Quantum recurrent neural network

Johannes Bausch. June 2020

Quantum information

Francesca Del Lungo

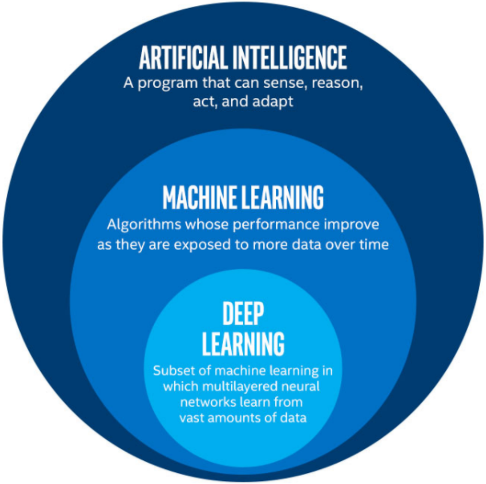
Prof. Paola Verrucchi & Prof. Filippo Caruso

23 July 2020

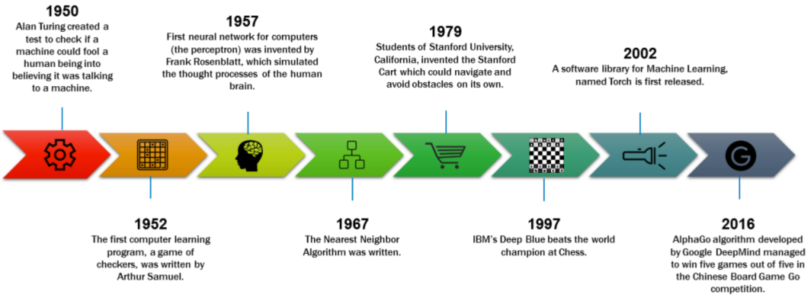
Contents overview

Machine learning

ML overview

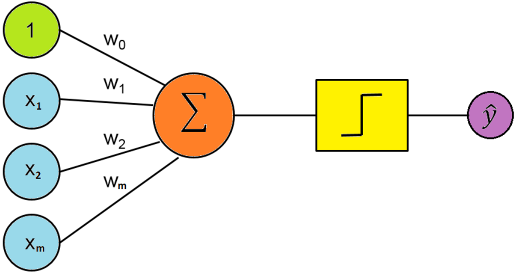

ARTIFICIAL INTELLIGENCE
A program that can sense, reason, act, and adapt
MACHINE LEARNING
Algorithms whose performance improve as they are exposed to more data over time
DEEP LEARNING
Subset of machine learning in which multilayered neural networks learn from vast amounts of data

Machine Learning timeline



Classical neural networks

Classical neural networks: perceptron[1]



Output

Linear combination of inputs

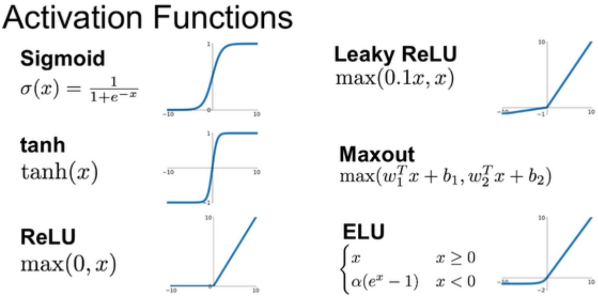
$$\hat{y} = g\left(w_0 + \sum_{i=1}^{m} x_i \, w_i \right)$$

Non-linear activation function

Bias

[1] *The perceptron: a probabilistic model for information storage and organization in the brain. Frank F. Rosenblatt. 1958*

