X^{\otimes x}S_u S_v X^{\otimes n})H_l$ e $H_l C_l(X^{\otimes n}S_u)S_v C_l(X^{\otimes n})H_l$ permite alcançar o mesmo propósito de construir um estado com P_0 como probabilidade de obter o estado após uma medição para cada qubits.

Observamos que os teoremas 1 e 2 implicam que um estado quântico com $2^{\frac{-d}{2}}f_d(z) \leq 1$ como coeficiente de superposição existe e, uma vez que qualquer estado quântico $|\Psi\rangle$ é

normalizado então $2^{\frac{-d}{2}} f_d(z) \leq 1$. De tais resultados, definindo $P_0 = \frac{1}{4}|1 + 2^{\frac{-d}{2}} f_d(z)|^2$ é possível reverter a equação para encontrar $f_d(z)$, uma vez dada a probabilidade P_0.

Portanto, o algoritmo do perceptron quântico consiste em uma estimativa da probabilidade P_0 viável com um número de operações de medição de todos os qubits. O erro sobre a estimativa de P_0 depende do número de amostras. A saída resultante do perceptron baseado em qubit é escrita

$$y_q = 2^{\frac{d}{2}}(2\sqrt{P} - 1)C_d \tag{4.59}$$

onde P é a estimativa de P0 e Cd é definido na Equação 4.45. Portanto y_q fornece a estimativa do valor $f_d(z)$, que é a expansão polinomial da função de ativação f na ordem d. Uma vez que a estimativa de P_0 é obtida por uma computação quântica, o valor y_q é derivado de uma computação clássica. A estimativa de P_0 é dada por $P = m/S$ onde S é o número total das medições de $|0\rangle\langle 0|_l \otimes |0\rangle\langle 0|_a \otimes |0\rangle\langle 0|_q$ e m é o número dessas medidas que retornam 1 como resultado.

5 RESULTADOS

Os resultados foram obtidos da seguinte forma: foi criada uma função para extrair o cálculo da probabilidade P_0, definida na Equação 4.58, e o valor de saída y_q definido na Equação 4.59. Em seguida, iteramos sobre a função definida a partir do mesmo conjunto de entradas e pesos alterando somente o primeiro valor do vetor de entradas, para gerar o cálculo do produto interno dentro do intervalo [-10,10]. Adotei o intervalo [-10, 10] porquê foi mais fácil de visualizar as nuances dos gráficos.

Os resultados obtidos a partir do circuito contruído não foram satisfátorios. Por conta de algumas informações não explicitadas no artigo, tomei a liberdade para testar o circuito com ângulos diferentes, que serviram como base para o cálculo da Equação 4.38. O range de ângulos escolhido foi de $-\pi/2$ até $\pi/2$, a escolha desses ângulos foi aleatória justamente por conta da falta de coerência no cálculo desse ângulo inicial descrita pelos autores e o range foi feito a partir do ângulo definido na Equação 4.21 do Teorema 2.

As funções de ativação usadas para os testes foram a sigmóide e a seno por serem mais popularmente usadas como funções de ativação. Para avaliar métricamente o valor da saída, pagmos a distância Euclídiana entre todos os pontos da função observada e todos os valores de sáida do neurônio para calcular a média dessas distâncias. Avaliamos essa média em relação à distancia ídeal de aproximação, 0.

Sigmóide

Para a função sigmóide, a configuração de teste que trouxe melhor resultado foi com a ordem de aproximação igual a 5, ângulo inicial igual a $\pi/2$ e 100 interações para o conjunto de entradas [-1, 0, 0, 0] e pesos [1, 0, 0, 0] com bias igual a 0.01 e range de [-10,10]. Além disso, os autores do artigo não utilizam o produto interno z para avaliar o circuito, eles utilizam um múltiplo desse valor, $4z$.

Nos testes subsequentes, notei que alterar o ângulo inicial do cálculo da Equação 4.38 não faz muita diferença na saída do neurônio quântico. Os parâmetros que mais afetam o output da rede são os valores dos vetores de entradas $\vec{v}$ e pesos $\vec{w}$ junto com a ordem da aproximação d. Com isso, os gráficos a seguir mostam os resultados dos testes feitos, alterando somente a ordem de aproximção d, com $d = 3$, $d = 5$ e $d = 7$, respectivamente.

