Um Framework para Treinamento e Análise de Redes Neurais Quânticas Aplicadas a Portas Lógicas

Pedro C. Delbem
pedrodelbem@usp.br

Orientador(a): Prof(a). Dr(a). Nome do Orientador

Universidade de São Paulo (USP)

23 de Agosto de 2025

Objetivos

O principal objetivo deste trabalho foi desenvolver um framework computacional robusto em Python para a implementação, treinamento e análise sistemática de Redes Neurais Quânticas (QNNs). O estudo focou em arquiteturas baseadas em um único neurônio quântico, investigando sua capacidade de resolver tarefas de classificação representadas por portas lógicas clássicas, incluindo o problema não linearmente separável XOR, que classicamente exige múltiplas camadas [**?**, **?**]. Para isso, o framework foi projetado para comparar diferentes estratégias de codificação de dados de entrada — especificamente, codificação de fase e de amplitude simplificada — e avaliar a eficácia de múltiplos algoritmos de otimização na busca pelos parâmetros ideais do circuito quântico.

Métodos e Procedimentos

A metodologia deste projeto abrange a construção de um pipeline experimental completo, desde a definição do circuito quântico até a análise dos resultados do treinamento.

Arquitetura do Neurônio Quântico. Inspirado no perceptron clássico, o modelo de neurônio quântico utiliza um único qubit de processamento e um qubit auxiliar para a saída. A computação é realizada através da aplicação de um conjunto de portas quânticas parametrizadas. Os dados de entrada clássicos (x_1, x_2) são codificados nos ângulos de rotação dessas portas. A medição final do qubit de saída na base computacional Z fornece a classificação. Essa abordagem explora a superposição e o entrelaçamento para realizar cálculos que superam as limitações de um perceptron clássico de camada única. O framework foi construído sobre a biblioteca Qiskit [**?**], permitindo tanto a simulação em computadores clássicos quanto a execução em hardware quântico real.

Estratégias de Codificação de Dados. Duas abordagens para codificar os dados de entrada no estado do qubit foram implementadas e analisadas, conforme os módulos `neuron_utils.py` e `amplitude_encoding_utils.py`:

- **Codificação de Fase ('phase'):** Os valores de entrada binários são mapeados para $\{+1, -1\}$ e utilizados para modular os ângulos de rotação de portas quânticas (e.g., $R_Z(\theta_1 \cdot x_1)$, $R_X(\theta_2 \cdot x_2)$). Nessa arquitetura, a informação é armazenada na fase relativa do estado do qubit.
- **Codificação de Amplitude Simplificada ('amplitude_simplified'):** Nesta abordagem, o estado inicial dos qubits de entrada é preparado para corresponder diretamente ao vetor de entrada (ex: $|01\rangle$ para a entrada $(0, 1)$). Em seguida, portas de rotação com ângulos treináveis são aplicadas. A classificação é realizada por uma porta de controle múltiplo (MCX) que atua sobre um qubit de saída. Conforme implementado em `trainer_utils.py`, cada vetor de entrada é avaliado em uma execução de circuito separada.

Framework de Treinamento e Otimização. O processo de treinamento, orquestrado pela classe `trainer_qNN`, visa encontrar os parâmetros (ângulos de rotação) que minimizam uma função de custo. A função de custo é definida como o erro médio entre a probabilidade de

medição do estado $|1\rangle$ e a saída esperada pela porta lógica. O arquivo `config.json` especifica os parâmetros dos experimentos, como as portas lógicas a serem aprendidas (XOR, AND, OR, etc.), as codificações e os métodos de treino. Para os resultados apresentados, foi utilizado o método de busca exaustiva em grade (`cg-exhaustive_search`) para garantir uma exploração completa do espaço de parâmetros.

Resultados

Os experimentos foram executados conforme definido no `training.ipynb`, que gerencia a execução paralela das simulações para cada combinação de porta lógica, codificação e método de treinamento. Os dados resultantes foram consolidados e analisados com o script `visualize.py`.

A Figura **??** apresenta um heatmap que resume o desempenho das duas codificações, exibindo a média de iterações necessárias para atingir a convergência (erro abaixo da tolerância definida). A análise do heatmap revela que a codificação de fase foi, em geral, mais eficiente para problemas não lineares como XOR e XNOR. Em contraste, a codificação de amplitude simplificada apresentou convergência ligeiramente mais rápida para portas linearmente separáveis como AND e OR.

As curvas de convergência, que detalham a evolução do erro ao longo das iterações, corroboram esses achados. Observou-se que, embora ambos os métodos fossem capazes de aprender todas as funções lógicas, a trajetória de otimização e a robustez a mínimos locais variavam significativamente entre eles.