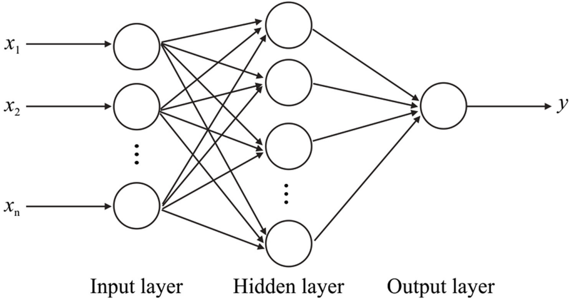
Activation function



- Introduces **non-linearity**
- Continuously differentiable (desiderable but not necessary)

Multilayer perceptron (MLP)

- **Feedforward** neural network
- Input layer, hidden layer(s), output layer



Sequence modeling: some examples I

Given the image of a ball, can you predict where it will go next ?

Sequence modeling: some examples II

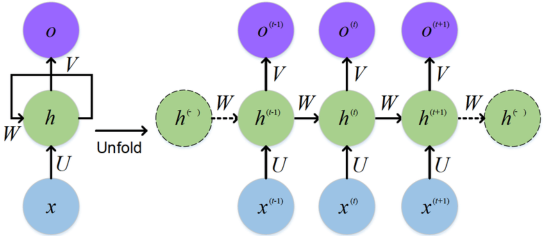
Given the image of a ball, can you predict where it will go next ?

Sequence modeling: some examples III

Given the image of a ball, can you predict where it will go next ?

"Vanilla" recurrent neural network

$$h_t = f_W(h_{t-1}, x_t)$$



Advantages[2]:

- input **sequences**
- network can **adapt** to quick changing input nodes

Disadvantages:

- cannot process very long sequences
- **vanishing or exploding gradient** problem in backpropagation

[2] *Recent Advances in Recurrent Neural Networks. Salehinejad, Hojjat and Sankar, Sharan and Barfett, Joseph and Colak, Errol and Valaee, Shahrokh. 2017*

GRU & LSTM

Gated Recurrent Unit (GRU) & Long Short Term Memory (LSTM)
Advantages:

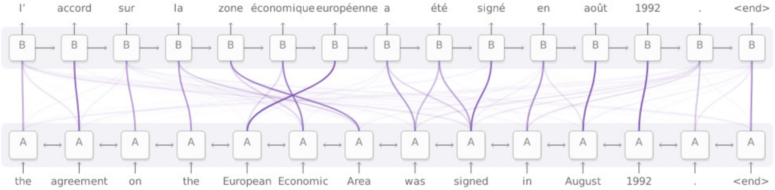
- **long-term** sequence **dependencies**
- more **robust** to the problem of vanishing gradients

Disadvantages:

- **more parameters** (increase the computing complexity compared to the RNN)
- **higher memory** required than the one of 'Vanilla' RNN
- typically **limited** to capturing about 200 tokens of context

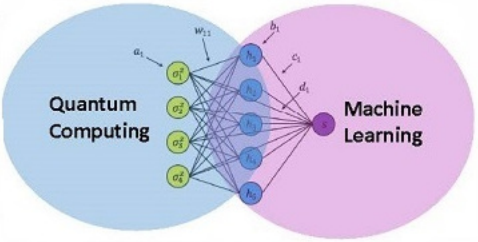
Other approaches

Other (non recurrent) approaches like **attention**, transformers ... (open field of research)



Quantum machine learning

Quantum machine learning



Quantum machine learning

Goal

Create a **quantum recurrent neural network** with a unitary cell that allows to remove the problem of gradient decay.

Quantum machine learning

Goal

Create a **quantum recurrent neural network** with a unitary cell that allows to remove the problem of gradient decay.