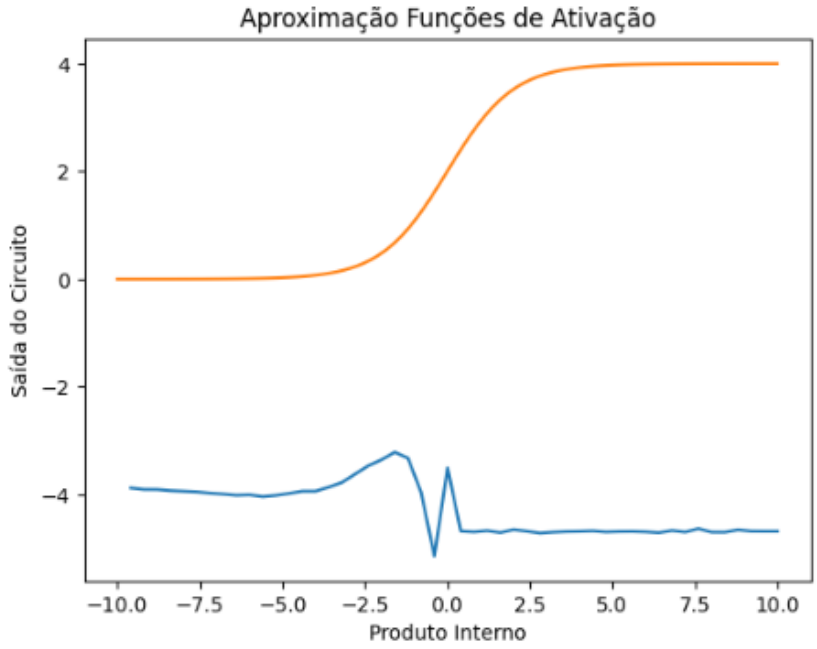


Figura 5: Relação entre a saída do circuito quântico ($\hat{y}$) com o valor do produto interno ($4z$) entre os vetores de entradas e pesos. d = 3. Média da distância Euclidiana igual a 4.67

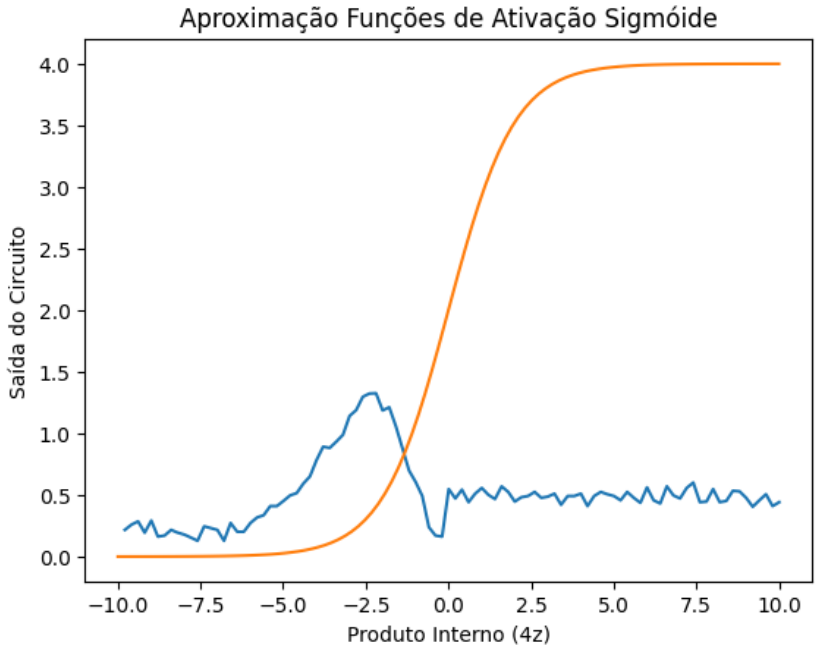


Figura 6: Relação entre a saída do circuito quântico ($\hat{y}$) com o valor do produto interno ($4z$) entre os vetores de entradas e pesos. d = 5. Média da distância Euclidiana igual a 10.60

