Resources

https://github.com/JQub/QuantumFlow_Tutorial (Source Code of All Hands-On in Tutorial)

https://github.com/JQub/qfnn (Source Code of QFNN API & Place to post Issues)

https://pypi.org/project/qfnn/ (Package of QFNN on PYPI)

https://libraries.io/pypi/qfnn/ (QFNN on Libraries.io)

https://jqub.ece.gmu.edu (JQub Website)

https://jqub.ece.gmu.edu/categories/QF (QuantumFlow Website for news and **slides**)

https://jqub.ece.gmu.edu/categories/QF/qfnn/ (QFNN Documents)

https://www.nature.com/articles/s41467-020-20729-5 (QuantumFlow Paper)

https://arxiv.org/pdf/2012.10360.pdf (Paper on How to Correct Map NN to Q)

https://arxiv.org/pdf/2109.03806.pdf (QF-Mixer)

https://arxiv.org/pdf/2109.03430.pdf (QF-RobustNN)

Tools to Be Used

Google CoLab

Github – Tutorial

Pytorch

Quirk

Qiskit

Tutorial on QuantumFlow: A Co-Design Framework of Neural Network and Quantum Circuit towards Quantum Advantage

Session 1: Introduction to Quantum Computing and Machine Learning

Weiwen Jiang, Ph.D.

Assistant Professor

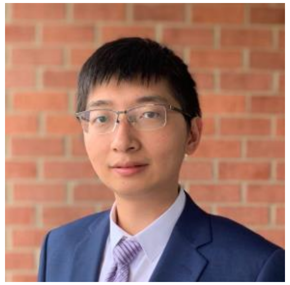
Electrical and Computer Engineering

George Mason University

wjiang8@gmu.edu

https://jqub.ece.gmu.edu

Presenter



Weiwen Jiang
Assistant Professor
Electrical and Computer Engineering (ECE)
George Mason University
Room3247, Nguyen Engineering Building
wjiang8@gmu.edu
(703)-993-5083
https://jqub.ece.gmu.edu/

- **Education Background**
  - Chongqing University (2013-2019)
  - **University of Pittsburgh (2017-2019)**
  - **University of Notre Dame (2019-2021)**
- **Research Interests**
  - **HW/SW Co-Design**
  - **Quantum Machine Learning**

First HW/SW Co-Design Framework using NAS

HW/SW Co-Design Framework
FNAS
[DAC'19*]
[TCAD'20*]

Application
- **Medical Imaging**: NAS for Medical Image Seg. [MICCAI'20]; 3D Cardiac MRI Seg. [ICCAD'20]
- **NLP (Transformer)**: FPGA [ICCD'20]; Mobile [DAC'21]; GPU [GLSVLSI'21]
- **Graph-Based**: Social Net [GLSVLSI'21]; Drug Discovery [ICCAD'21]

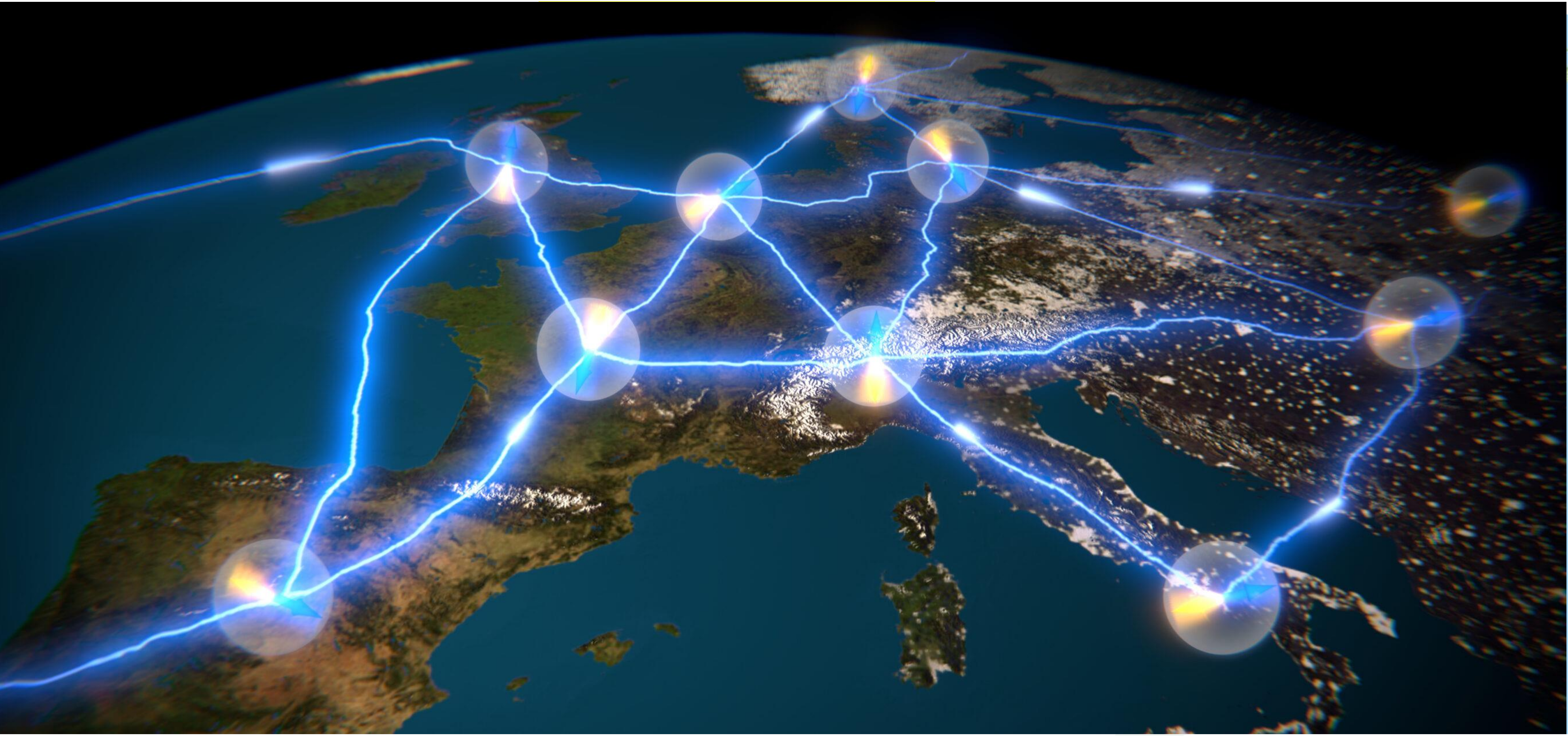
Algorithm
- **NAS Acc.**: HotNAS [CODES+ISSS'20]
- **Model Compression**: NAS for Quan. [ICCAD'19]; Compre.-Compilation [IJCAI'21]
- **Secure Infernece**: NASS [ECAI'20]; BUNET [MICCAI'20]

Hardware
- **FPGA**: XFER [CODES+ISSS'19*]
- **ASIC**: NANDS [ASP-DAC'20*]; ASICNAS [DAC'20]
- **Computing-in-Memory**: Device-Circuit-Arch. [IEEE TC'20]

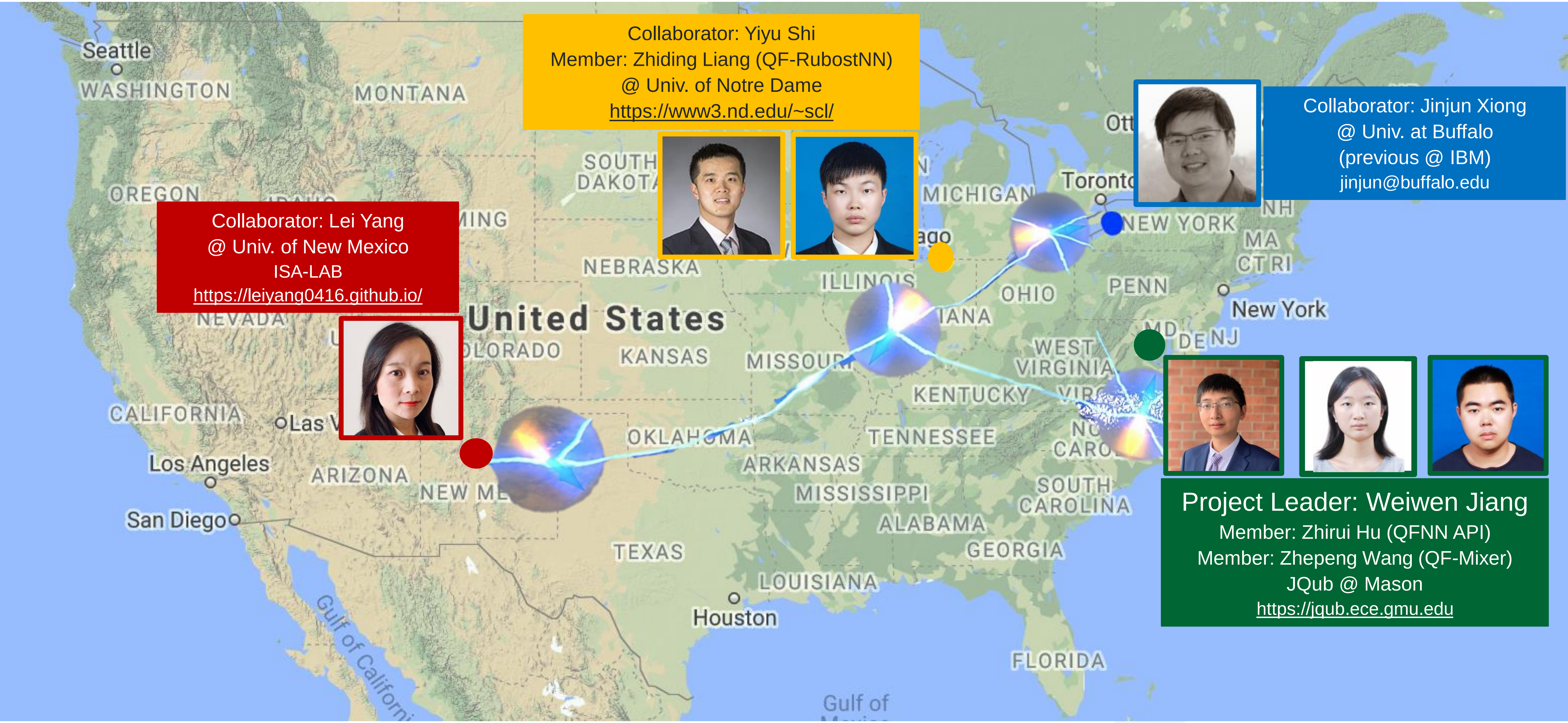
Best Paper Nominations:

- Multi-FPAG for NN
- FNAS: HW/SW Co-Design via NAS
- NANDS: NAS & NoC
- FNAS Journal Version

Entanglement of Qubits!



Entanglement of QuantumFlow Collaborators



Collaborator: Yiyu Shi
Member: Zhiding Liang (QF-RubostNN)
@ Univ. of Notre Dame
https://www3.nd.edu/~scl/

Collaborator: Jinjun Xiong
@ Univ. at Buffalo
(previous @ IBM)
jinjun@buffalo.edu

Collaborator: Lei Yang
@ Univ. of New Mexico
ISA-LAB
https://leiyang0416.github.io/

Project Leader: Weiwen Jiang
Member: Zhirui Hu (QFNN API)
Member: Zhepeng Wang (QF-Mixer)
JQub @ Mason
https://jqub.ece.gmu.edu

Our Quantum Works

Published Papers:

[1] Weiwen Jiang, Jinjun Xiong, and Yiyu Shi. "A co-design framework of neural networks and quantum circuits towards quantum advantage." ***Nature communications*** 12.1 (2021): 1-13.

[2] Weiwen Jiang, Jinjun Xiong, and Yiyu Shi. "When Machine Learning Meets Quantum Computers: A Case Study." *Asia and South Pacific Design Automation Conference* ***(ASP-DAC)***. IEEE, 2021.