Problem

Classical neural networks

affine transformations + nonlinear activation functions = **non linear** transformations

Quantum circuit

unitary gates = **linear** transformations

Quantum neuron

Quantum neuron [3]

Idea: **Parametrized Quantum Gates**

$$\mathbf{R}(\theta) := exp(i\mathbf{Y}\theta)$$

where **Y** is the Pauli matrice that acts like:

$$\mathbf{R}(\theta) = exp\left(i\theta\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\right) = \begin{pmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{pmatrix}$$

[3] *Quantum Neuron: an elementary building block for machine learning on quantum computers. Yudong Cao, Gian Giacomo Guerreschi, and Alán Aspuru-Guzik. 2017*

Quantum neuron

Controlled rotation $\mathbf{cR}(\mathrm{i}, \theta_i)$ conditioned on the i-th qubit of state $|x\rangle$ for $\mathrm{x} \in \{0, 1\}^n$

The following map is generated:

$$\mathbf{R}(\theta_0)\mathbf{cR}(1, \theta_1)...\mathbf{cR}(n, \theta_n)|x\rangle|0\rangle = |x\rangle \left(cos(\eta)|0\rangle + sin(\eta)|1\rangle\right)$$

$$\text{where } \eta = \theta_0 + \sum_{i=1}^{n} \theta_i x_i$$

$$\mathrm{x} = (\mathrm{x}_1, ..., x_n) \in \{0, 1\}^n$$
$$\theta = (\theta_0, \theta_1, ..., \theta_n)$$

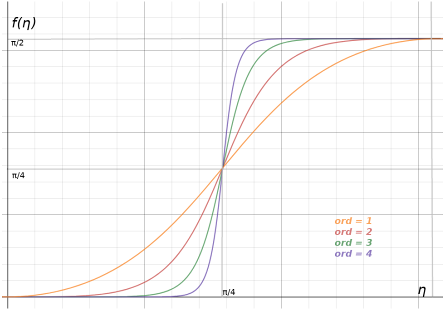
Quantum neuron - activation function

- Non-linearity ✓
- Sufficiently "flat" region $\times$

Quantum neuron

parameter **ord** ≥ 1:
"order" of the neuron

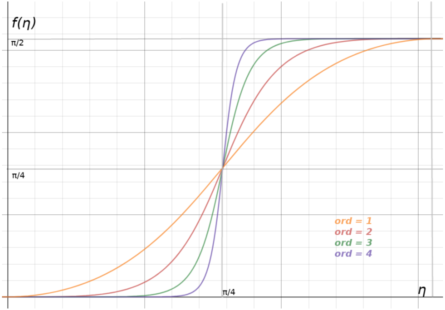
$$f(\eta) = \arctan\left(\tan(\eta)^{2^{ord}}\right)$$



Quantum neuron

parameter **ord** ≥ 1:
"order" of the neuron

$$f(\eta) = \arctan\left(\tan(\eta)^{2^{ord}}\right)$$



and so:

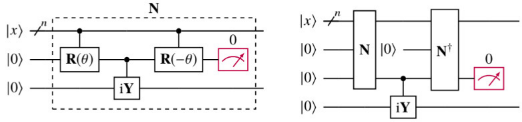
$$\cos f(\eta) = \frac{1}{\sqrt{1+\tan(\eta)^{2\times 2^{ord}}}} \qquad \sin f(\eta) = \frac{\tan(\eta)^{2^{ord}}}{\sqrt{1+\tan(\eta)^{2\times 2^{ord}}}}$$

The complete transformation is:

$$\mathbf{R}(\theta_0)\mathbf{cR}(1,\theta_1)...\mathbf{cR}(n,\theta_n)|x\rangle|0\rangle = |x\rangle\left(cos(f(\eta))|0\rangle + sin(f(\eta))|1\rangle\right)$$

Quantum neuron - RUS

Repeat-until-success (RUS) circuit



Algorithm 1 RUS circuit algorithm[4]

1: Prepare ancilla qubit in the state $|0\rangle$
2: result $\leftarrow$ apply rotation to the ancilla conditioned on input $|x\rangle$
3: **if** result $= |0\rangle$ **then**
4: end $\{$success$\}$
5: **else if** result $= |1\rangle$ **then**
6: do recovery operation and **repeat** from 2 $\{$failure$\}$
7: **end if**