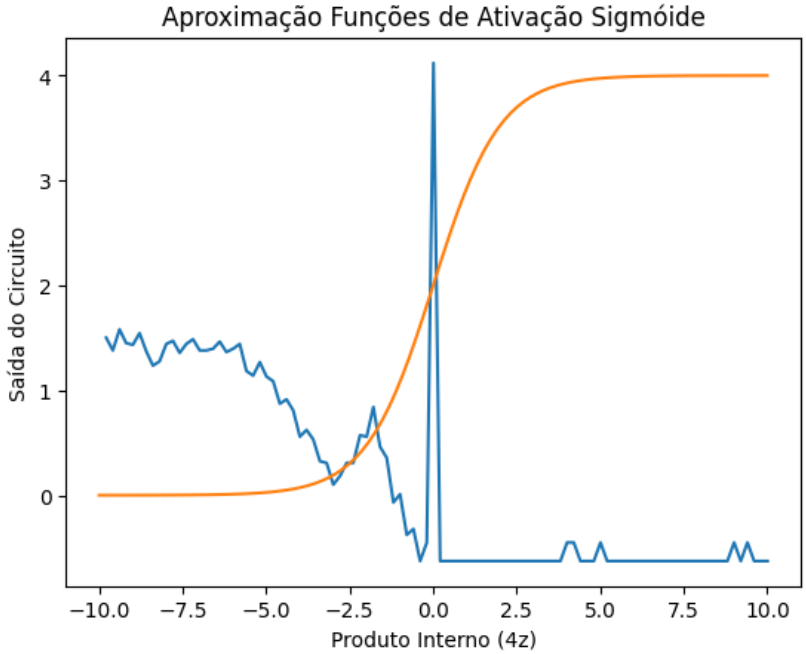


Figura 7: Relação entre a saída do circuito quântico ($\hat{y}$) com o valor do produto interno ($4z$) entre os vetores de entradas e pesos. d = 7. Média da distância Euclidiana igual a 22.55

Seno

Para a função seno, a configuração de teste que trouxe melhor resultado foi com a ordem de aproximação igual a 5, ângulo inicial igual a $-\pi/4$ e 100 interações para o conjunto de entradas [-1, 0, 0, 0] e pesos [1, 0, 0, 0] com bias igual a 0.01 e range de [-10,10]. O produto interno utilizado para testar a função seno foi $4z$.

Da mesma forma que a sigmóide, alterar o ângulo base para o cálculo da Equação 4.38 não muda drasticamente a saída do perceptron, o que faz ter uma divergência maior nos resultados são os valores de entradas e pesos.

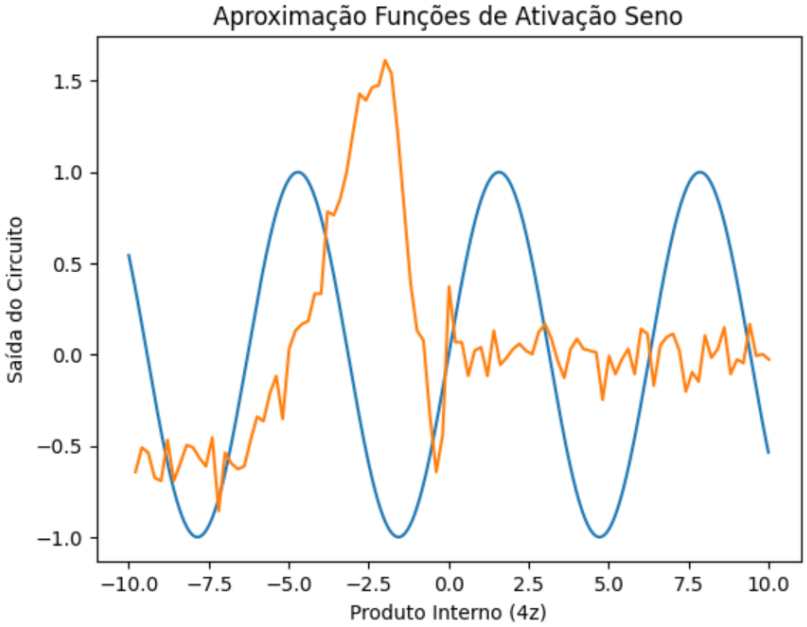


Figura 8: Relação entre a saída do circuito quântico ($\hat{y}$) com o valor do produto interno ($4z$) entre os vetores de entradas e pesos. d = 3. Média da distância Euclidiana igual a 9.34