[3] Zhepeng Wang, Zhiding Liang, Shangling Zhou, Caiwen Ding, Jinjun Xiong, Yiyu Shi, Weiwen Jiang, "Exploration of Quantum Neural Architecture by Mixing Quantum Neuron Designs." *International Conference On Computer-Aided Design (**ICCAD**), IEEE/ACM, 2021.*

[4] Zhiding Liang, Zhepeng Wang, Junhuan Yang, Lei Yang, Jinjun Xiong, Yiyu Shi, Weiwen Jiang, "Can Noise on Qubits Be Learned in Quantum Neural Network? A Case Study on QuantumFlow." *International Conference On Computer-Aided Design (**ICCAD**), IEEE/ACM, 2021.*

Invited Talks:

[1] Weiwen Jiang, "A Co-Design Framework of Neural Networks and Quantum Circuits Towards Quantum Advantage." ***IBM Quantum Summit 2020.***

[2] Weiwen Jiang, "Tutorial on QuantumFlow: A Co-Design Framework of Neural Network and Quantum Circuit towards Quantum Advantage.“ **ESWEEK 2021**

[3] Weiwen Jiang, "Tutorial on QuantumFlow: An End-to-End Quantum Neural Network Acceleration Framework.“ **QuantumWEEK 2021**

Agenda

- **Session 1: Introduction <u>(12:45 - 13:30)</u>**
- **Session 2: QuatnumFlow Co-Design Framework <u>(13:40 - 14:40)</u>**
- **Session 3: QFNN: Open-Source Library <u>(14:40 - 14:50)</u>**
- **Session 4: QF-Mixer and QF-RobustNN <u>(15:00 - 16:20)</u>**
- **Session 5: Roadmap <u>(16:30 - 17:15)</u>**

Agenda – Session 1: Introduction

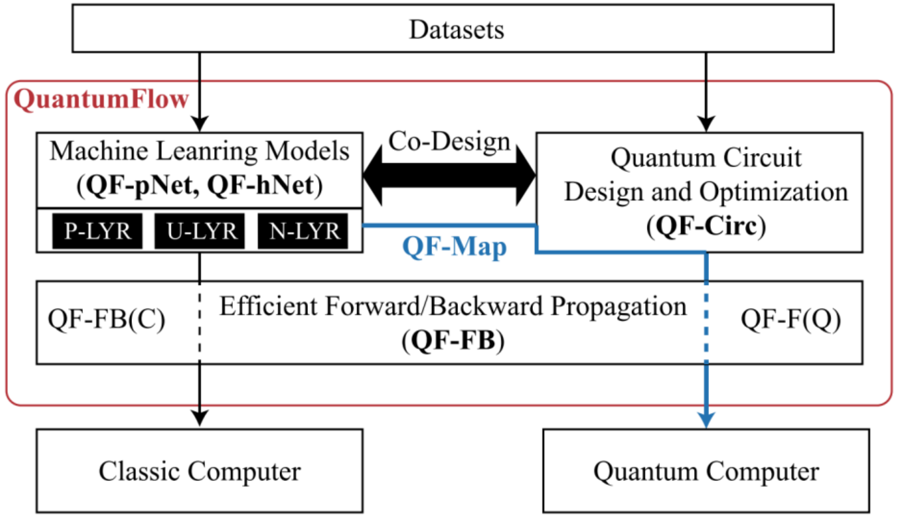
- **Introduction to Quantum Computing**
  - From Bit to Qubit
  - From Logic Gates to Quantum Logic Gates
  - *Colab Hands-On (1): Basic Quantum Gates*
- **Introduction to Machine Learning**
  - Why Neural Networks
  - Biological Neuron
  - Artificial Neuron and Neural Network
  - Learning
- **Why Quantum Machine Learning**

Agenda – Session 2: QuantumFlow

- **General Framework for Quantum-Based Neural Network Accelerator**
  - Data Preparation and Encoding
  - *Colab Hands-On (2): From Classical Data to Quantum Data*
  - Quantum Circuit Design
  - *Colab Hands-On (3): A Quantum Neuron*
- **Co-Design toward Quantum Advantage**
  - Challenges?
  - Feedforward Neural Network
  - *Colab Hands-On (4): End-to-End Neural Network on MNIST*
  - Optimization for Quantum Neuron
  - *Colab Hands-On (5): QuantumFlow*
  - Results

Agenda – Session 3: QFNN API

- **Introduction to QFNN**
  - Structure: qf_circ, qf_net, qf_fb, qf_map
- **Building QuantumFlow using QFNN**
  - QF-pNet
  - QF-hNet
  - QF-FB
- **Beyond QuantumFlow with QFNN**
  - FFNN
  - VQC
  - QF-Mixer



Agenda – Session 4: Extensions

- **QF-Mixer: Exploring Quantum Neural Architecture**
  - Motivation: Existing Quantum Neuron Designs Can Be Complementary
  - Design Principle: Mixing Designs is Harder Than Your Thoughts!
  - Results
- **QF-RobustNN: Learning Noise in Quantum Neural Networks**
  - Introduction to Noise in Quantum Computing
  - Motivation: Error Can Corrupt Quantum NN and Compiling Leads to Lengthy Learning
  - Application-Specific Compiler is Needed
  - Results
- **Open Questions and Future Work**

Agenda – Session 5: Roadmap

- **Roadmap of Quantum Machine Learning**
- **Call for paper at “Electronics”**
- **Conclusion**
- **Q&A**

Tools to Be Used

Google CoLab

Github – Tutorial

Pytorch

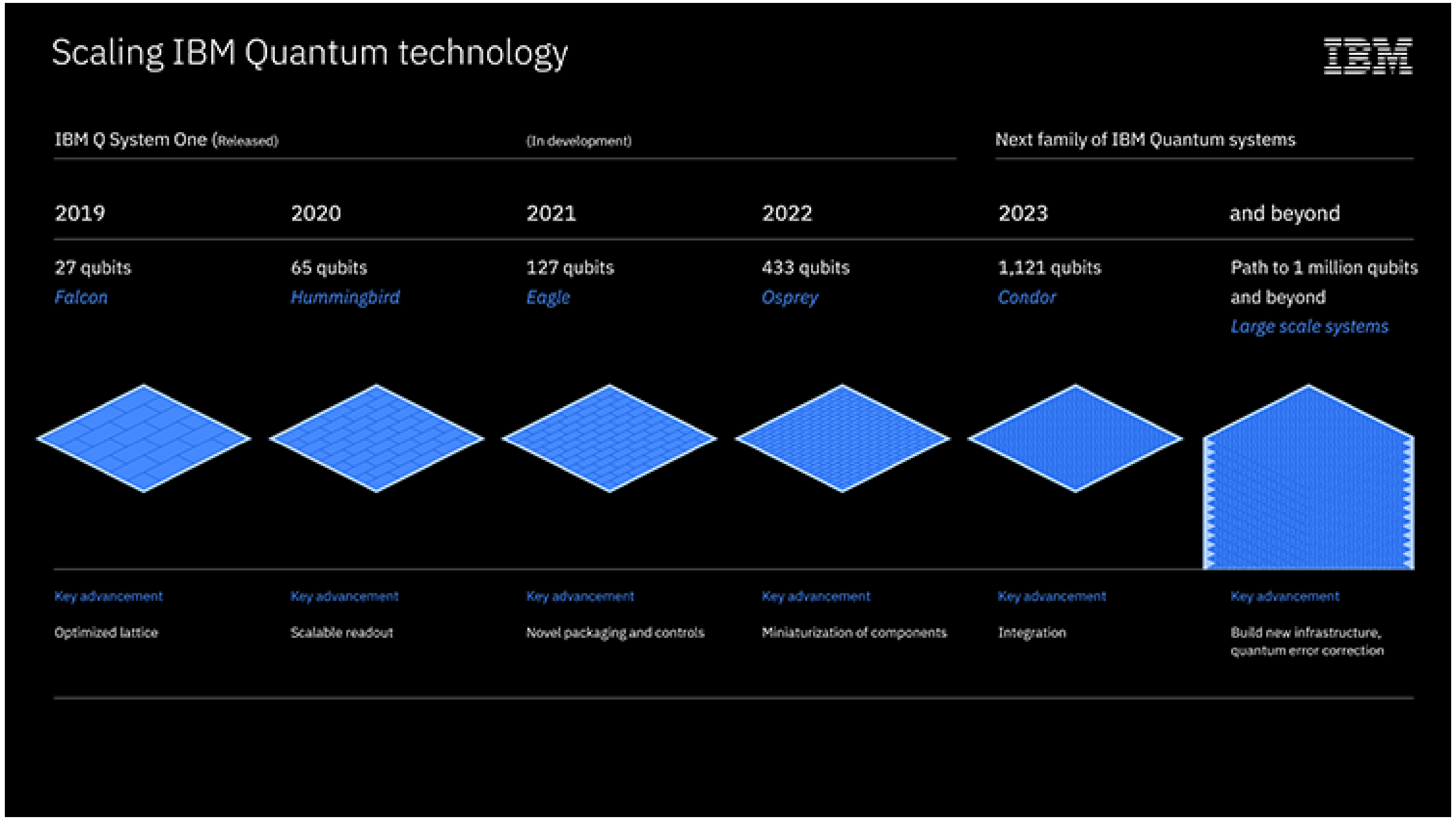
Quirk

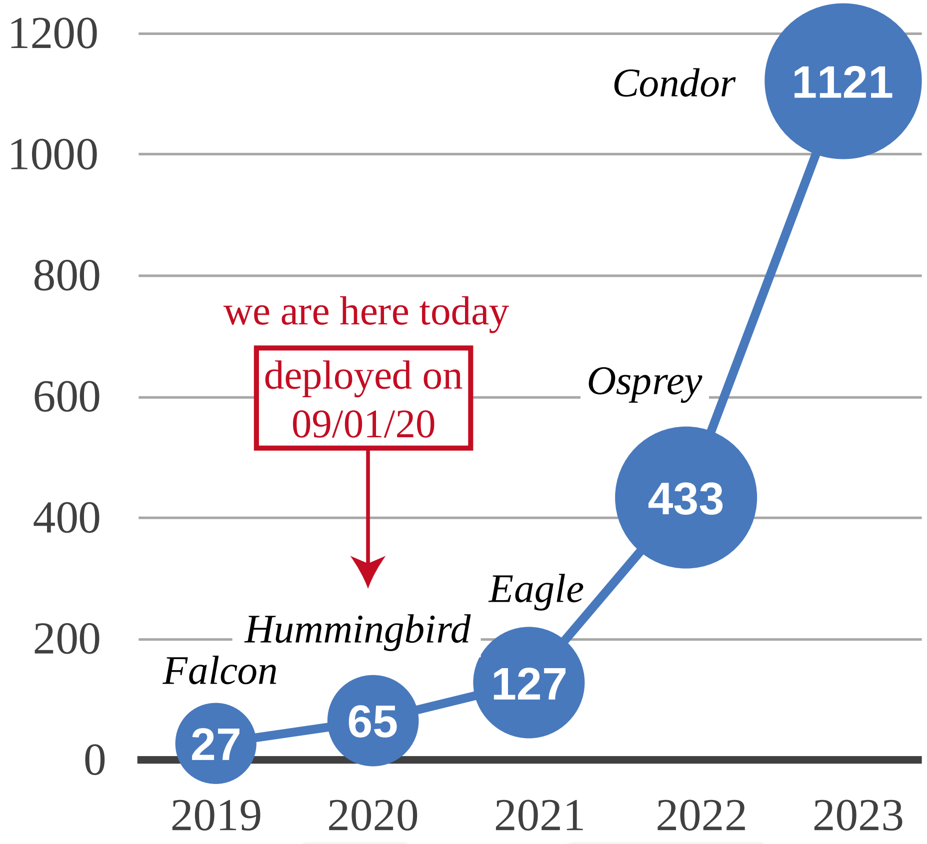
Qiskit

Agenda – Session 1: Introduction

- **Introduction to Quantum Computing**
  - From Bit to Qubit
  - From Logic Gates to Quantum Logic Gates
  - *Colab Hands-On (1): Basic Quantum Gates*
- **Introduction to Machine Learning**
- **Why Quantum Machine Learning**

Consistently Increasing Qubits in Quantum Computers





The Power of Quantum Computers: Qubit

Classical Bit

$$X = 0 \textbf{ or } 1$$

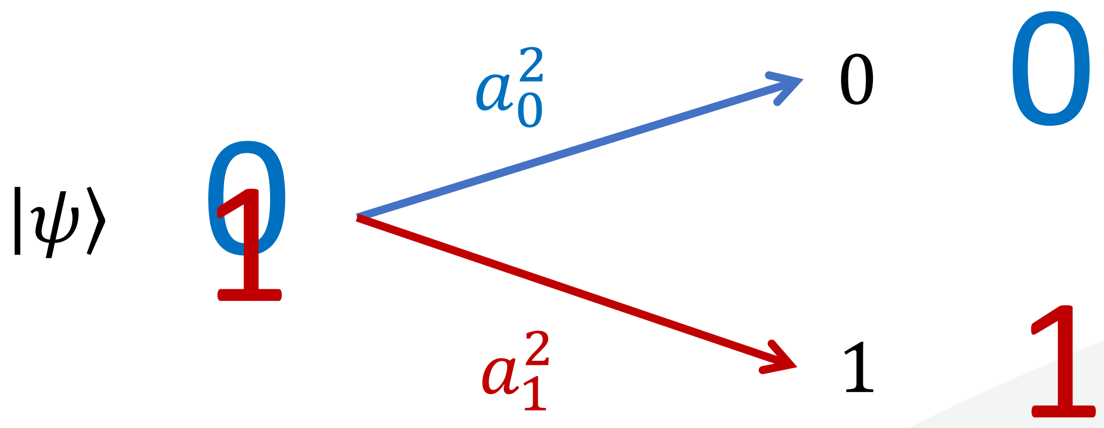
Quantum Bit (Qubit)

$$|\psi\rangle = |0\rangle \textbf{ and } |1\rangle$$

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle$$

$$\text{s.t. } a_0^2 + a_1^2 = 100\%$$

Reading out Information from Qubit (Measurement)



