Deep learning with quantum neural networks

Trial lecture by Vegard Børve Sørdal

Advisor: Joakim Bergli

Co-advisors: Yuri Galperin and Luiza Angheluta

Opponents: Vitaly Shumeiko and David Sánchez

Neural networks have been proven very successful for solving complex high dimensional problems

Quantum mechanics is all about manipulating vectors in a high dimensional space, with added exotic effects

Can we combine them?

Neural networks

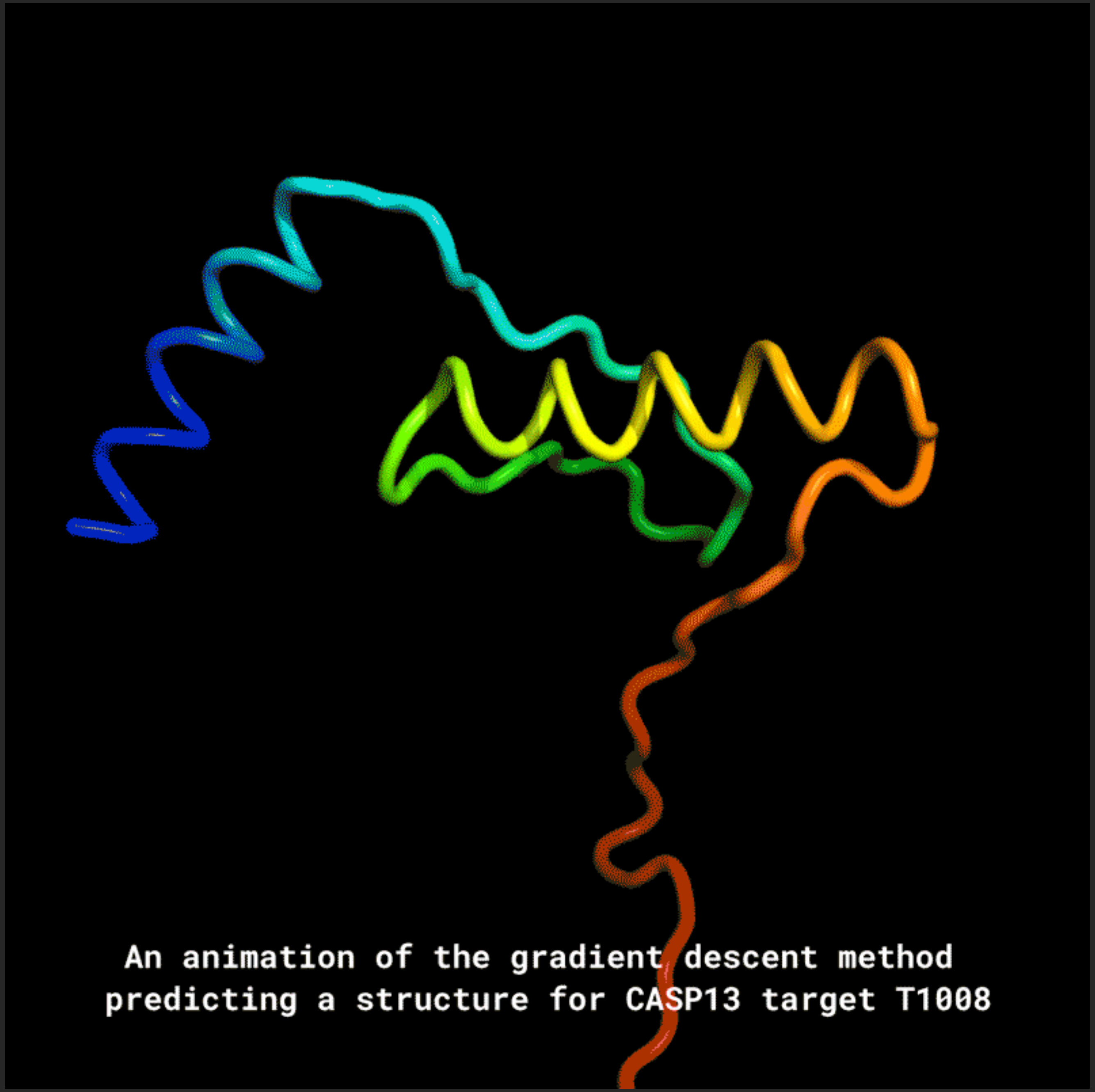
Neural networks is the core of the recent AI boom

If you want the NN to preform some task, it needs relevant data to learn from

Self driving cars

Data: interaction with simulated environment

Protein folding



Data: folded structure of known proteins

Generate new faces?

Figure 5: 1024 × 1024 images generated using the CELEBA-HQ dataset. See Appendix F for a larger set of results, and the accompanying video for latent space interpolations.