---

[4] *Repeat-Until-Success: Non-deterministic decomposition of single-qubit unitaries. Paetznick, Adam and Svore, Krysta M. 2014*

Quantum neuron - superposition I

Problem

For states in **superposition** there is an *amplitude distortion*: the amplitudes depend on the history of success

Solution

Post-selection on measuring outcome 0

$\Downarrow$

Fixed-point oblivious amplitude amplification

Quantum neuron - superposition II

Amplitude amplification (AA)

- Variant of Grover search
- Given a state

$$|\Psi'\rangle = \alpha|0\rangle|x\rangle + \sqrt{1-\alpha^2}|1\rangle|y\rangle$$

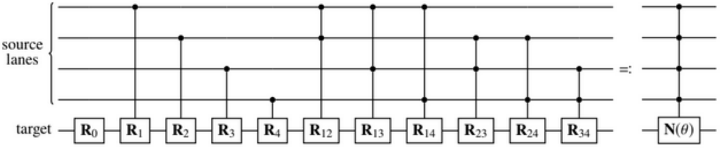
with likelihood $\propto |\alpha|^2$ to be measured in state $|0\rangle|x\rangle$.

After AA probability $\rightsquigarrow 1$.

Fixed-point oblivious amplitude amplification (FPOAA)

- Unitary **U** s.t. $U|\Psi\rangle = |\Psi'\rangle$
- α unknown

Quantum neuron with multi-control gates



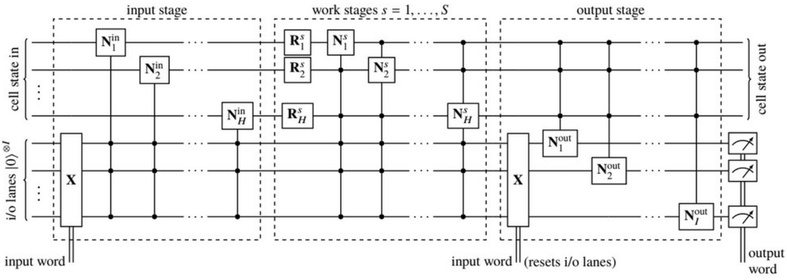
$$\eta' = \theta_0 + \sum_{i=1}^{n} \theta_i x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} \theta_{ij} x_i x_j + \ldots = \sum_{I \subseteq [n] \, |I| \leq d} \theta_I \prod_{i \in I} x_i$$

- Degree d = 2 controlled rotation
- n = 4 input neurons
- $R_I := R(\theta_I)$

Quantum recurrent neural networks

QRNN cell

Quantum neuron → quantum RNN cell → QRNN

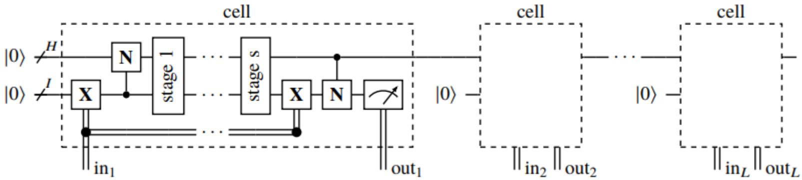


QRNN cell structure:

- **input** and **output lanes** (reset after each step)
- internal **cell state** (passed on to the next network iteration)

QRNN

Idea: iteratively apply QRNN cell to a sequence of input words $\text{in}_1, in_2, ..., in_L$.