Probability

Non–Deterministic Computing

$$a_0^2 + a_1^2 = 100\%$$

$$40\% + 60\% = 100\%$$

The Power of Quantum Computers: Qubit

Classical Bit

$$X = 0 \textbf{ or } 1$$

Quantum Bit (Qubit)

$$|\psi\rangle = |0\rangle \textbf{ and } |1\rangle$$

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle$$

$$\text{s.t. } a_0^2 + a_1^2 = 100\%$$

Representation:

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}$$

Initially:

$$|\psi\rangle = |0\rangle$$

, where $a_0 = 1$ and $a_1 = 0$

$$|\psi\rangle = |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

The Power of Quantum Computers: Qubits

2 Classical Bits

00 **or** 01 **or** 10 **or** 11

n bits for 1 value
$\boldsymbol{x \in [0, 2^n - 1]}$

2 Qubits

$c_{00}|00\rangle$ **and** $c_{01}|01\rangle$ **and**
$c_{10}|10\rangle$ **and** $c_{11}|11\rangle$

n bits for 2^n values
$\boldsymbol{a_{00}, a_{01}, a_{10}, a_{11}}$

Qubits: $\boldsymbol{q_0, q_1}$

$$|q_0\rangle = a_0|0\rangle + a_1|1\rangle$$

$$|q_1\rangle = b_0|0\rangle + b_1|1\rangle$$

$$\boldsymbol{|q_0, q_1\rangle = |q_0\rangle \otimes |q_1\rangle}$$
$$\boldsymbol{= c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle}$$

- $|00\rangle$: Both q_0 and q_1 are in state $|0\rangle$
- c_{00}^2: Probability of both q_0 and q_1 are in state $|0\rangle$
- $c_{00}^2 = a_0^2 \times b_0^2$
- $c_{00} = \sqrt{a_0^2 \times b_0^2} = a_0 \times b_0$

The Power of Quantum Computers: Qubits

2 Classical Bits

00 **or** 01 **or** 10 **or** 11

n bits for 1 value
$\boldsymbol{x \in [0, 2^n - 1]}$

2 Qubits

$c_{00}|00\rangle$ **and** $c_{01}|01\rangle$ **and** $c_{10}|10\rangle$ **and** $c_{11}|11\rangle$

n bits for 2^n values
$\boldsymbol{a_{00}, a_{01}, a_{10}, a_{11}}$

Qubits: $\boldsymbol{q_0, q_1}$

$$|q_0\rangle = a_0|0\rangle + a_1|1\rangle$$

$$|q_1\rangle = b_0|0\rangle + b_1|1\rangle$$

$$\boldsymbol{|q_0, q_1\rangle = |q_0\rangle \otimes |q_1\rangle}$$
$$\boldsymbol{= c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle}$$

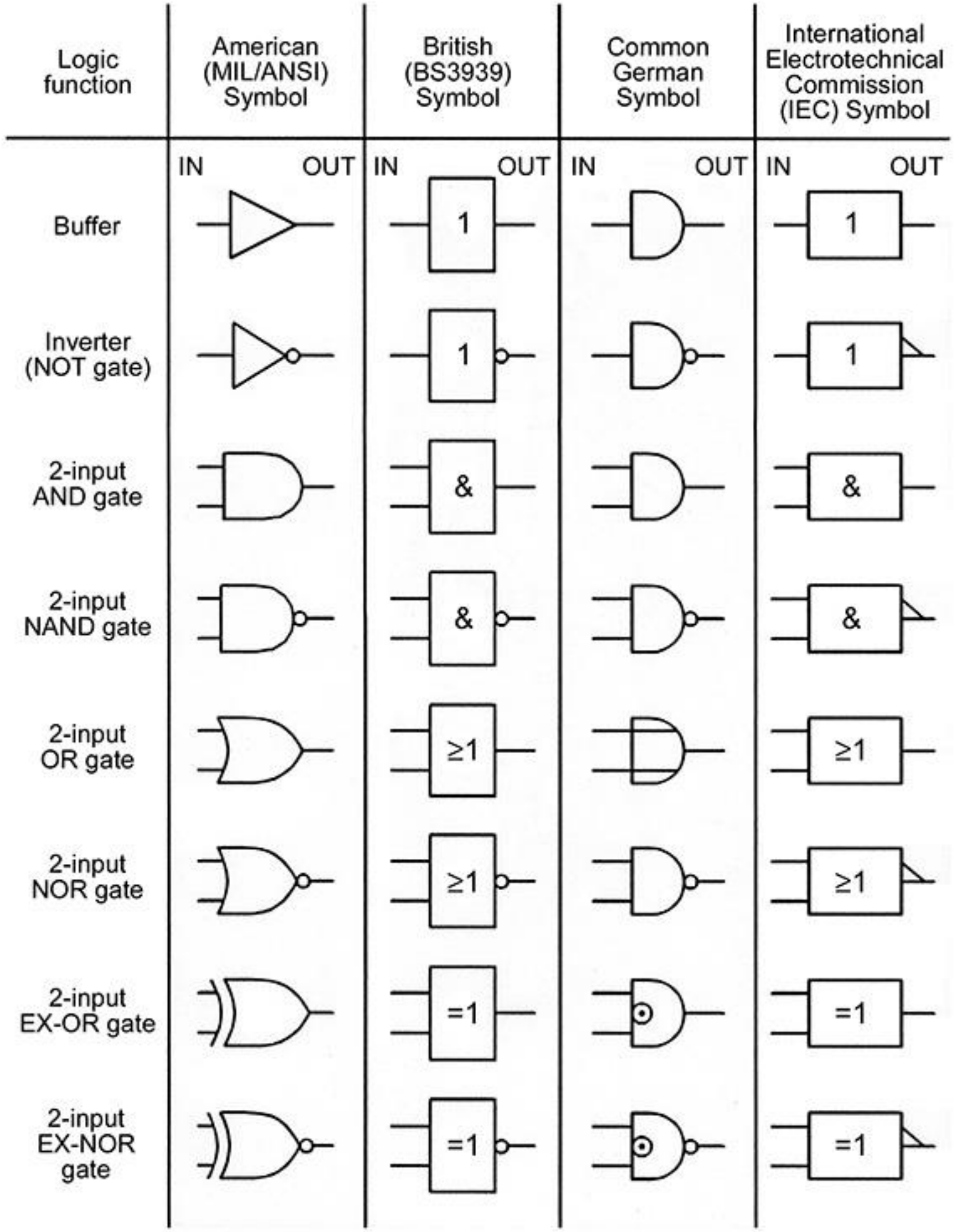
$$\boldsymbol{|q_0, q_1\rangle = |q_0\rangle \otimes |q_1\rangle = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} \otimes \begin{pmatrix} b_0 \\ b_1 \end{pmatrix}}$$

$$\boldsymbol{= \begin{pmatrix} a_0 \times \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} \\ a_1 \times \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{pmatrix} = \begin{pmatrix} c_{00} \\ c_{01} \\ c_{10} \\ c_{11} \end{pmatrix}}$$

Agenda – Session 1: Introduction

- **Introduction to Quantum Computing**
  - From Bit to Qubit
  - From Logic Gates to Quantum Logic Gates
  - *Colab Hands-On (1): Basic Quantum Gates*
- **Introduction to Machine Learning**
- **Why Quantum Machine Learning**

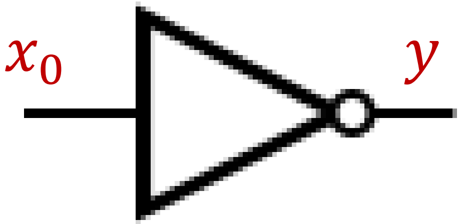
Logic Gates v.s. Quantum Logic Gates



Operator	Gate(s)	Matrix
Pauli-X (X)	$\boxed{X}$, $\oplus$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)	$\boxed{Y}$	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)	$\boxed{Z}$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)	$\boxed{H}$	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Phase (S, P)	$\boxed{S}$	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\pi/8$ (T)	$\boxed{T}$	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$
Controlled Not (CNOT, CX)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Controlled Z (CZ)	$\boxed{Z}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
SWAP		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Toffoli (CCNOT, CCX, TOFF)		$\begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0 \end{bmatrix}$

Logic Gates v.s. Quantum Logic Gates

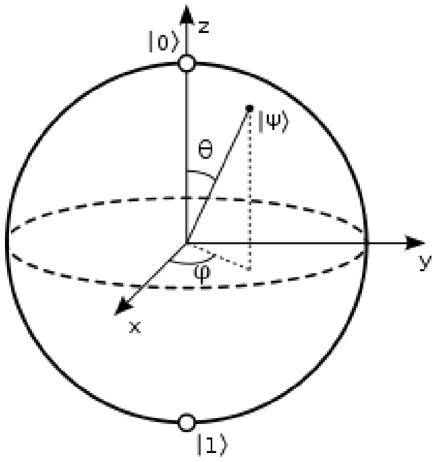
Single-bit Gate



Not Gate

x_0	y
0	1
1	0

Single-Qubit Gates



- **Pauli operators: X, Y, Z Gates**
- Hadamard gate: H Gate
- General gate: U Gate

$|0\rangle$ — $\boxed{X}$ — $\qquad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

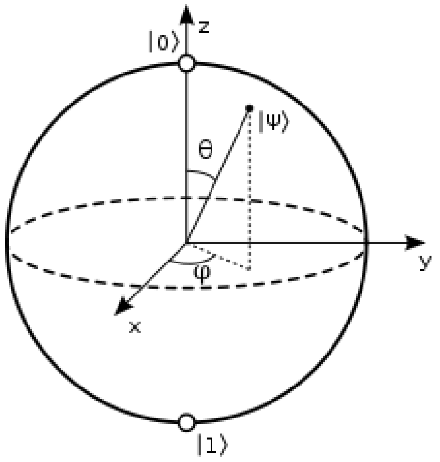
$|1\rangle$ — $\boxed{Z}$ — $\qquad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

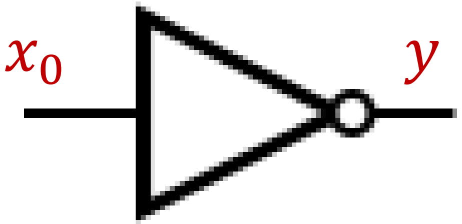
$|0\rangle \qquad |1\rangle$

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

Superposition



Single-bit Gate



Not Gate

x_0	y
0	1
1	0

Single-Qubit Gates

- Pauli operators: X, Y, Z Gates
- **Hadamard gate: H Gate**
- **General gate: U Gate**

$$|0\rangle \;—\boxed{H}—\quad \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$$

$$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \times \begin{bmatrix}1\\ 0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\ 1\end{bmatrix}$$

$$|0\rangle \;—\boxed{U}—\quad \begin{bmatrix}\cos(\theta/2) & -e^{i\lambda}\sin(\theta/2)\\ e^{i\phi}\sin(\theta/2) & e^{i(\phi+\lambda)}\cos(\theta/2)\end{bmatrix}$$

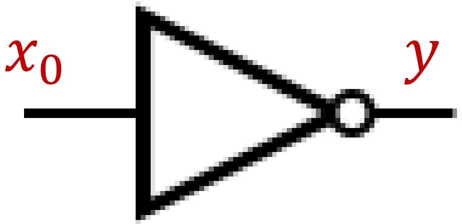
$$R_x(\theta) = \exp(-iX\theta/2) = \begin{bmatrix}\cos(\theta/2) & -i\sin(\theta/2)\\ -i\sin(\theta/2) & \cos(\theta/2)\end{bmatrix},$$

$$R_y(\theta) = \exp(-iY\theta/2) = \begin{bmatrix}\cos(\theta/2) & -\sin(\theta/2)\\ \sin(\theta/2) & \cos(\theta/2)\end{bmatrix},$$

$$R_z(\theta) = \exp(-iZ\theta/2) = \begin{bmatrix}\exp(-i\theta/2) & 0\\ 0 & \exp(i\theta/2)\end{bmatrix}.$$