Conclusões

Este trabalho demonstrou com sucesso o desenvolvimento de um framework versátil para a criação e avaliação de Redes Neurais Quânticas. A plataforma permitiu uma comparação detalhada entre diferentes arquiteturas de codificação de dados, confirmando a capacidade de um único neurônio quântico de resolver problemas não linearmente separáveis, em linha com as conclusões de estudos de referência como o de Grossu (2021) [**?**].

Os resultados indicam que a escolha da estratégia de codificação é um fator crucial que impacta diretamente a eficiência do treinamento, com diferentes codificações sendo mais adequadas para tipos distintos de problemas. O framework estabelecido serve como uma base sólida para futuras investigações, incluindo a expansão para neurônios com múltiplos qubits, a exploração de codificações mais complexas e a avaliação do impacto do ruído em processadores quânticos reais.

Declaração de Conflito de Interesses: Os autores declaram não haver conflito de interesses.

Contribuição dos Autores: Pedro C. Delbem concebeu, planejou o estudo, implementou o código, realizou a coleta e análise dos dados, e participou da redação do manuscrito.

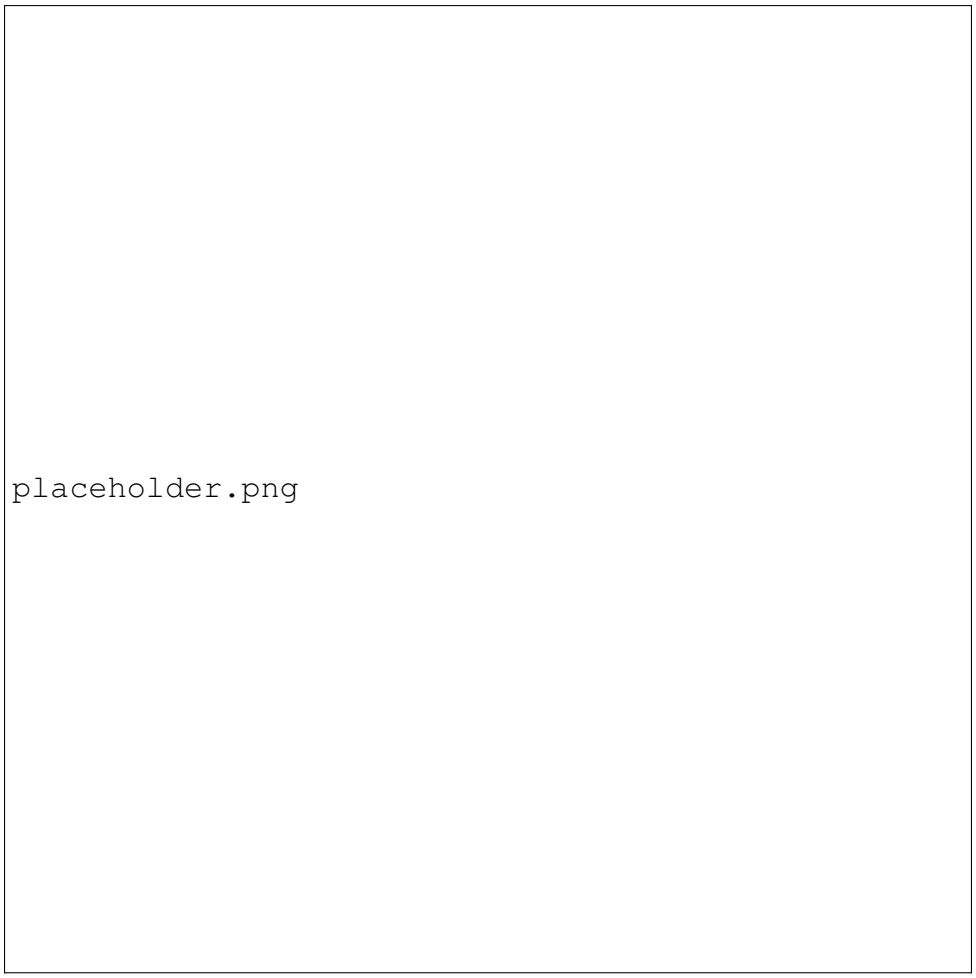

Figure 1: Heatmap comparando o número médio de iterações para convergência entre as codificações de Fase e Amplitude Simplificada para diferentes portas lógicas. Valores menores (azul escuro) indicam melhor desempenho. A imagem é uma representação ilustrativa dos resultados esperados.

References

[1] I.V. Grossu, "Single qubit neural quantum circuit for solving Exclusive-OR", *MethodsX*, vol. 8, p. 101573, 2021.

[2] A. Asfaw, L. Bello, Y. Ben-Haim et al., "Learn quantum computation using Qiskit", 2020. [Online]. Disponível em: `http://community.qiskit.org/textbook`.

[3] M. Minsky and S. Papert, "Perceptrons: An Introduction to Computational Geometry", MIT Press, Cambridge, 1969.