- H: cell state workspace size
- I: input token width (in bits)
- ord: quantum neuron activation order

Experiments

Implementation & Training

Implementation details:

- PyTorch library
- Cross-entropy loss
- Different optimizers

Experiments

1. Sequence memorization
2. Finding structure in time
3. MNIST classification
4. Tests on long sequences

1. Sequence memorization I

Task

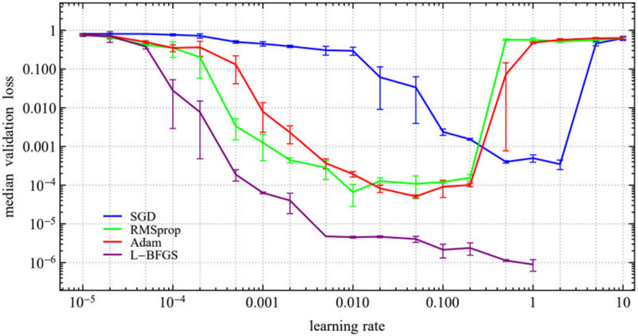
Reproduce the two sequences 44444...4 and 12312...3.

- QRNN setup:
  - 2 stages
  - neuron degree 3
  - workspace size of 5 (1162 parameters)
- Good convergence (with small network)

Goal

Benchmark **optimizer** and **learning rate** hyperparameters

1. Sequence memorization II



- **Validation loss** achieved after 500 training steps
- Median values over **5 runs**
- Chosen optimizer: Adam, lr: 0.05

2. Finding structure in time[5]

Task 1

Learning **XOR sequences**: binary strings $s = s_1 s_2 s_3 \ldots s_L$, so that

$$s_{3i} = s_{3i-1} \oplus s_{3i-2}$$

(each third digit is the XOR value of the preceding two)

QRNN setup:

- workspace size 4
- 1 work stage

Goal

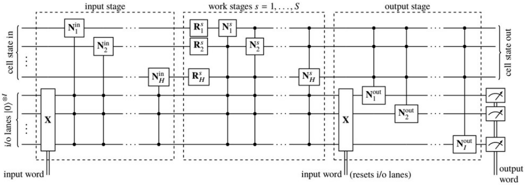
Explore which **parameter initialization** converges to a validation loss threshold of 10^{-3} first.

[5] Finding structure in time. J Elman. June 1990

2. Finding structure in time I

Two groups of parameters:

- **neurons** $N_j{}^i$ (bias R_0 and weights)
- **single-qubit unitaries** in the work stages $R_j{}^i$



- Parameters **initialization**: normal distribution with mean μ and width σ
- Most influential parameter: bias $\mu = \frac{\pi}{4}$

2. Finding structure in time II

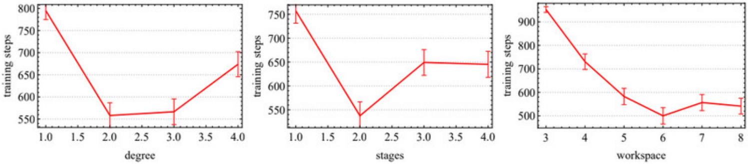
Task 2

Learning the **structure of a sentence** made up of the three words "ba", "dii" and "guuu".

Goal

How the QRNN **topology** influence **convergence speed**.

2. Finding structure in time III



best performance: degree 2, stages 2 and workspace 6.

3. MNIST classification I

Task

Handwritten integer digit classification.

- 70.000 images
- **Pre-processing** images: crop, downscale and binarize.

3. MNIST classification II

Digit Set	Method	Data Augmentation	Accuracy [%]
{0, 1}	QRNN (12 qubits, Adam)	-	99.2 ± 0.2
	ensemble of 4	-	$\mathbf{99.6} \pm \mathbf{0.16}$
{3, 6}	VQE (17 qubits)	ambiguous samples removed	98
	QRNN (12 qubits, Adam)	-	89.7 ± 0.8
	QRNN (10 qubits, L-BFGS)	-	97.1 ± 0.7
	ensemble of 6	-	$\mathbf{99.0} \pm \mathbf{0.3}$
full MNIST	VQE (10 qubits)	{even, odd} partitioned	82
	LSTM	-	98.2
	QRNN (10 qubits, Adam)	PCA, t-SNE	94.6 ± 0.4
	QRNN (13 qubits, Adam)	UMAP	96.7 ± 0.2
	ensemble of 3	UMAP	$\mathbf{99.23} \pm \mathbf{0.05}$

4. Long sequence tests I

Task

A **DNA sequence** consisting of the bases "G", "A", "T" and "C" is considered where a "U" is inserted in a random position. The network must identify the base following the "U".