Single-Qubit Gates in Parallel

Single-bit Gate

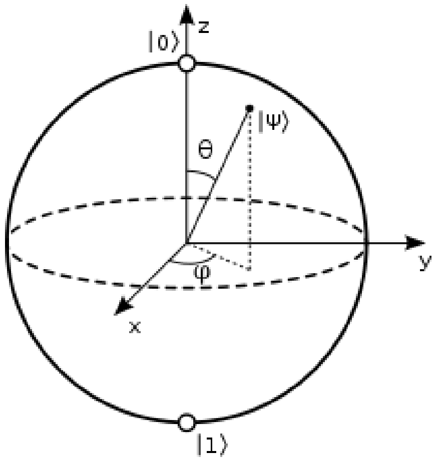


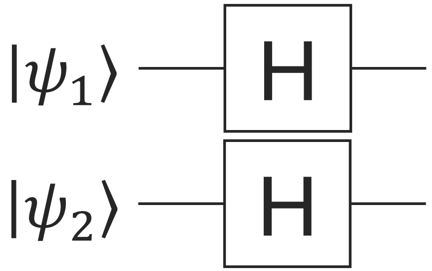
Not Gate

x_0	y
0	1
1	0

Single-Qubit Gates

- **Pauli operators: X, Y, Z Gates**
- Hadamard gate: H Gate
- General gate: U Gate





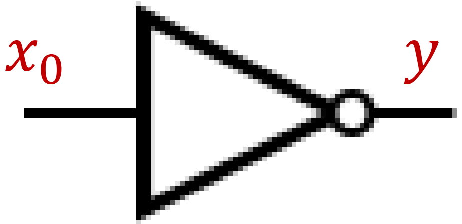
$$|00\rangle = \begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$

$$H^{\otimes 2} = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1 & 1 & 1 & 1\\1 & -1 & 1 & -1\\1 & 1 & -1 & -1\\1 & -1 & -1 & 1\end{bmatrix}$$

$$H^{\otimes 2}|00\rangle = \frac{1}{2}\begin{bmatrix}1 & 1 & 1 & 1\\1 & -1 & 1 & -1\\1 & 1 & -1 & -1\\1 & -1 & -1 & 1\end{bmatrix} \times \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1\\1\\1\\1\end{bmatrix}$$

Single-Qubit Gates in Parallel

Single-bit Gate

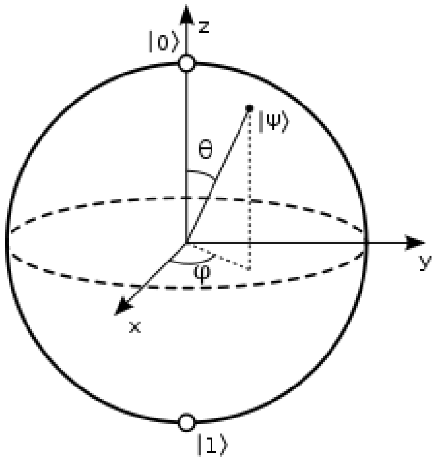


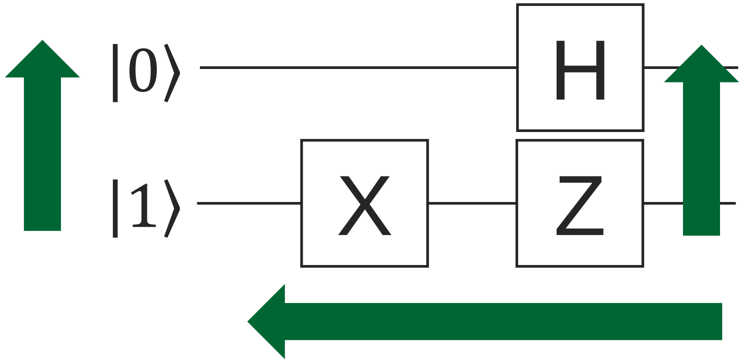
Not Gate

x_0	y
0	1
1	0

Single-Qubit Gates

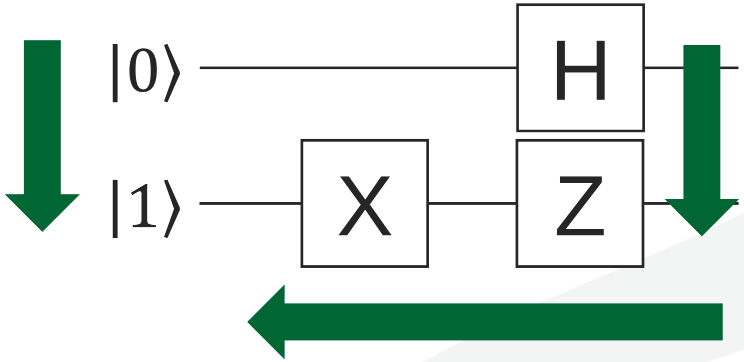
- **Pauli operators: X, Y, Z Gates**
- Hadamard gate: H Gate
- General gate: U Gate





Qiskit / Quirk

$$|\psi\rangle = (Z \otimes H) \times (X \otimes I) \times |10\rangle$$



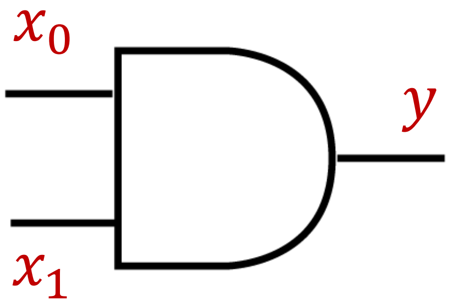
Wiki Quantum Gate

This Tutorial

$$|\psi\rangle = (H \otimes Z) \times (I \otimes X) \times |01\rangle$$

Logic Gates v.s. Quantum Logic Gates

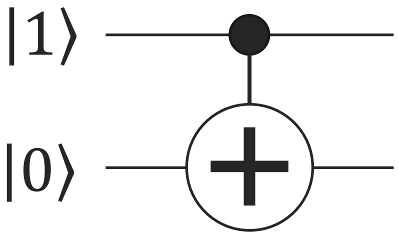
Two-bits Gate

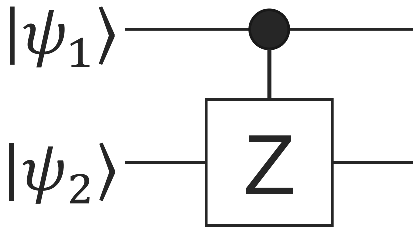


AND Gate

x_0	x_1	y
0	0	0
0	1	0
1	0	0
1	1	1

- Multi-Qubit Gates
  - **Controlled-Pauli gates**
  - **Toffoli gate or CCNOT**
  - ……





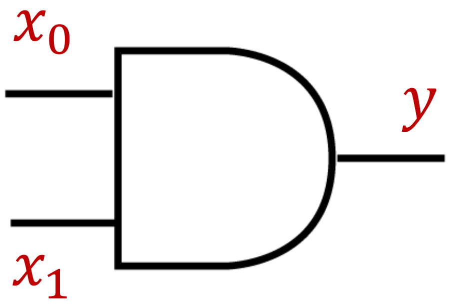
$$|10\rangle = \begin{bmatrix}0\\1\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} \quad |11\rangle = \begin{bmatrix}0\\1\end{bmatrix} \otimes \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

$$CZ \times |\psi_1\psi_2\rangle = \begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & -1\end{bmatrix} \times \begin{bmatrix}a\\b\\c\\d\end{bmatrix} = \begin{bmatrix}a\\b\\c\\-d\end{bmatrix}$$

$$CNOT \times |10\rangle = \begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix} \times \begin{bmatrix}0\\0\\1\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

Entanglement

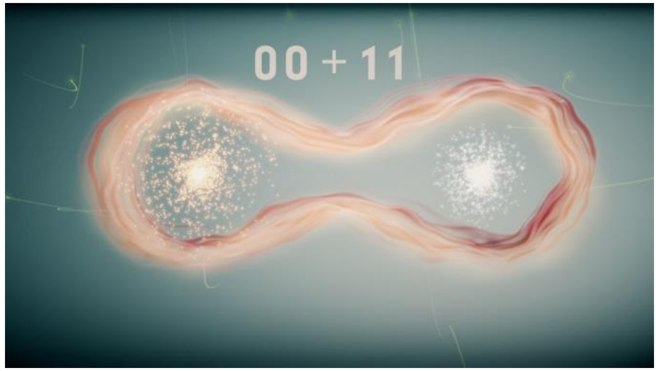
Two-bits Gate

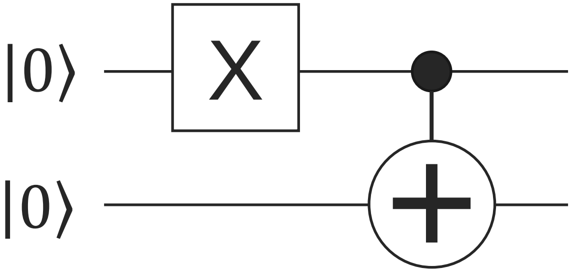


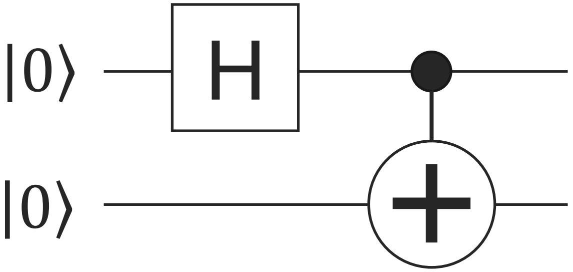
AND Gate

x_0	x_1	y
0	0	0
0	1	0
1	0	0
1	1	1

- Multi-Qubit Gates
  - **Controlled-Pauli gates**
  - **Toffoli gate or CCNOT**
  - ……







$$CNOT \times |10\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = |1\rangle \otimes |1\rangle$$

$$CNOT \times (H \otimes I) \times |00\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

$$\times |00\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{array}{c} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \end{array} \begin{array}{l} |00\rangle \\ |01\rangle \\ |10\rangle \\ |11\rangle \end{array}$$

Agenda – Session 1: Introduction

- **Introduction to Quantum Computing**
  - From Bit to Qubit
  - From Logic Gates to Quantum Logic Gates
  - *Colab Hands-On (1): Basic Quantum Gates*
- **Introduction to Machine Learning**
- **Why Quantum Machine Learning**

Hands-On Tutorial (1)
Basic Quantum Gates

Agenda – Session 1: Introduction

- **Introduction to Quantum Computing**
- **Introduction to Machine Learning**
  - Why Neural Networks
  - Biological Neuron
  - Artificial Neuron and Neural Network
  - Learning
- **Why Quantum Machine Learning**

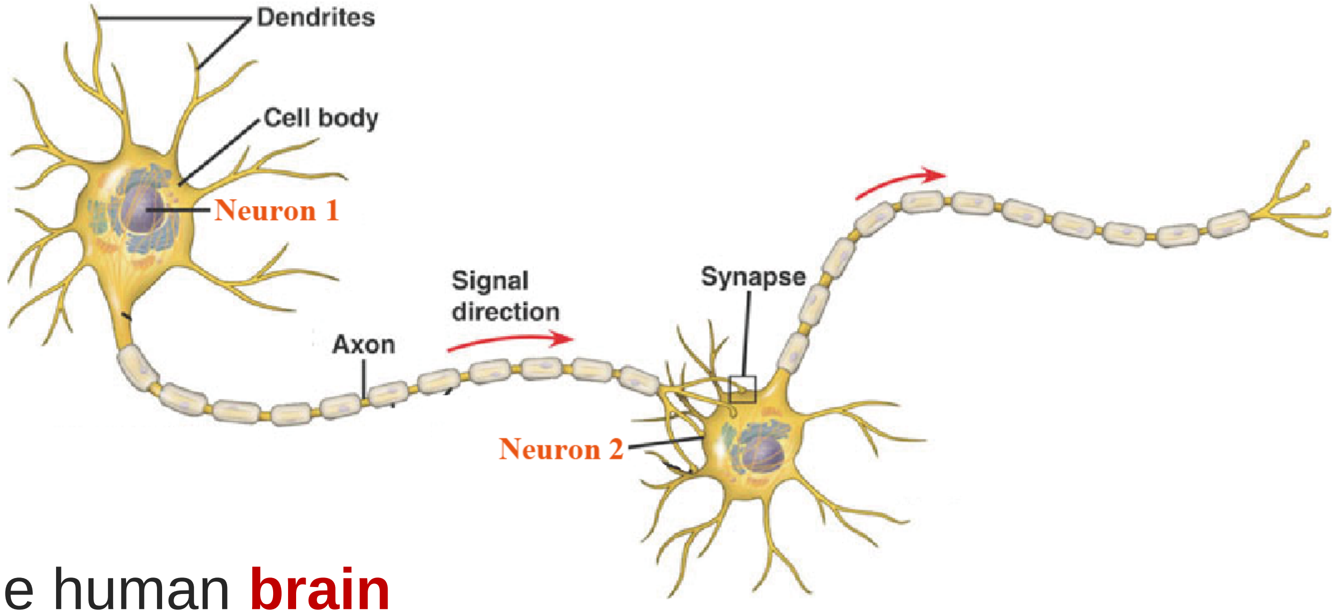
Why Neural Networks

- **An emulation of the biological neural systems**
  - Parallel computation
  - Adaptive connections
- **Very different style from sequential computation**
  - Should be good for things that brains are good at (e.g., vision)
  - Should be bad for things that brains are bad at (e.g., 23 x 7!)
- **To solve practical problems by using novel learning algorithms inspired by the brain**
  - Learning algorithms can be very useful even if they are not how the brain actually works.

