Quantum Neural Network Simulation

Project Proposal

Alex Heilman

November 18, 2022

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

Classical Neural Networks have shown to be effective in a wide variety of uses.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

One proposed framework is that in Kerstin Beer, et al's *Training Deep Quantum Neural Networks*, which shall be considered here.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

$$\{|\phi_i^{in}\rangle, |\phi_i^{out}\rangle\}$$

Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

One proposed framework is that in Kerstin Beer, et al's *Training Deep Quantum Neural Networks*, which shall be considered here.

Overview

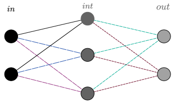
Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

One proposed framework is that in Kerstin Beer, et al's *Training Deep Quantum Neural Networks*, which shall be considered here.

$$\{|\phi_i^{in}\rangle, |\phi_i^{out}\rangle\}$$

$$\downarrow$$



Quantum Neural Network Simulation

Alex Heilman

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

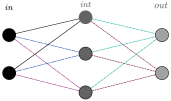
Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

One proposed framework is that in Kerstin Beer, et al's *Training Deep Quantum Neural Networks*, which shall be considered here.

$$\{|\phi_i^{in}\rangle, |\phi_i^{out}\rangle\}$$

$$\downarrow$$



$$\downarrow$$

$$\max_U \left(\sum_{i=1}^{N} \langle \psi_i^{out} | \rho_{out} | \psi_i^{out} \rangle \right)$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

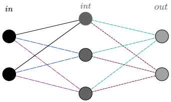
Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

One proposed framework is that in Kerstin Beer, et al's *Training Deep Quantum Neural Networks*, which shall be considered here.

$$\{|\phi_i^{in}\rangle, |\phi_i^{out}\rangle\}$$

$$\downarrow$$



$$\downarrow$$

$$\max_U \left(\sum_{i=1}^{N} \langle \psi_i^{out} | \rho_{out} | \psi_i^{out} \rangle \right)$$

$$\downarrow$$

$$U \rightarrow e^{i\epsilon K} U$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview

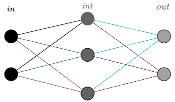
Classical Neural Networks have shown to be effective in a wide variety of uses.

With the advent of quantum computers, some modern research has been investigating quantum analogues of classical neural networks.

One proposed framework is that in Kerstin Beer, et al's *Training Deep Quantum Neural Networks*, which shall be considered here.

$$\{|\phi_i^{in}\rangle, |\phi_i^{out}\rangle\}$$

$$\downarrow$$



$$\downarrow$$

$$\max_U \left(\sum_{i=1}^{N} \langle \psi_i^{out} | \rho_{out} | \psi_i^{out} \rangle \right)$$

$$\downarrow$$

$$U \rightarrow e^{i\epsilon K} U$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Training Data Structure

Data will be provided for the training of the network via a set of arbitrary states (inputs), and the set of these same states after having some common unitary action act upon them (outputs). Hence, we will assume some given data set of the following form:

$$\text{Training Data: } \{(|\psi_i\rangle, V|\psi_i\rangle) \mid 1 \leq i \leq N\}$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Training Data Structure

Data will be provided for the training of the network via a set of arbitrary states (inputs), and the set of these same states after having some common unitary action act upon them (outputs). Hence, we will assume some given data set of the following form:

$$\text{Training Data:} \;\; \{(|\psi_i\rangle, V|\psi_i\rangle) \mid 1 \leq i \leq N\}$$

This is a reasonable set of data since the most general quantum network will apply an arbitrary unitary gate, and hence the most general circuit should be able to approximate such actions.

Architecture

The overall action of the network is composed of layer-by-layer composition of the transition map ϵ^{ℓ} for each layer ℓ s.t. $in \leq \ell \leq out$.

Architecture

The overall action of the network is composed of layer-by-layer composition of the transition map ϵ^{ℓ} for each layer ℓ s.t. $in \leq \ell \leq out$.

Each layer may have a different number of qubits M_{ℓ}. Explicitly, the ℓ-th layer's transition map takes the form:

$$\epsilon^{\ell}\left(\rho_{\ell-1}\right) =$$

$$\mathrm{Tr}_{\ell-1}\left[\left(\prod_{m=1}^{M_{\ell}} U_{\ell}^{m-M_{\ell}}\right)\left(\left(|0\rangle^{\otimes M_{\ell}}\langle 0|^{\otimes M_{\ell}}\right)_{\ell} \otimes \rho_{\ell-1}\right)\left(\prod_{m=1}^{M_{\ell}} U_{\ell}^{m\dagger}\right)\right]$$

$$= \rho_{\ell}$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Architecture

The overall action of the network is composed of layer-by-layer composition of the transition map ϵ^{ℓ} for each layer ℓ s.t. $in \leq \ell \leq out$.