Data: pictures of celebrity faces

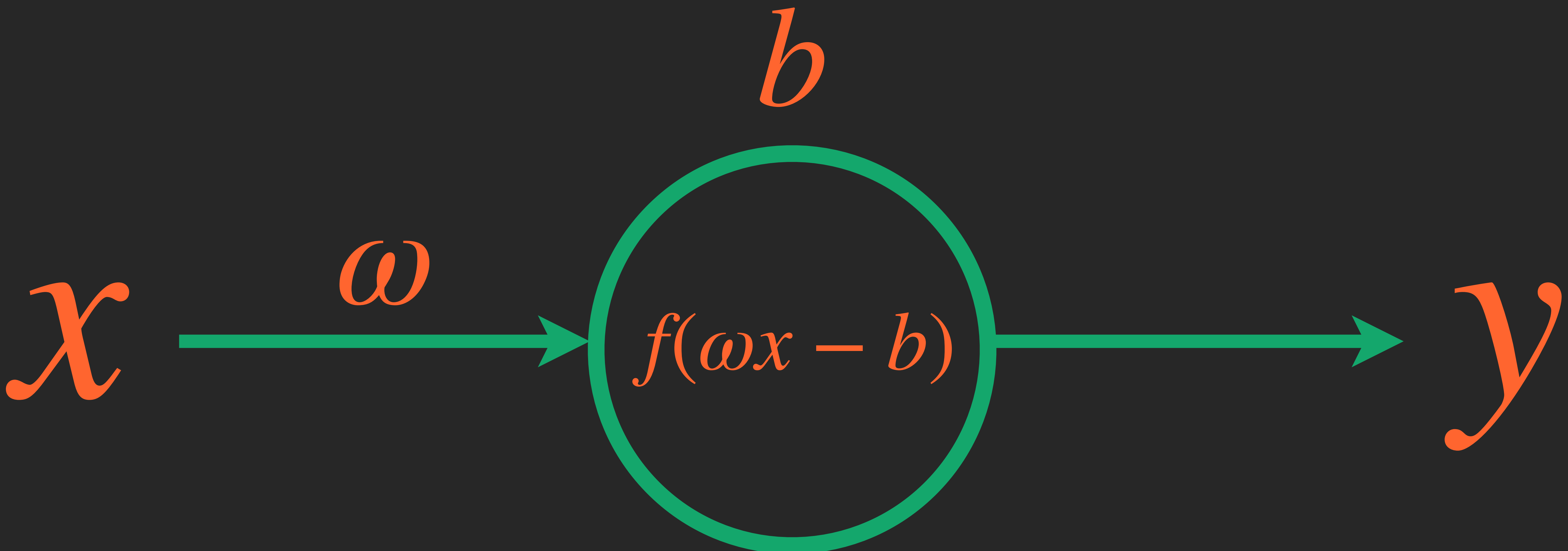
Generate new faces?