Agenda – Session 1: Introduction

- **Introduction to Quantum Computing**
- **Introduction to Machine Learning**
  - Why Neural Networks
  - Biological Neuron
  - Artificial Neuron and Neural Network
  - Learning
- **Why Quantum Machine Learning**

Biological Neuron



Human intelligence reside

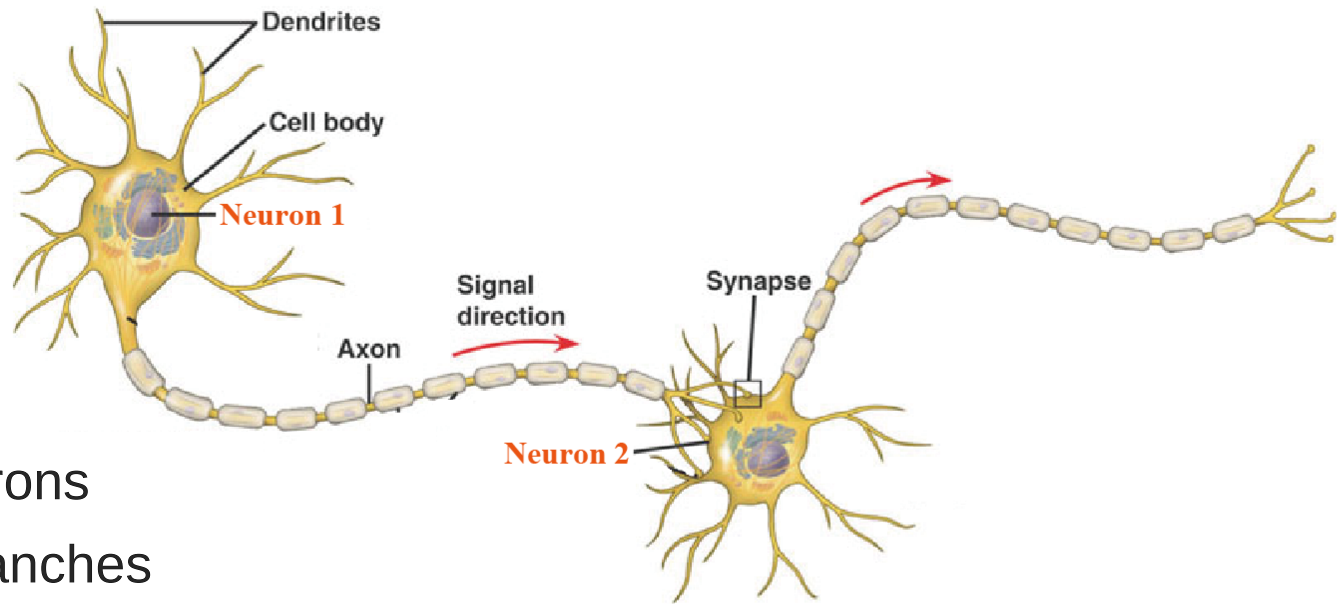
in the brain:

- Approximately **86 billion neurons** in the human **brain**
- The brain is a **network** of **neurons**, connected with nearly $10^{14} - 10^{15}$ **synapses**

How to equip intelligence in the machine?

- **To understand how the brain network is constructed**
- **To mimic the brain**

Biological Neuron



Neurons work together:

- **Cell body** process the information
- **Dendrites** receive messages from other neurons
- **Axon** transmit the output to many smaller branches
- **Synapses** are the contact points between axon (Neuron 1) and dendrites (Neuron 2) for message passing

Cell body receives input signal from **dendrites** and produce output signal along **axon**, which interact with the next neurons via **synaptic weights**

Synaptic weights are **learnable** to perform useful computations

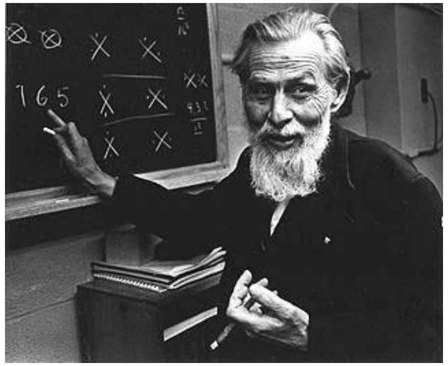
(e.g., Recognizing objects, understanding language, making plans, controlling the body.)

Agenda – Session 1: Introduction

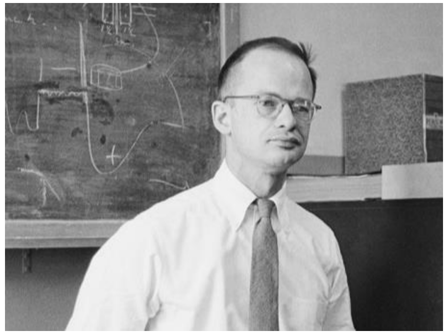
- **Introduction to Quantum Computing**
- **Introduction to Machine Learning**
  - Why Neural Networks
  - Biological Neuron
  - Artificial Neuron and Neural Network
  - Learning
- **Why Quantum Machine Learning**

McCulloch-Pitts (MP) Neuron

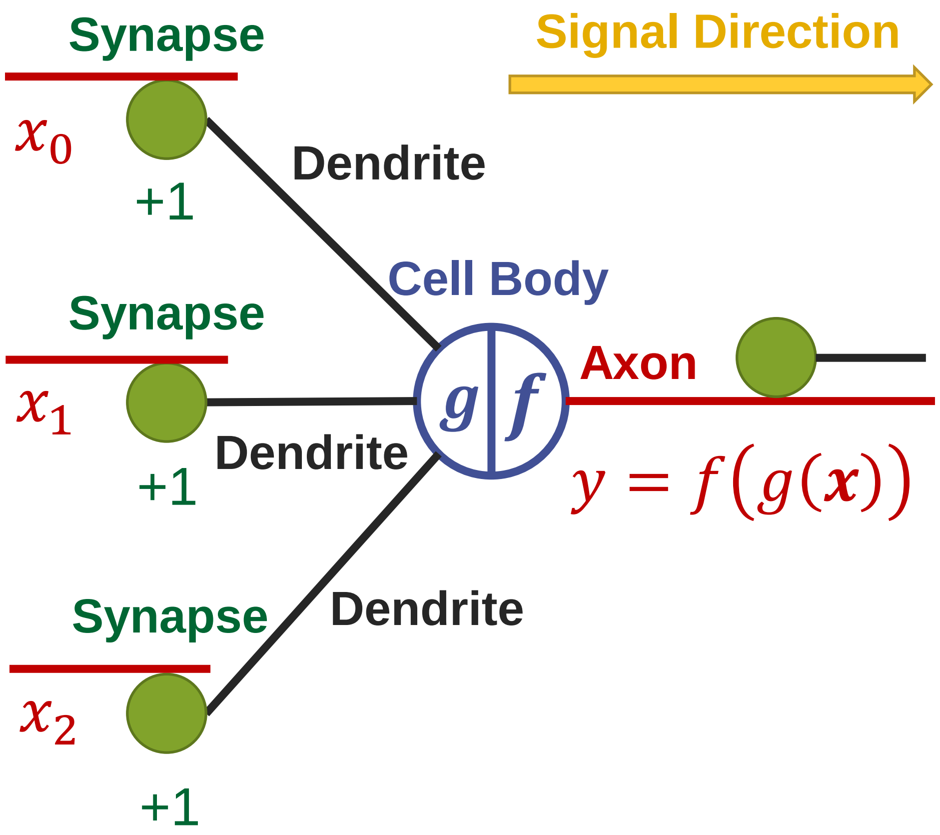
The first computational model of a biological neuron @ 1943



Warren McCulloch

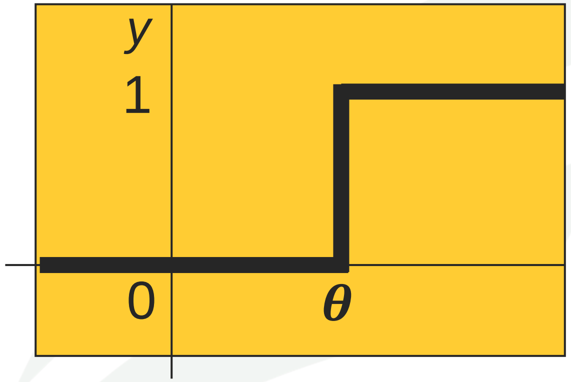


Walter Pitts



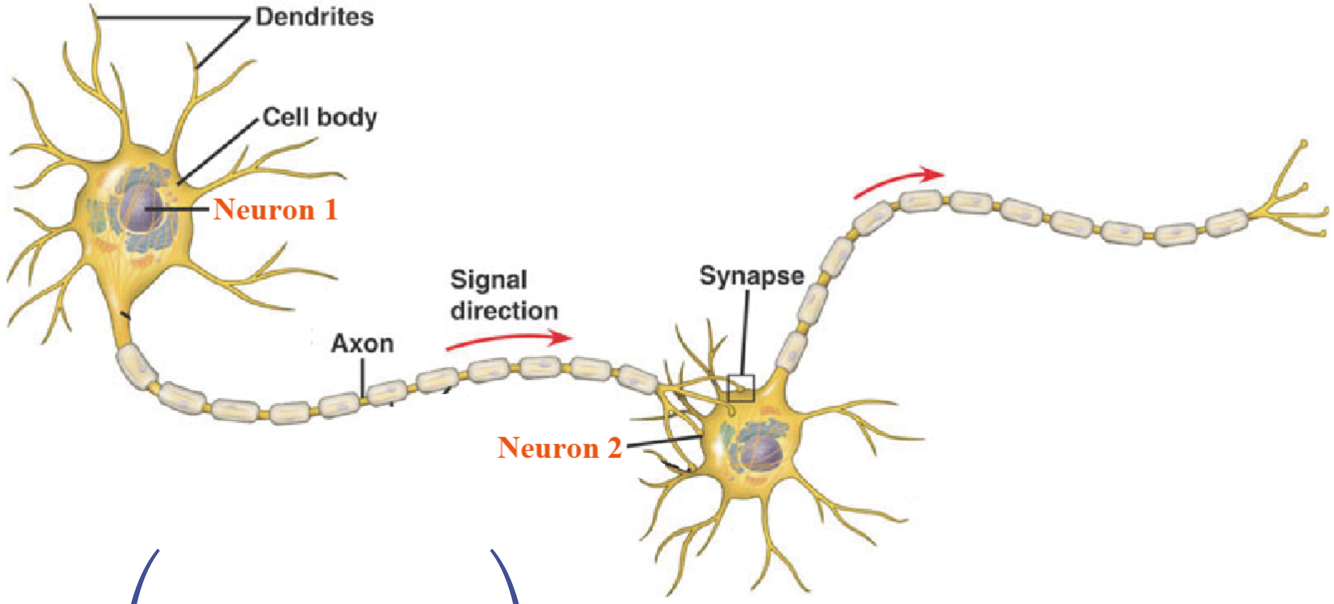
Assumptions:

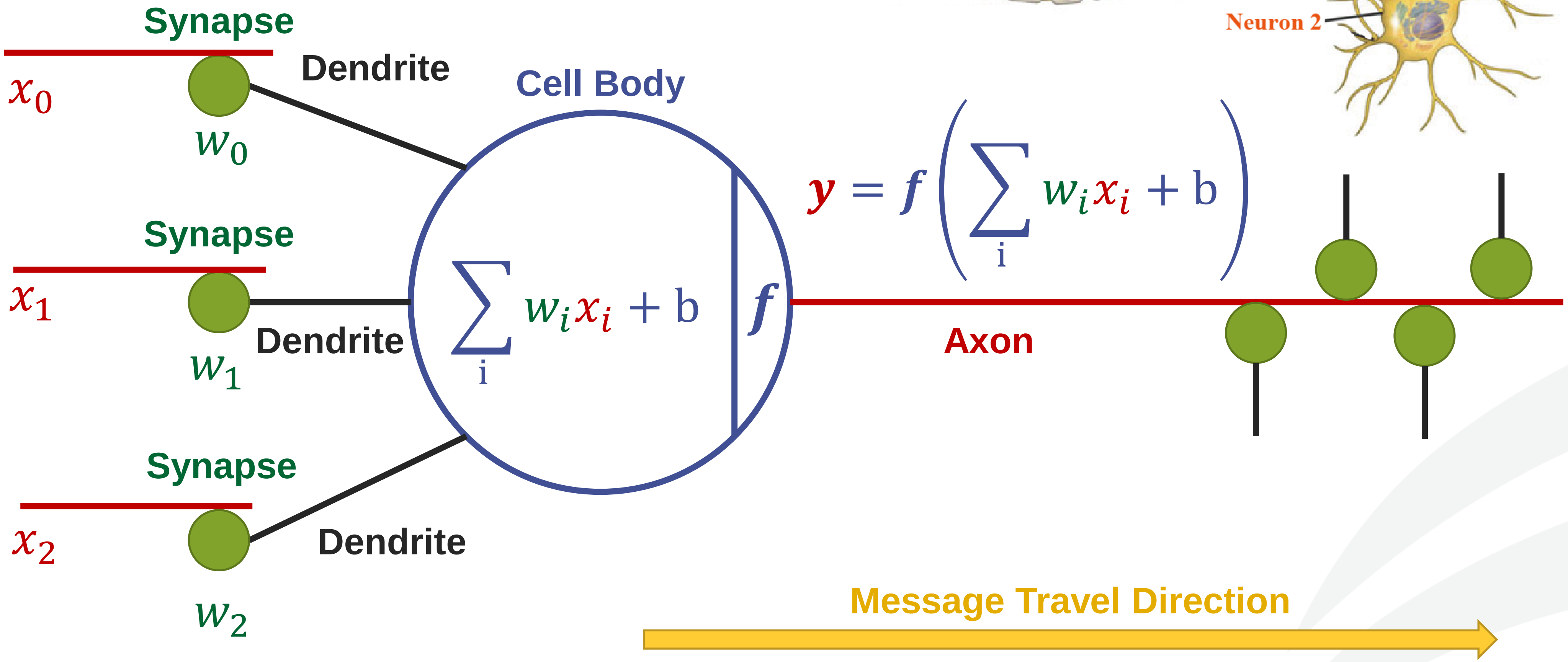
- Binary devices (i.e., $x_i \in \{0,1\}$ and $y \in \{0,1\}$)
- Identical synaptic weights (i.e., +1)
- Activation function $\boldsymbol{f}$ has a fixed threshold $\boldsymbol{\theta}$



Perceptron

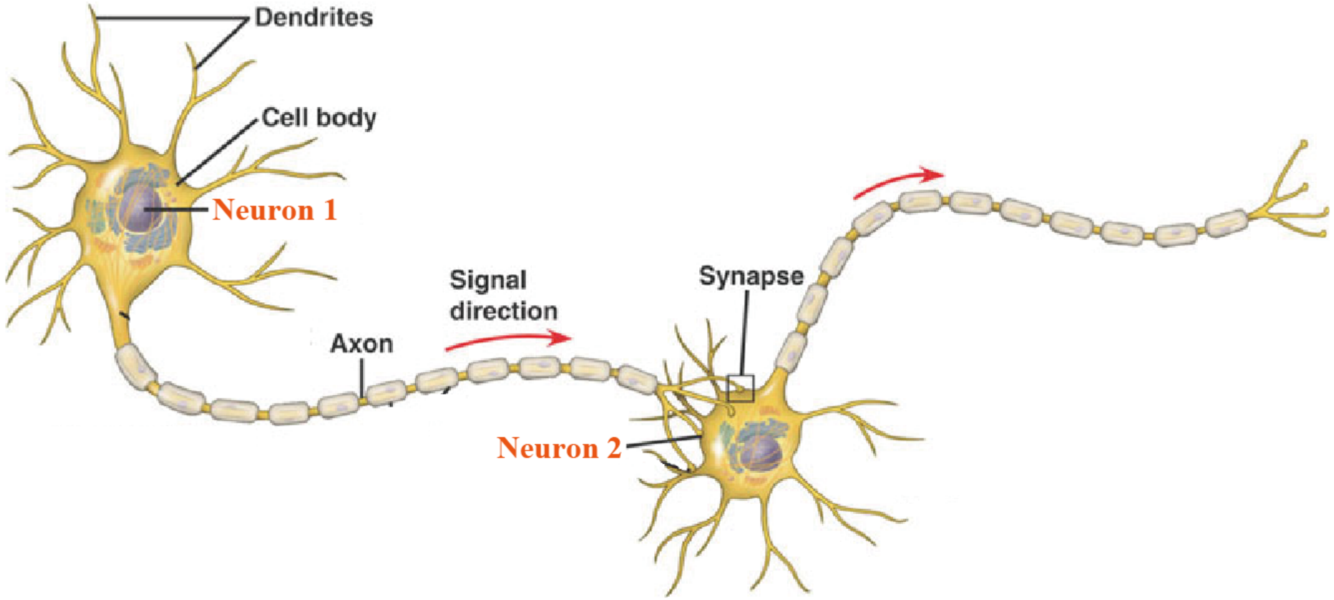
Frank Rosenblatt @ 1958

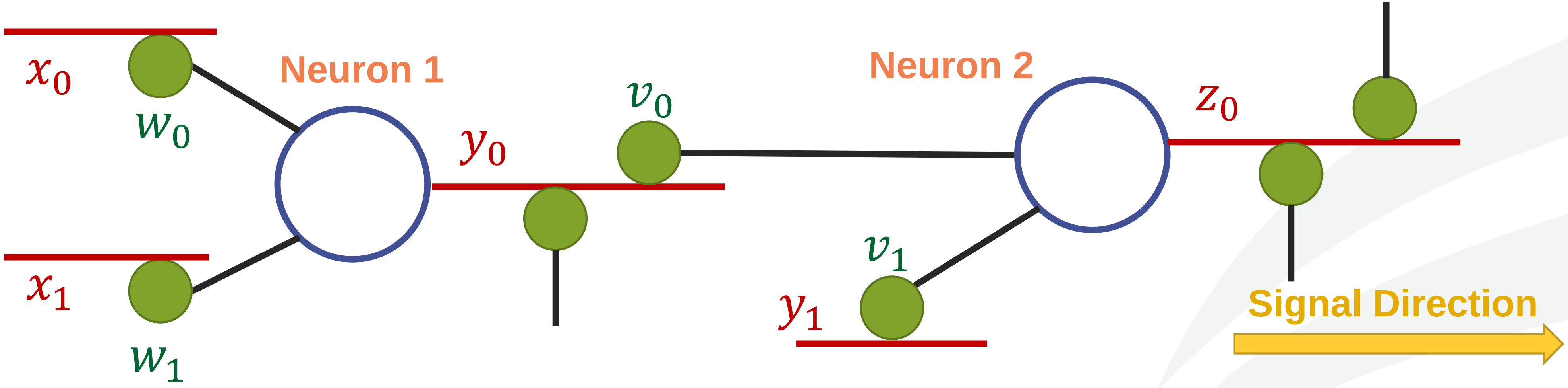




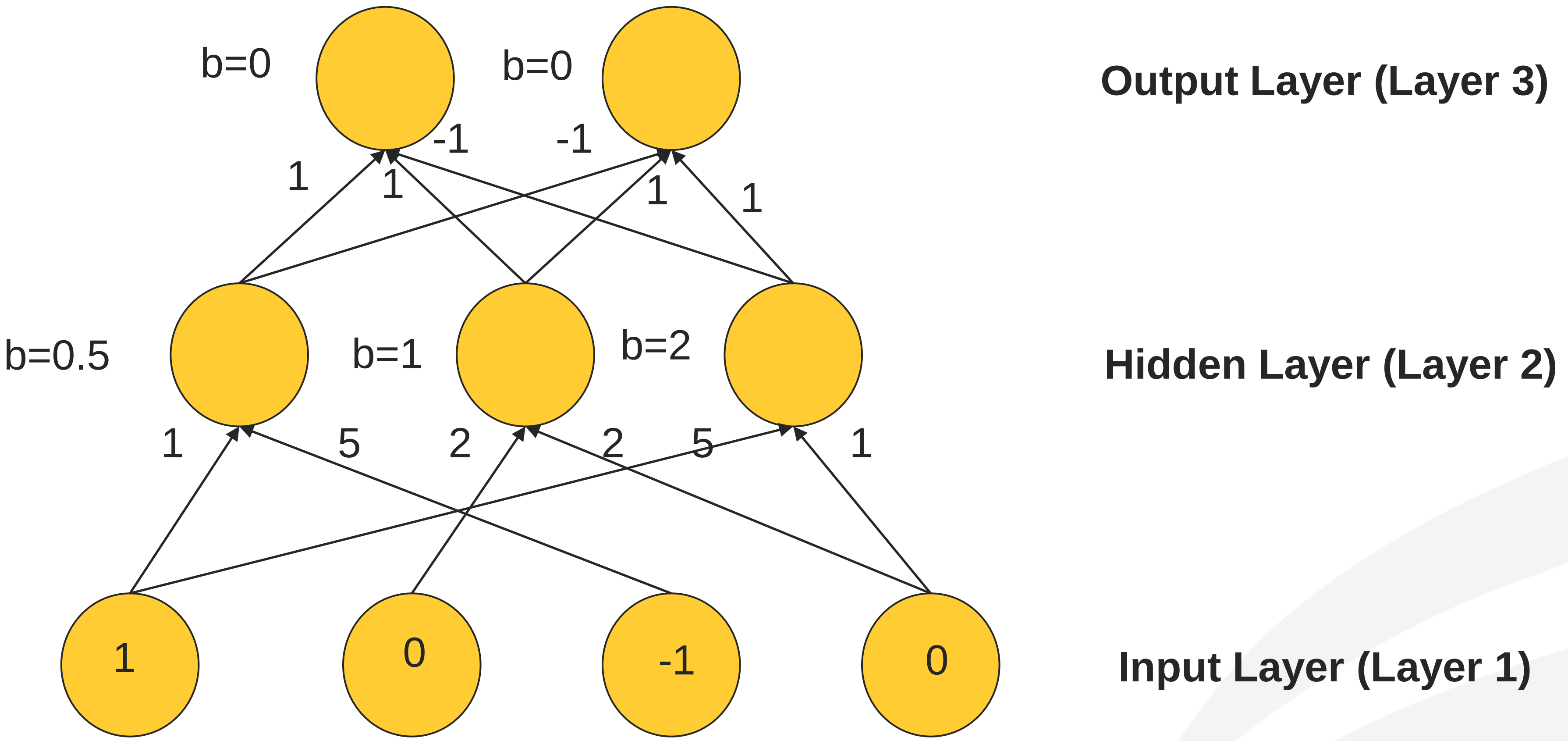
$$y = f\left(\sum_i w_i x_i + b\right)$$

Multi-Layer Perceptron (MLP)
Connect two neurons



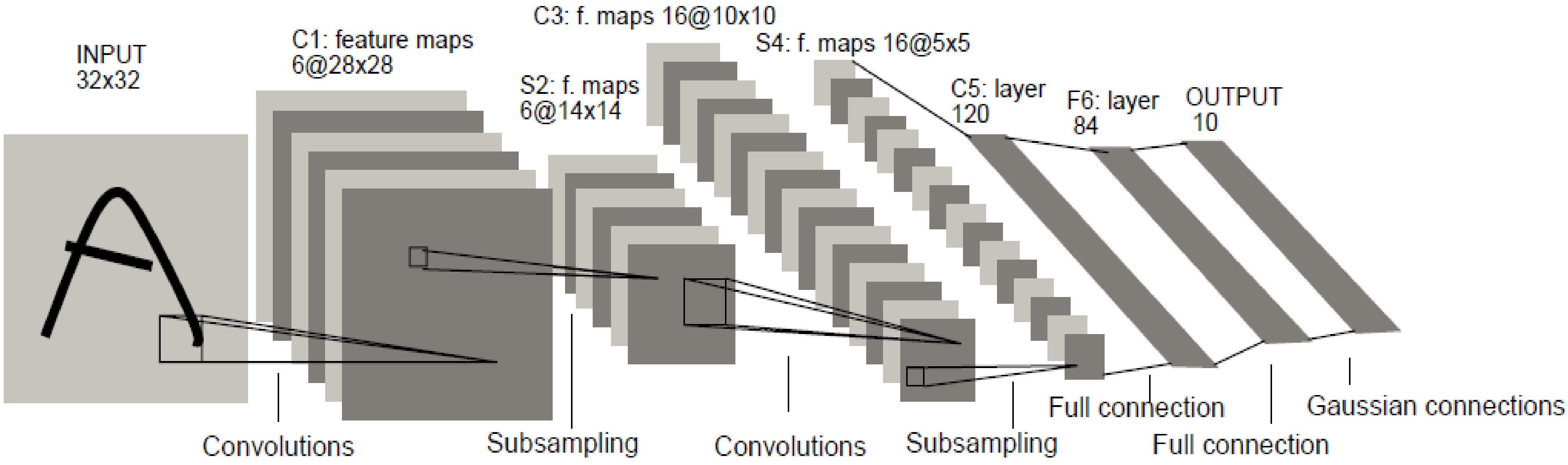


Multi-Layer Perceptron (MLP)
Connect more neurons and more layers



Convolutional Neural Network: LeNet

- The most known CNN for recognizing handwritten digits
  - [LeCun et al., 1998]



Agenda – Session 1: Introduction

- Introduction to Quantum Computing
- **Introduction to Machine Learning**
  - Why Neural Networks
  - Biological Neuron
  - Artificial Neuron and Neural Network
  - Learning
- **Why Quantum Machine Learning**

What is Machine Learning?