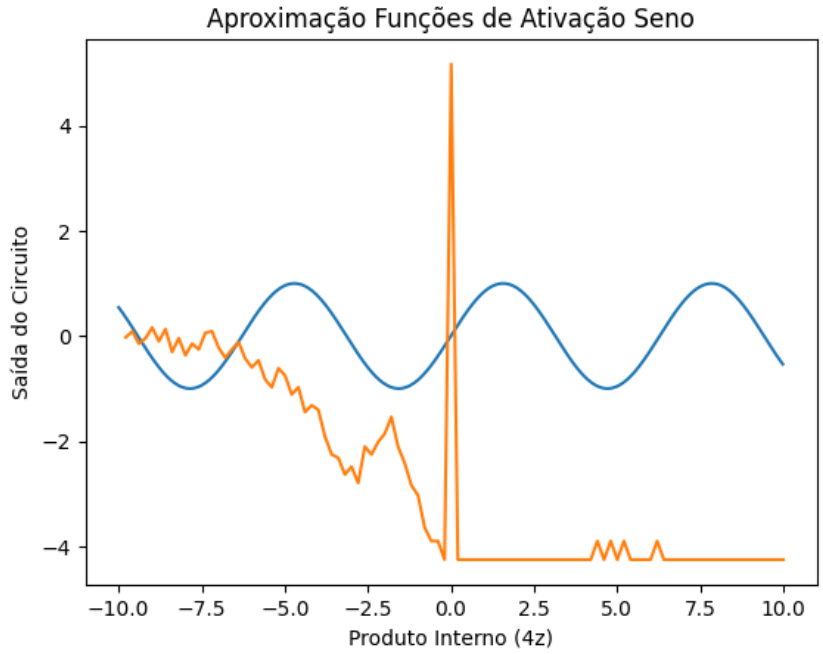


Figura 9: Relação entre a saída do circuito quântico ($\hat{y}$) com o valor do produto interno ($4z$) entre os vetores de entradas e pesos. d = 5. Média da distância Euclidiana igual a 21.19

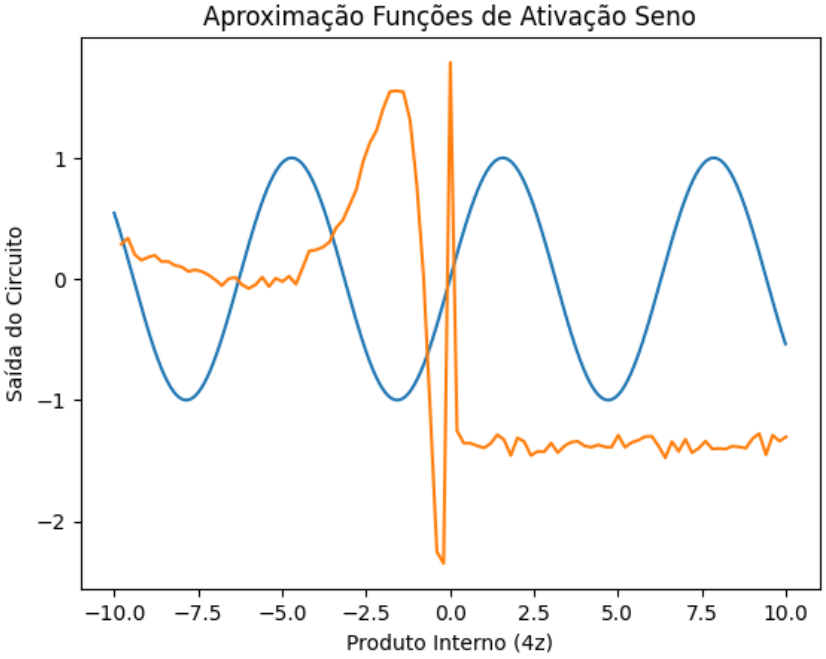


Figura 10: Relação entre a saída do circuito quântico ($\hat{y}$) com o valor do produto interno ($4z$) entre os vetores de entradas e pesos. d = 7. Média da distância Euclidiana igual a 45.06

5.1 Discussão

Podemos ver que a aproximação não é feita corretamente ao longo da função. Existem alguns fatores que podem ter interferido no resultado. Um deles é a definição fornecida do operador Ph, que no artigo definem como um estado contruído a partir de hipergrafos [40] mas não consegui seguir a mesma implementação e utilizei uma correção clássica para tentar simular o mesmo comportamento. Outro fator já explicitado antes é a falta de coerência no cálculo do ângulo definido na Equação 4.38, que mesmo os testes mostrando que alterar esse ângulo não faz tanta diferença no circuito proposto, essa falta de interferência pode ser justamente por conta dos cálculos inconclusivos.