Data: pictures of celebrity faces

Huge range of other applications

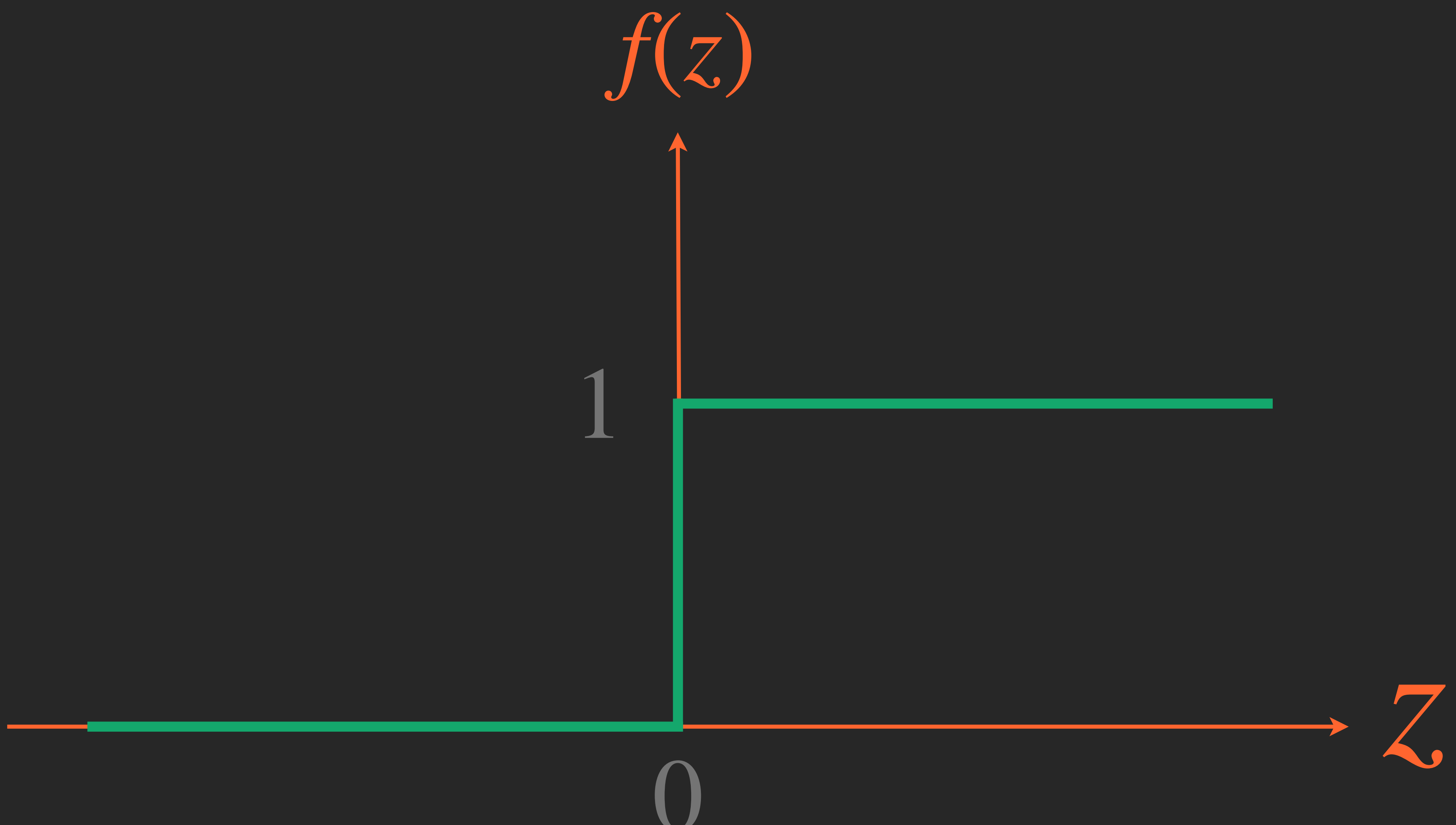
But lets start from scratch and describe the smallest unit of the neural network