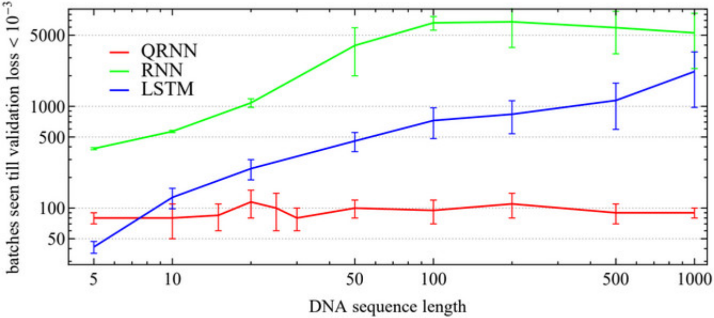
'AGAUATTCAGAAT' $\rightarrow$ **'A'**

Goal

Test the **gradient quality** for long sequences.

4. Long sequence tests II

The **number of steps** s.t. validation loss $< 10^{-3}$



- **QRNN**: 5 workspace size, 1 work stage $\rightarrow$ 837 parameters
- **RNN**: 1 layer, hidden layer size 22, 1 final linear layer $\rightarrow$ 888
- **LSTM**: 1 layer, hidden layer size 10, 1 final linear layer $\rightarrow$ 888

Conclusions

Conclusions & future works

Conclusions

- Recurrent model that deletes gradient decay for long sequences
- Data processing of far more than a few bits of size
- In real-word task RNN better than QRNN
  - **difficult to simulate** many qubit on a classical hardware
  - QRNN **simple architecture** compared to an RNN (or LSTM)

Future works

- More specialized circuit structure
- Variants in conjunction with other quantum ml algorithms

The end

"Nature isn't classical, damnit, so if you want to make a simulation of nature, you'd better make it quantum mechanical."
Richard Feynman

Thanks for the attention!

Rotation Operator

$$R(\theta) := exp(iY\theta)$$

where **Y** is the Pauli matrix
if operator A satisfies A^2=I, it can be shown that:

$$e^{i\theta A} = \cos\theta I + i\sin\theta A$$

so:

$$R(\theta) = exp\left(i\theta Y\right) = \cos\theta I + i\sin\theta Y =$$

$$\cos\theta\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} + i\sin\theta\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix} = \begin{pmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{pmatrix}$$

Limited-memory BFGS

L-BFGS uses an *estimate of the inverse Hessian matrix* to steer its search through variable space, but where BFGS stores a dense n $\times$ n approximation to the inverse Hessian (n being the number of variables in the problem), L-BFGS stores only a few vectors that represent the approximation implicitly.

Stochastic gradient descent (SGD)

Parameters update

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x, y)$$

$$\eta = \text{learning rate}$$

Adaptive Moment Estimation (Adam)

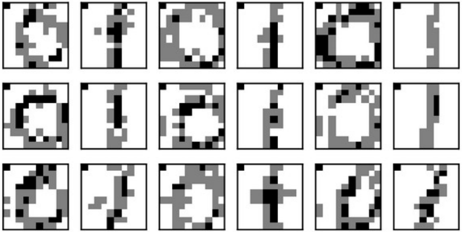
Techniques used to converge faster:

- Momentum
- Adaptive Learning Rates

QRNNs as generative models

Task

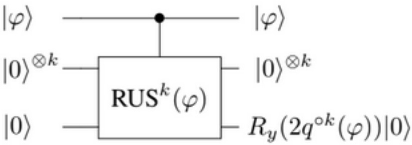
Re-generate handwritten digits starting from a '0' or '1' as input to the QRNN.