Na função sigmóide, apesar da saída do neurônio divergir da função real, podemos notar que o resultado tem algumas similaridades interessantes, como a diferenciação da curva em torno do ponto 0 do produto interno, que indica uma mudança brusca no estado da rede e a constância da curva ao se distanciar do ponto 0, similarmente como o que acontece na função sigmoid real.

Na função seno, percebi grande variação nos resultados e novamente um comportamento não esperado quando o produto interno dos vetores chega próximo do 0. Em todos os testes realizados o valor de saída do circuito estabiliza com o valor do produto interno $z > 0$.

Esses comportamentos também podem surgir por conta da profundidade do circuito quântico desenvolvido. Como existe uma recursão na transformação S_v, a profundidade do circuito fica muito grando com o crescimento do valor da ordem de aproximação d, e, como estamos trabalhando com um simulador quântico ruidoso, essa profundidade afeta diretamente o erro correlacionado do circuito.

Ao comparar com os resultados do artigo, posso dizer que o circuito desenvolvido não é satisfatório pois a aproximação não consegue ser feita, mas os resultados mostram também que é gerada uma função a partir do circuito e que essa informação pode ser propagada. Acredito que alguns ajustes no algoritmo desenvolvido podem vir a acarretar em resultados melhores.

6 CONCLUSÃO

Em resumo, os resultados desta pesquisa não foram satisfatórios em aproximar as funções analíticas, seno e sigmóide, dentro de um processador quântico a partir dos termos das Series de Taylor dessas funções. Apesar de termos obtido alguns insights interessantes, o estudo apresentou limitações que impediram a obtenção de resultados mais precisos. Discutimos algumas possíveis razões para essas limitações, como a inconsistencia de alguns cálculos feitos e os desafios metodológicos encontrados durante a revisão bibliográfica. É importante reconhecer que, embora os resultados não tenham sido positivos, ainda há muito a aprender com esse estudo sobre as áreas de computação quântica e aprendizado de máquina quântico.

6.1 Trabalhos Futuros

Para o futuro, podemos melhorar os resultados do algoritmo e aplicar o modelo desenvolvido na resolução de problemas complexos na área de computação quântica pois é uma área de grande potêncial de aplicações como química quântica, criptografia e inteligência artificial

REFERÊNCIAS

[1] ROSENBLATT, F. The perceptron, a perceiving and recognizing automaton project. *Cornell Aeronautical Laboratory*, 1957.

[2] SUTER, B. W. The multilayer perceptron as an approximation to a bayes optimal discriminant function. *IEEE Transactions on Neural Networks 1, 291*, 1990.

[3] HORNIK, K. Approximation capabilities of multilayer feed-forward networks. *Neural networks 4, 251±257*, 1991.

[4] PRESKILL, J. Quantum computing in the nisq era and beyond. *Quantum 2, 79*, 2018.

[5] AARONSON, S. Read the fine print. *Nature Physics 11, 291±293*, 2015.

[6] PRATI, E. Quantum neuromorphic hardware for quantum artificial intelligence. *Journal of Physics: Conference Series, vol. 880.*, 2017.

[7] BIAMONTE, J. e. a. Quantum machine learning. *Nature 549, 195*, 2017.

[8] ROCUTTO L., D. C. . P. E. Quantum semantic learning by reverse annealing of an adiabatic quantum computer. *Advanced Quantum Technologies 2000133*, 2020.

[9] FARHI E. NEVEN, H. Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002*, 2018.

[10] BENEDETTI M., L. E. S. S. . F. M. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology 4, 043001*, 2019.

[11] BROUGHTON, M. e. a. Tensorflow quantum: A software framework for quantum machine learning. *arXiv preprint arXiv:2003.02989*, 2020.

[12] MCCLEAN J. R., B. S. S. V. N. B. R. . N. H. Barren plateaus in quantum neural network training landscapes. *Nature communications 9, 1±6*, 2018.

[13] GRANT E., W. L. O. M. . B. M. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum 3, 214*, 2018.

