Differentiable learning of quantum circuit Born machines

Pöri & Sooki

Aalto University

December 10, 2024

Outline

1. Introduction
   - Motivation
   - High-level view of QCBM
2. Model and learning algorithm
   - Quantum circuit architecture design
   - Loss function and gradient-based optimization
3. Numerical experiments
   - Bars-and-stripes dataset
   - Mixture of Gaussians
4. Implementation

Introduction

Motivation

Goal

The goal of generative modeling is to model the probability distribution of observed data and generate new samples accordingly.

Applications

Computer vision, speech synthesis, chemical design and crucial component towards artificial general intelligence.

Challenges

Efficiently represent, learn, and sample from high-dimensional probability distributions

QCBM

Quantum Boltzmann machines

Generalize the energy function of classical Boltzmann machines to quantum Hamiltonian

Quantum Circuit Born Machines

1. Implicit generative model
2. Probability distribution using a quantum pure state
3. Probabilistic interpretation of quantum wave functions
4. Direct samples by measurement

Model and learning algorithm

Learning algorithm

Gradient-free optimization

1. Noisy hardware problems
2. Bad scalability

Gradient-based learning

1. Shows scalability in classical neural network
2. Application to QCBM training is nontrivial
   - QCBM $\rightarrow$ bit strings
   - Target function has to be an expectation value
   - Loss functions used in generative modeling fail to differentiate

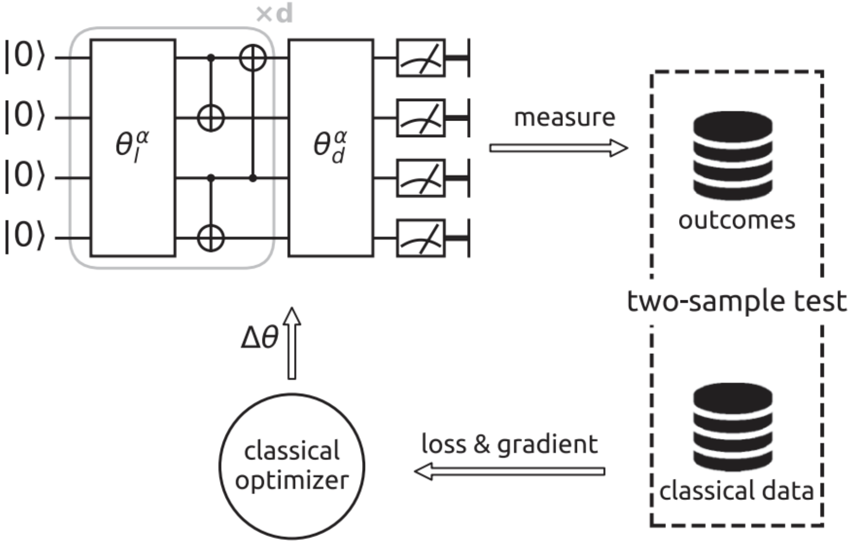
Model

Goal

1. Given $D = \{x\}$ containing independent and identically distributed (i.i.d.) samples from a target distribution $\pi(x)$.
2. QCBM generates samples close to the unknown target distribution.
3. The QCBM takes the product state $|0\rangle$ as an input and evolves it to a final-state $|\psi\rangle$ by a sequence of unitary gates.

Quantum circuit architecture

QCBM



×d
|0⟩
|0⟩
|0⟩
|0⟩
θ_l^α
θ_d^α
measure
outcomes
two-sample test
classical data
loss & gradient
$\Delta\theta$
classical optimizer

Loss function and gradient

Loss function

Goal

1. Kernel two sample test
2. Approaches zero if and only if the output distribution matches the target distribution exactly

Squared maximum mean discrepancy (MMD)

$$\mathcal{L} = \left\| \sum_{x} p_\theta(x)\phi(x) - \sum_{x} \pi(\mathbf{x})\phi(\mathbf{x}) \right\|^2 = \underset{x \sim p_\theta,\, y \sim p_\theta}{\mathbb{E}}[K(x,y)]$$

$$\text{- } 2 \underset{x \sim p_\theta,\, y \sim \pi}{\mathbb{E}}[K(x,y)] + \underset{x \sim \pi,\, y \sim \pi}{\mathbb{E}}[K(x,y)]$$

Gradient-based optimization

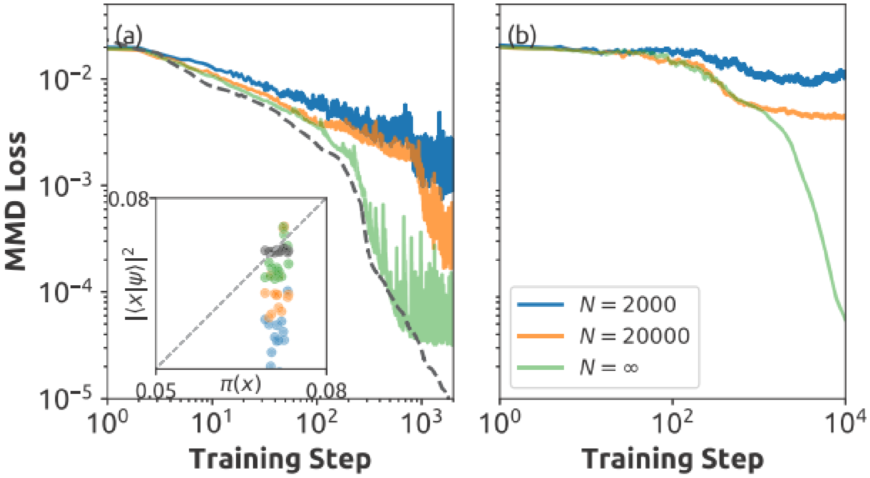
Derivative of the loss function

$$\frac{\partial \mathcal{L}}{\partial \theta_l^{\alpha}} = \underset{x \sim p_{\theta+}, y \sim p_{\theta}}{\mathbb{E}}[K(x,y)] - \underset{x \sim p_{\theta-}, y \sim p_{\theta}}{\mathbb{E}}[K(x,y)]$$

$$- \underset{x \sim p_{\theta+}, y \sim \pi}{\mathbb{E}}[K(x,y)] + \underset{x \sim p_{\theta-}, y \sim p_{\theta}}{\mathbb{E}}[K(x,y)]$$

Numerical experiments

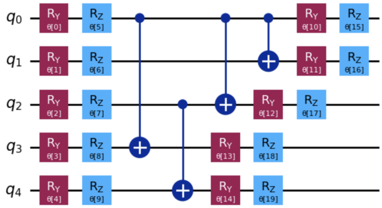
MMD loss as a function of training steps



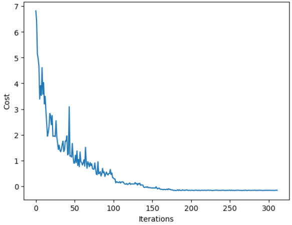
Samples from the QCBM trained under different measurement batch size N

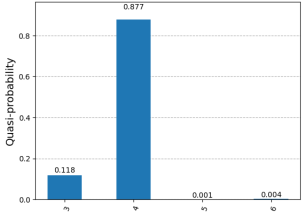
Implementation

Strongly entangling layer



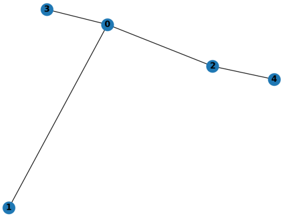
Cost over iterations





Sampling

Chow-Liu tree



https://github.com/0xSooki/QCBM

Any questions OR Clarification