Each layer may have a different number of qubits M_{ℓ}. Explicitly, the ℓ-th layer's transition map takes the form:

$$\epsilon^{\ell}\left(\rho_{\ell-1}\right) =$$

$$\mathrm{Tr}_{\ell-1}\left[\left(\prod_{m=1}^{M_{\ell}} U_{\ell}^{m-M_{\ell}}\right)\left(\left(|0\rangle^{\otimes M_{\ell}}\langle 0|^{\otimes M_{\ell}}\right)_{\ell} \otimes \rho_{\ell-1}\right)\left(\prod_{m=1}^{M_{\ell}} U_{\ell}^{m\dagger}\right)\right]$$

$$= \rho_{\ell}$$

And, hence, a total circuit of L layers returns ρ_{out}, defined below, for some given input state ρ_{in}.

$$\rho_{out} = \epsilon^{out}\left(\epsilon^{L}\left(\epsilon^{L-1}\left(\ldots\epsilon^{1}\left(\rho_{in}\right)\ldots\right)\right)\right)$$

Architecture: Step-by-step

For each layer ℓ,

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Architecture: Step-by-step

For each layer ℓ,

1. The next layer's M qubits are prepared in the initial state $|0\rangle^{\otimes m}\langle 0|_\ell^{\otimes m}$ and tensor producted with the previous layer's output $\rho_{\ell-1}$.

$$\rho'_\ell = \left(|0\rangle^{\otimes M}\langle 0|^{\otimes M}\right)_\ell \otimes \rho_{\ell-1}$$

Architecture: Step-by-step

For each layer ℓ,

1. The next layer's M qubits are prepared in the initial state $|0\rangle^{\otimes m}\langle 0|_{\ell}^{\otimes m}$ and tensor producted with the previous layer's output $\rho_{\ell-1}$.

$$\rho'_{\ell} = \left(|0\rangle^{\otimes M}\langle 0|^{\otimes M}\right)_{\ell} \otimes \rho_{\ell-1}$$

2. The ℓ-th layer's M associated unitary matrices U_{ℓ}^{m} are applied to this tensor product state (from top to bottom).

$$\rho''_{\ell} = \left(\prod_{m=0}^{M-1} U_{\ell}^{M-m}\right)\left(\rho'_{\ell}\right)\left(\prod_{m=1}^{M} U_{\ell}^{m\dagger}\right)$$

Architecture: Step-by-step

For each layer ℓ,

1. The next layer's M qubits are prepared in the initial state $|0\rangle^{\otimes m}\langle 0|_{\ell}^{\otimes m}$ and tensor producted with the previous layer's output $\rho_{\ell-1}$.

$$\rho'_{\ell} = \left(|0\rangle^{\otimes M}\langle 0|^{\otimes M}\right)_{\ell} \otimes \rho_{\ell-1}$$

2. The ℓ-th layer's M associated unitary matrices U_{ℓ}^{m} are applied to this tensor product state (from top to bottom).

$$\rho''_{\ell} = \left(\prod_{m=0}^{M-1} U_{\ell}^{M-m}\right)(\rho'_{\ell})\left(\prod_{m=1}^{M} U_{\ell}^{m\dagger}\right)$$

3. The partial trace over the $(\ell-1)$th layer's Hilbert space is taken, resulting in the output state ρ_{ℓ} of the ℓ-th layer.

$$\rho_{\ell} = \mathrm{Tr}_{\ell-1}[\rho''_{\ell}]$$

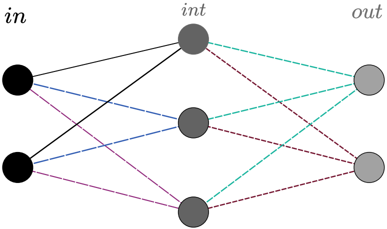
Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Simple Example: $2 \times 3 \times 2$

As a simple example, consider a QNN with one hidden layer of three qubits, and a two qubit input and output.