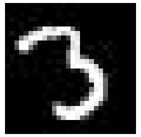
Supervised Learning

Example: Classification

Training

Given: Labeled data as training dataset

(x_i, y_i): x_i training data, y_i: label

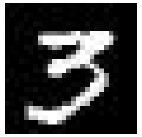
$x_i =$  $\quad y_i = 3$

Output: A learned function $\boldsymbol{f}$ from X to Y

$$\boldsymbol{f}: x \mapsto y$$

Inference/Execution

Given: Unseen data test dataset
A learned function $\boldsymbol{f}$

Do: $\boldsymbol{f}($  $) = 3$

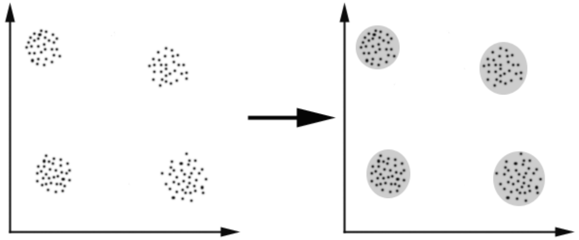
Unsupervised Learning

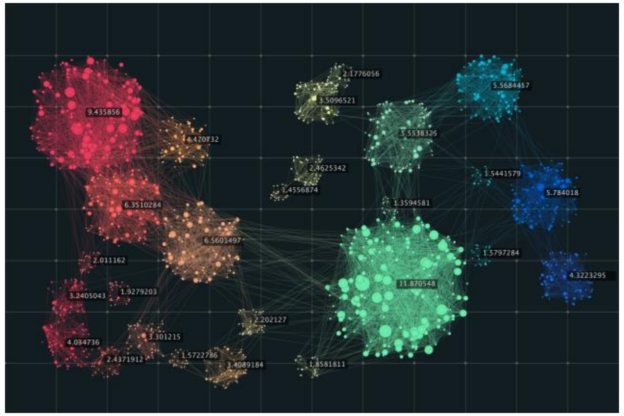
Example: Clustering

Given: Unlabeled data

$$(x_i)$$

Goal: discover the "natural groupings" present in the data



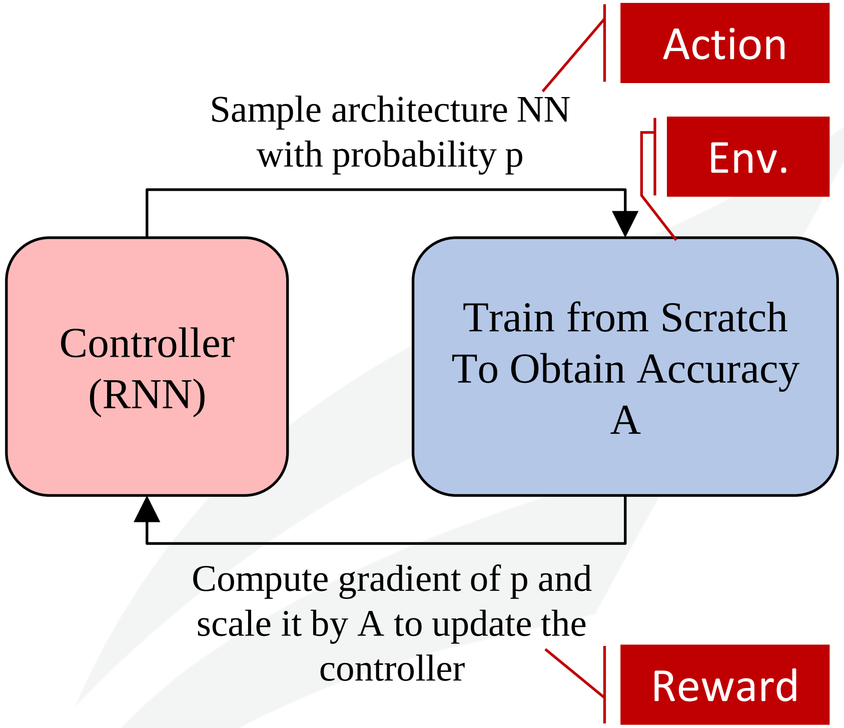


Reinforcement Learning

Example: Neural Architecture Search

Given: An environment that can give us reward based on our action

Goal: Maximize the expected rewards



What is Machine Learning? --- Our Focus

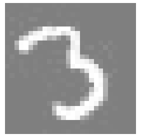
Supervised Learning

Example: Classification

Training

Given: Labeled data as training dataset

(x_i, y_i): x_i training data, y_i: label

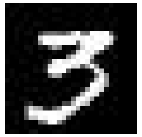
$x_i =$  $y_i = 3$

Output: A learned function f from X to Y

$$f: x \mapsto y$$

Inference/Execution

Given: Unseen data test dataset
A learned function f

Do: $f($  $) = 3$

Unsupervised Learning

Example: Clustering

Given: Unlabeled data

$$(x_i)$$

Goal: discover the "natural groupings" present in the data

Reinforcement Learning

Example: Neural Architecture Search

Given: An environment that can give us reward based on our action

Goal: Maximize the expected rewards

What is Neural Network?

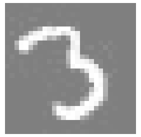
Supervised Learning

Example: Classification

Training

Given: Labeled data as training dataset

(x_i, y_i): x_i training data, y_i: label

$x_i =$  $\quad y_i = 3$

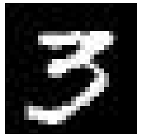
Output: A learned function f from X to Y

$$f: x \mapsto y$$

Inference/Execution

Given: Unseen data test dataset

A learned function f

Do: $f($  $) = 3$

An unknown classification function: $\boldsymbol{g}$

$$\boldsymbol{y = g(x);\ s.t.\ y_i = g(x_i)}$$

Learn a function $\boldsymbol{f}$ with parameters $\boldsymbol{\theta, b}$ to approximate $\boldsymbol{g}$:

$$\boldsymbol{\hat{y} = f(x, \theta, b)}$$

Training is to minimize the loss function by adjusting parameters $\boldsymbol{\theta, b}$

$$\boldsymbol{min:\ \mathcal{L}(f) = \sum_i (f(x_i, \theta, b) - y_i)}$$

Perceptron model, where $\boldsymbol{\sigma}$ is a non-linear function:

$$\boldsymbol{\hat{y} = \sigma(\theta x + b)}$$

Feedforward neural network:

$$\boldsymbol{l_1 = \sigma_1(\theta_1 x + b_1)}$$
$$\boldsymbol{l_2 = \sigma_2(\theta_2 l_1 + b_2)}$$
$$\dots\dots$$
$$\boldsymbol{l_n = \sigma_n(\theta_n l_{n-1} + b_n)}$$
$$\boldsymbol{\hat{y} = classifier(l_n)}$$

What is Neural Network?

Supervised Learning

Example: Classification

Training

Given: Labeled data as training dataset

(x_i, y_i): x_i training data, y_i: label

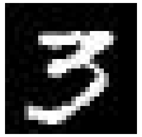
$x_i =$ $y_i = 3$

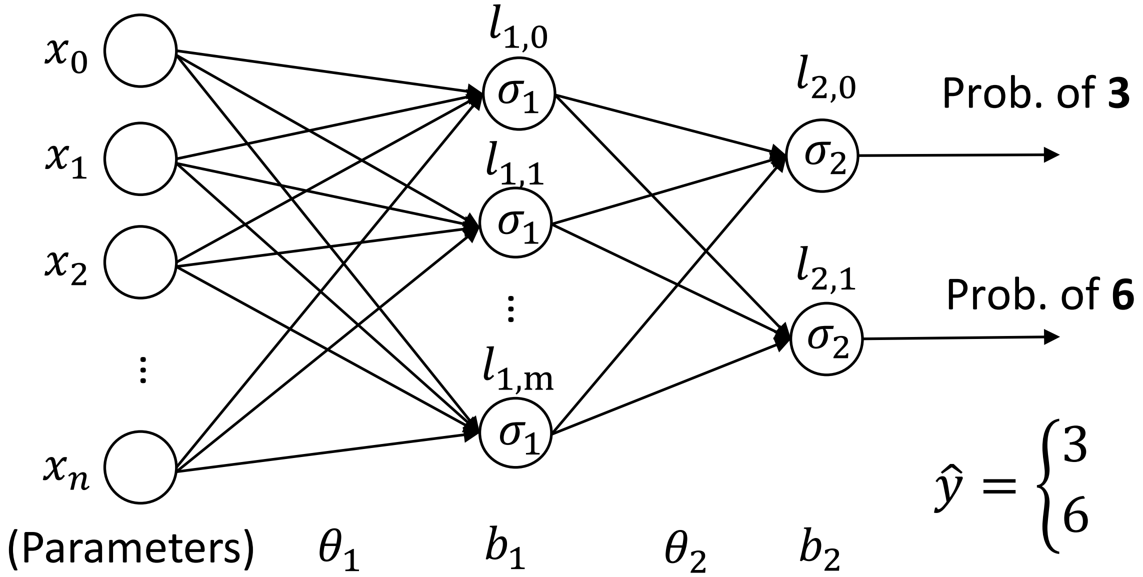
Output: A learned function f from X to Y

$$f: x \mapsto y$$

Inference/Execution

Given: Unseen data test dataset
A learned function f

Do: $f($  $) = 3$



$$\hat{y} = \begin{cases} 3 & l_{2,0} > l_{2,1} \\ 6 & l_{2,0} \leq l_{2,1} \end{cases}$$

Example of feedforward neural network for $n = 2$

Perceptron model, where $\boldsymbol{\sigma}$ is a non-linear function:

$$\hat{\boldsymbol{y}} = \boldsymbol{\sigma}(\boldsymbol{\theta x} + \boldsymbol{b})$$

Feedforward neural network:

$$\boldsymbol{l_1} = \boldsymbol{\sigma_1}(\boldsymbol{\theta_1 x} + \boldsymbol{b_1})$$
$$\boldsymbol{l_2} = \boldsymbol{\sigma_2}(\boldsymbol{\theta_2 l_1} + \boldsymbol{b_2})$$
$$\dots\dots$$
$$\boldsymbol{l_n} = \boldsymbol{\sigma_n}(\boldsymbol{\theta_n l_{n-1}} + \boldsymbol{b_n})$$
$$\hat{\boldsymbol{y}} = \boldsymbol{classifier}(\boldsymbol{l_n})$$

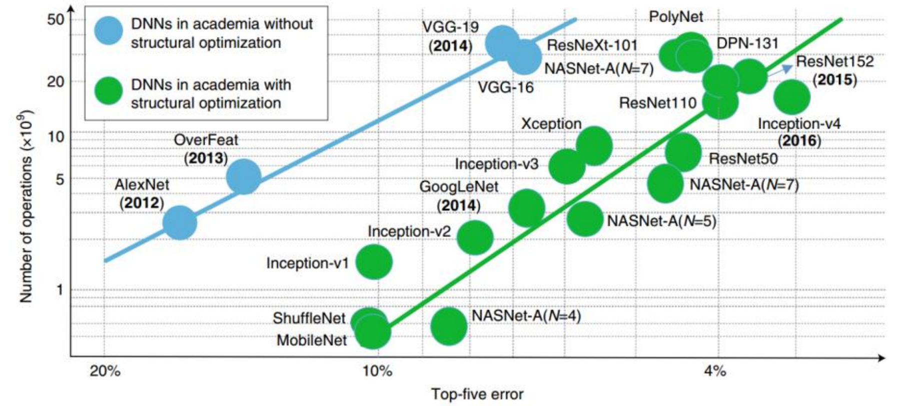
Agenda – Session 1: Introduction

- Introduction to Quantum Computing
- Introduction to Machine Learning
- **Why Quantum Machine Learning**

Why Using Quantum Computer for Machine Learning?