[14] CEREZO M., S. A. V. T. C. L. . C. P. J. Cost-function dependent barren plateaus in shallow quantum neural networks. *Nature Communications 12, 1791*, 2021.

[15] HORNIK K., S. M. W. H. e. a. Multilayer feed-forward networks are universal approximators. *Neural networks 2, 359±366*, 1989.

[16] DASKIN, A. A simple quantum neural net with a periodic activation function. *2018 IEEE International 26 Conference on Systems, Man, and Cybernetics (SMC), 2887±2891*, 2018.

[17] TORRONTEGUI E. GARCíA-RIPOLL, J. J. Unitary quantum perceptron as efficient universal approximator. *EPL (Europhysics Letters) 125, 30004*, 2019.

[18] CAO Y., G. G. G. . A.-G. A. Quantum neuron: an elementary building block for machine learning on quantum computers. *arXiv preprint arXiv:1711.11240*, 2017.

[19] HU, W. Towards a real quantum neuron. *Natural Science 10, 99±109*, 2018.

[20] SILVA A. J., d. O. W. R. . L. T. B. da. Weightless neural network parameters and architecture selection in a quantum computer. *Neurocomputing 183, 13± 22*, 2016.

[21] MATSUI N., N. H. . I. T. Qubit neural network: Its performance and applications. *Complex-Valued Neural Networks: Utilizing High-Dimensional Parameters, 325±351*, 2009.

[22] SILVA A. J., L. T. B. . d. O. W. R. da. Quantum perceptron over a field and neural network architecture selection in a quantum computer. *Neural Networks 76, 55±64*, 2016.

[23] VENTURA D. MARTINEZ, T. Quantum associative memory. *Information Sciences 124, 273±296*, 2000.

[24] SILVA A. J. DE OLIVEIRA, R. L. F. da. Neural networks architecture evaluation in a quantum computer. *In 2017 Brazilian Conference on Intelligent Systems (BRACIS), 163±168*, 2017.

[25] SCHULD M., S. I. . P. F. The quest for a quantum neural network. *Quantum Information Processing 13, 2567±2586*, 2014.

[26] SHAO, C. A quantum model for multilayer perceptron. *arXiv preprint arXiv:1808.10561*, 2018.

[27] TACCHINO F., M. C. G. D. . B. D. An artificial neuron implemented on an actual quantum processor. *npj Quantum Information 5, 26*, 2019.

[28] KAMRUZZAMAN A., A. Y. L. A. . T. C. C. Quantum deep learning neural networks. *In Future of Information and Communication Conference, 299±311*, 2019.

[29] TACCHINO, F. e. a. Quantum implementation of an artificial feed-forward neural network. *Quantum Science and Technology*, 2020.

[30] MARONESE M. PRATI, E. A continuous rosenblatt quantum perceptron. *International Journal of Quantum Information 19, 2140002*, 2021.

[31] SHENDE V. V., B. S. S. . M. I. L. Synthesis of quantum-logic circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 25, 1000±1010*, 2006.

[32] KUZMIN V. V. SILVI, P. Variational quantum state preparation via quantum data buses. *Quantum 4, 290*, 2020.

[33] LAZZARIN M., G. D. E. . P. E. Multi-class quantum classifiers with tensor network circuits for quantum phase recognition. *arXiv preprint arXiv:2110.08386*, 2021.

[34] ROMERO J., O. J. P. . A.-G. A. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology 2, 045001*, 2017.

[35] RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. [S.l.]: Alan Apt, 1995.

[36] AGGARWAL, C. C. *Neural Networks and Deep Learning*. [S.l.]: Springer, 2018.

[37] BERNHARDT, C. *Quantum Computing for Everyone*. [S.l.]: The MIT press, 2019.

[38] HONGBAO MARGARET YOUNG, Y. Y. M. Quantum entanglement introduction. *Academia Arena 2016;8(7)*, 2016.

[39] WIEBE, N.; KLIUCHNIKOV., V. Floating point representations in quantum circuitsynthesis. *New Journal of Physics, 15:1–17*, 2013.

[40] ROSSI M., H. M. B. D. . M. C. Quantum hypergraph states. *New Journal of Physics 15, 113022*, 2013.

[41] E. DESTRI C., M. M. P. Quantum activation functions for quantum neural networks. *arXiv preprint arXiv:2201.03700*, 2022.