Simple Example: $2 \times 3 \times 2$

As a simple example, consider a QNN with one hidden layer of three qubits, and a two qubit input and output.



$$\mathrm{Tr}_{int}\Big[U_2^{out}U_1^{out}\Big(\mathrm{Tr}_{in}\Big[U_3^{int}U_2^{int}U_1^{int}(\rho_{in}\otimes|000\rangle\langle 000|_{int})U_1^{int\dagger}U_2^{int\dagger}U_3^{int\dagger}\Big]\Big)U_1^{out\dagger}U_2^{out\dagger}\Big] = \rho_{out}$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Cost

The metric by which we will judge the performance of the network on the training data is the cost, here taken as the average fidelity between the networks output state and the corresponding state given in training and explicity defined as:

$$C = \frac{1}{N}\sum_{i=1}^{N}\langle\psi_i^{out}|\rho_{out}|\psi_i^{out}\rangle$$

Note that this cost function is only applicable for training data based on pure states, for which the fidelity takes an especially nice form.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Cost

The metric by which we will judge the performance of the network on the training data is the cost, here taken as the average fidelity between the networks output state and the corresponding state given in training and explicity defined as:

$$C = \frac{1}{N}\sum_{i=1}^{N}\langle\psi_i^{out}|\rho_{out}|\psi_i^{out}\rangle$$

Note that this cost function is only applicable for training data based on pure states, for which the fidelity takes an especially nice form.

For input mixed states, we may replace the above with an averaged fidelity between output and target states of the form:

$$C = \frac{1}{N}\sum_{i=1}^{N}\left(\mathrm{Tr}\left[\sqrt{\sqrt{\rho_i}\rho_i^{out}\sqrt{\rho_i}}\right]\right)^2$$

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Training

We now wish to maximize the previously defined cost function (which has a maximum value of 1). This may be accomplished through training.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Training

We now wish to maximize the previously defined cost function (which has a maximum value of 1). This may be accomplished through training.

Training may be performed by evolving each unitary via the following map:

$$U_m^\ell \to e^{i\varepsilon K_m^\ell} U_m^\ell$$

which is parameterized by the step size ε, and where K_ℓ^m is derived from the derivative of the cost function and takes the following form:

$$K_m^\ell = \eta \frac{2^{m_\ell - 1}}{N} \sum_{i=1}^{N} \mathrm{Tr}_{\neg\ell} \Bigg[\left(\prod_{n=0}^{m-1} U_{m-n}^\ell \right) \left(\left(|0\rangle^{\otimes m_\ell} \langle 0|^{\otimes m_\ell} \right)_\ell \otimes \rho_i^{\ell-1} \right) \left(\prod_{n=1}^{m} U_m^{\ell\dagger} \right), \left(\prod_{n=m+1}^{m_\ell} U_n^{\ell\dagger} \right) \left(\sigma_i^\ell \otimes \mathbb{I}_{\ell-1} \right) \left(\prod_{n=1}^{m_\ell-(m+1)} U_{m_\ell-n}^\ell \right) \Bigg]$$

where the square brackets denote a commutator and $\sigma_i^\ell = \mathcal{F}^{\ell+1}(\ldots\mathcal{F}^{out}(\rho_i^{out})\ldots)$ is the adjoint channel to the layer-to-layer transition map ϵ^ℓ for layer ℓ.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Implementation: Software Choice

The project's task will be to implement this general structure in a quantum computational SDK.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Implementation: Software Choice

The project's task will be to implement this general structure in a quantum computational SDK.

The most likely candidate (as of now) is IBM's Qiskit, which may be developed in python.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Implementation: Software Choice

The project's task will be to implement this general structure in a quantum computational SDK.

The most likely candidate (as of now) is IBM's Qiskit, which may be developed in python. Other options include: Google's Cirq, the open source Qutip, Pennylane, etc.

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Implementation: Goals

Goals will include the following:

1. Define auxilary functions in an appropriate manner (partial trace, concatenated unitary actions, etc.)
2. Define layer-to-layer transition maps of arbitrary qubit size
3. Define arbitrary depth layer composition (network of arbitrary depth)
4. Define appropriate cost function
5. Implement corresponding training scheme
6. Test on set(s) of data
   (a) Compare performance on different underlying unitaries
   (b) Compare performance for data set size used in training
   (c) (*time permitting*) Compare performance on noisy data

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks

Beer, K., Bondarenko, D., Farrelly, T., Osborne, T. J., Salzmann, R., Scheiermann, D., and Wolf, R. (2020).
Training deep quantum neural networks.
Nature communications, 11(1):1–6.

Nielsen, M. A. and Chuang, I. (2002).
Quantum computation and quantum information.

Thanks!

Thanks for your time!

Overview
Data
Architecture
Example
Cost
Training
Implementation
Software
Goals
Thanks