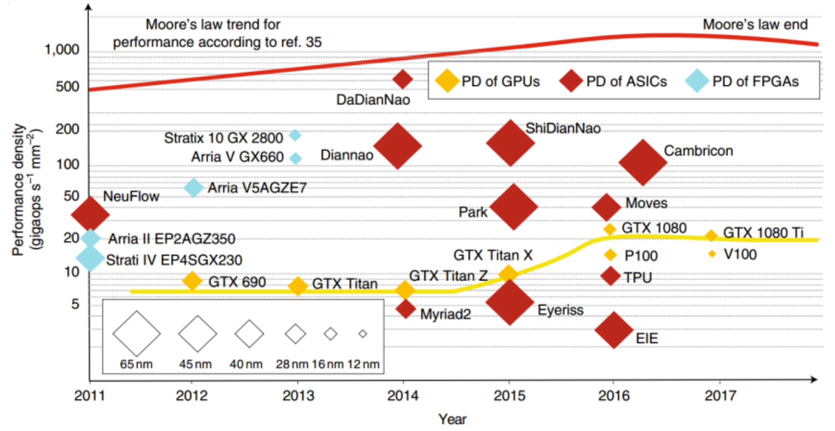
- Imbalanced “demand and supply” of NN on classical computing
- The growing power of quantum computing
- Linear algebra is central for both quantum computing and machine learning

NN on Classical Computer: Computation & Storage Demand > Supply



Number of DNN operations increases exponentially

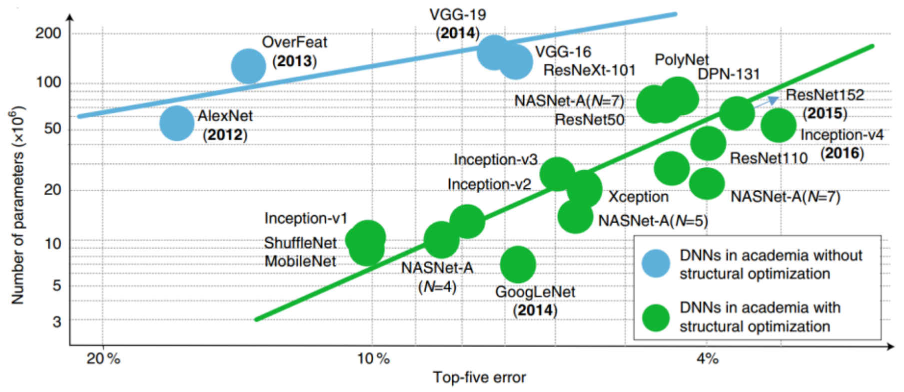
Computation Gap



Performance density almost stops increasing

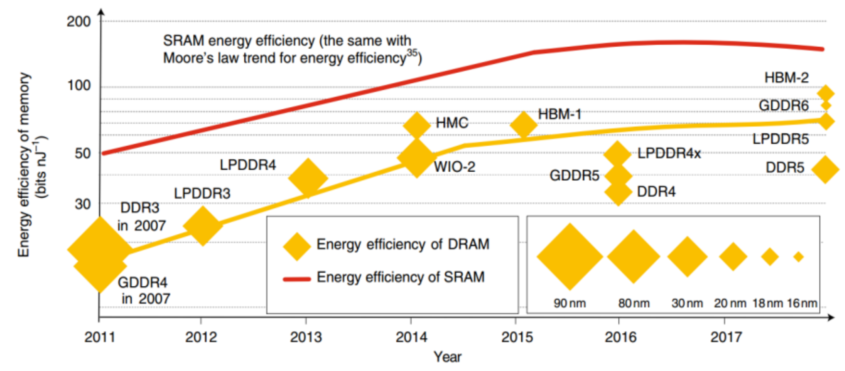
Neural Network Size

Traditional Hardware Capability



Number of DNN parameters increases exponentially

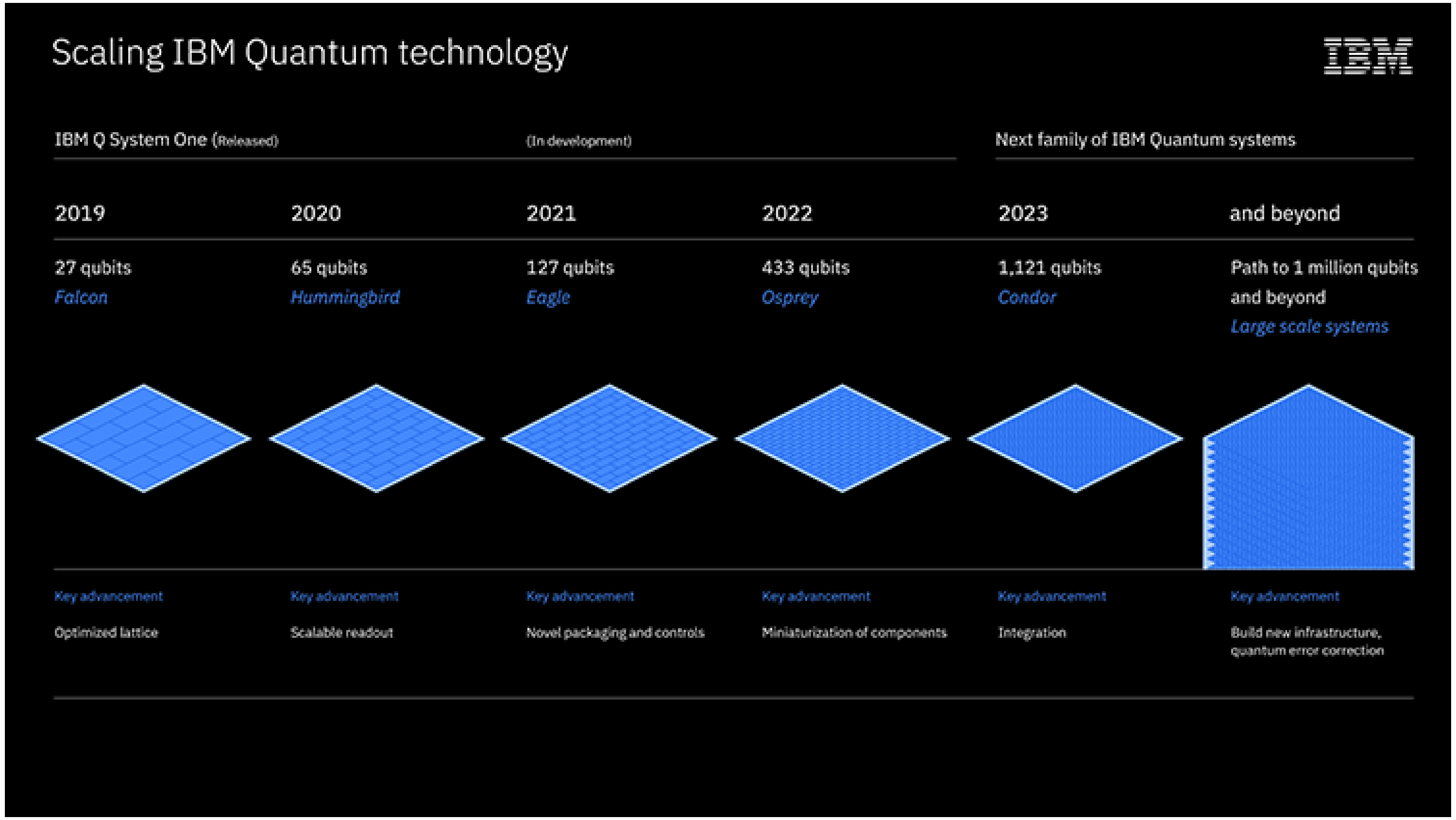
Storage Gap

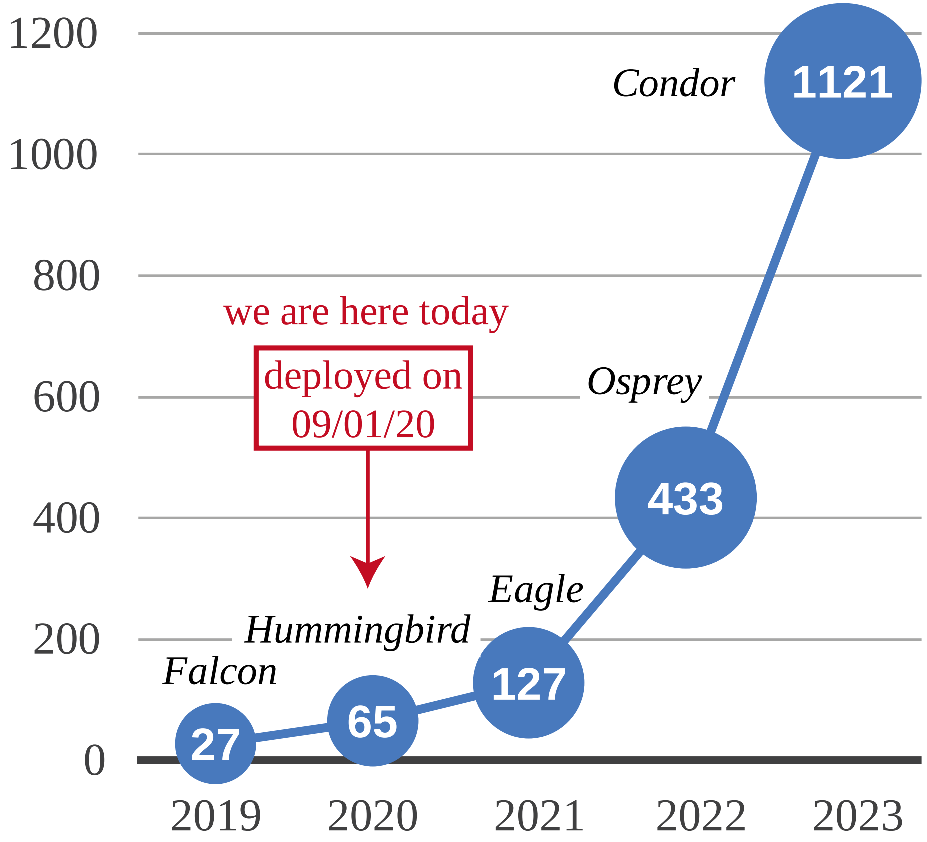


Energy efficiency of memory stops increasing

[ref] Xu, X., et al. 2018. Scaling for edge inference of deep neural networks. *Nature Electronics*, *1*(4), pp.216-222.

Consistently Increasing Qubits in Quantum Computers





Linear Algebra is Central for Quantum Computing

$$A_{N,N} \times B_{N,1} = \frac{1}{2} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \times \begin{bmatrix} c_{00} \\ c_{01} \\ c_{10} \\ c_{11} \end{bmatrix} = \begin{bmatrix} d_{00} \\ d_{01} \\ d_{10} \\ d_{11} \end{bmatrix}$$

$$|q_0, q_1\rangle = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle$$

$$\rightarrow \begin{bmatrix} c_{00} \\ c_{01} \\ c_{10} \\ c_{11} \end{bmatrix}$$ (vector representation)

$$H \otimes H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = A_{N,N}$$

$$H \otimes H|q_0, q_1\rangle$$

$$= d_{00}|00\rangle + d_{01}|01\rangle + d_{10}|10\rangle + d_{11}|11\rangle$$

Matrix multiplication on classical computer using 16bit number

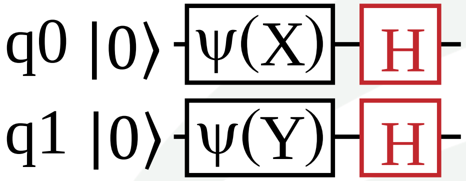
$$A_{N,N} \times B_{N,1} = C_{N,1}$$

Data: $(M \times M + 2 \times M) \times 16bit, \quad M = 2^2$

Operation: Multiplication: $\boldsymbol{M \times M}$

Accumulation: $\boldsymbol{M \times (M - 1)}$

Special matrix multiplication on quantum computer



Data: **K** Qbits, **K = logM = 2**

Operation: **K** Hadamard (H) Gates

Takeaway

- Quantum Computing
  - # of qubits grows rapidly
  - Q-Circuit design is similar to classical ones, using quantum gates
- Machine Learning meets Quantum Computing
  - Potential to solve computation-bound / memory-wall in classical
  - What is quantum neural network? VQC v.s. Q-Based Accelerator

wjiang8@gmu.edu

George Mason University

4400 University Drive
Fairfax, Virginia 22030

Tel: (703)993-1000