- network learns the **global structure** of the two digits
- **randomness** due to quantum measurements

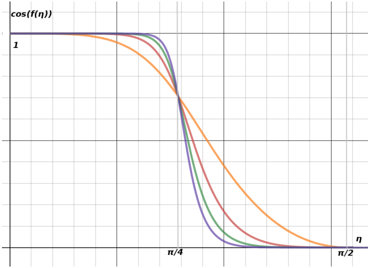
RUS multiple iterations

General k-iteration RUS circuit.

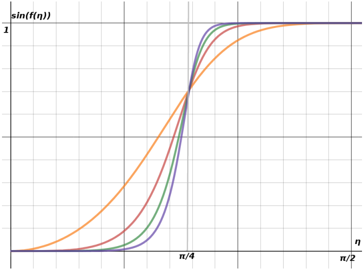


RUS threshold behaviour

$$R(f(\eta)) = \begin{cases} R(\pi/2)\,|0\rangle = |1\rangle & \textit{if}\ \eta > \frac{\pi}{4} \\ R(0)\,|0\rangle = |0\rangle & \textit{if}\ \eta < \frac{\pi}{4} \end{cases}$$



Cosine function

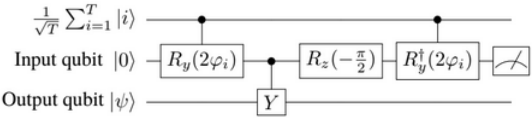


Sine function

$$\mathbf{R}(\theta_0)\mathbf{cR}(1,\theta_1)...\mathbf{cR}(n,\theta_n)|x\rangle|0\rangle = |x\rangle\left(cos(f(\eta))|0\rangle + sin(f(\eta))|1\rangle\right)$$

RUS superposition of states

Control register in a general superposition $\sum_{i=1}^{T} \alpha_i |i\rangle$



Single iteration of the circuit.
State of the system before measure is:

$$\sum_{i=1}^{T} \alpha_i |i\rangle \otimes \left[\sqrt{p(\varphi_i)} |0\rangle R(q(\varphi_i)) |0\rangle + \sqrt{1 - p(\varphi_i)} |1\rangle R\left(\frac{\pi}{4}\right) |0\rangle \right]$$

RUS superposition of states

Probability of measuring $|0\rangle$:

$$P_{|0\rangle} = \sum_{i=1}^{T} |\alpha_i|^2 p(\varphi_i)$$

yielding a state 0:

$$\sum_{i=1}^{T} \alpha_i \sqrt{\frac{p(\varphi_i)}{P_{|0\rangle}}} |i\rangle |0\rangle R(q(\varphi_i)) |0\rangle$$

Probability of measuring $|1\rangle$:

$$P_{|1\rangle} = \sum_{i=1}^{T} |\alpha_i|^2 (1 - p(\varphi_i))$$

yielding a state 1:

$$\sum_{i=1}^{T} \alpha_i \sqrt{\frac{1 - p(\varphi_i)}{P_{|1\rangle}}} |i\rangle |1\rangle R\left(\frac{\pi}{4}\right) |0\rangle$$

RUS superposition of states

Final state

In general if RUS fails r-1 times and succeeds (measuring $|0\rangle$) at the r-th trial, we have a state:

$$\sum_{i=1}^{T} \alpha_i \sqrt{\frac{(1-p(\varphi_i))^{r-1} p(\varphi_i)}{P_r}} |i\rangle |0\rangle R(q(\varphi_i)) |0\rangle$$

where:

$$P_r = \sum_{i=1}^{T} |\alpha_i|^2 (1-p(\varphi_i))^{r-1} p(\varphi_i)$$

Loss

Cross-entropy loss

For discrete probability distributions p and q :

$$H(p, q) = -\sum_{x \in X} p(x) \log q(x)$$