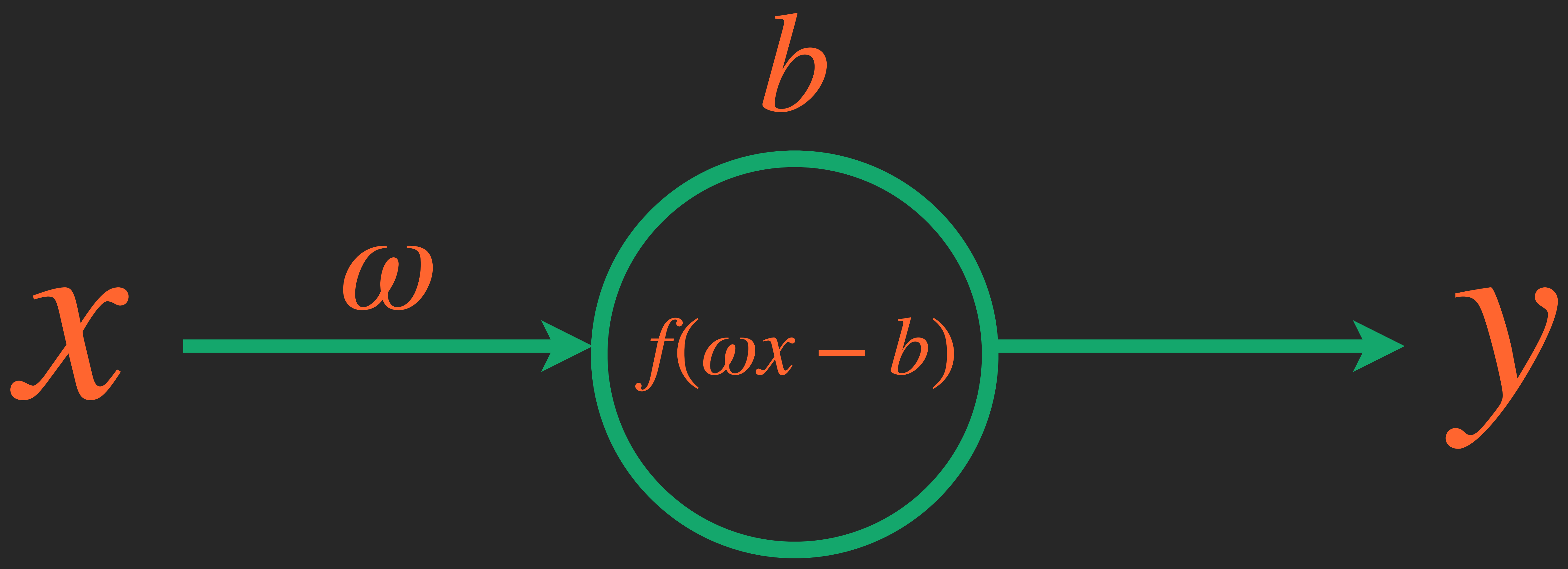
Artificial neuron



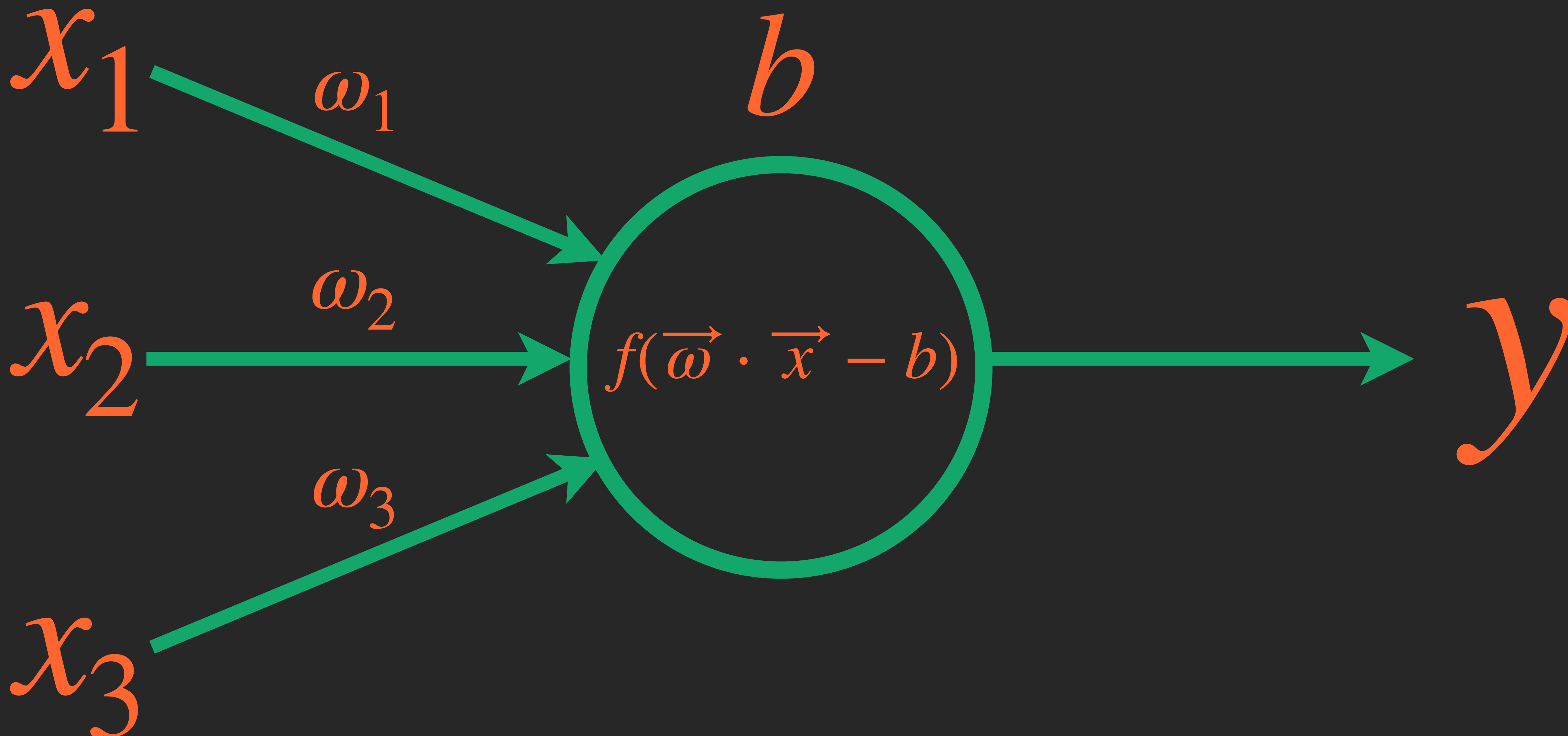
$$y = f(\omega x - b)$$

$$f(z) = \begin{cases} 0 & \textbf{if} \quad z < 0 \\ 1 & \textbf{if} \quad z \geq 0 \end{cases}$$

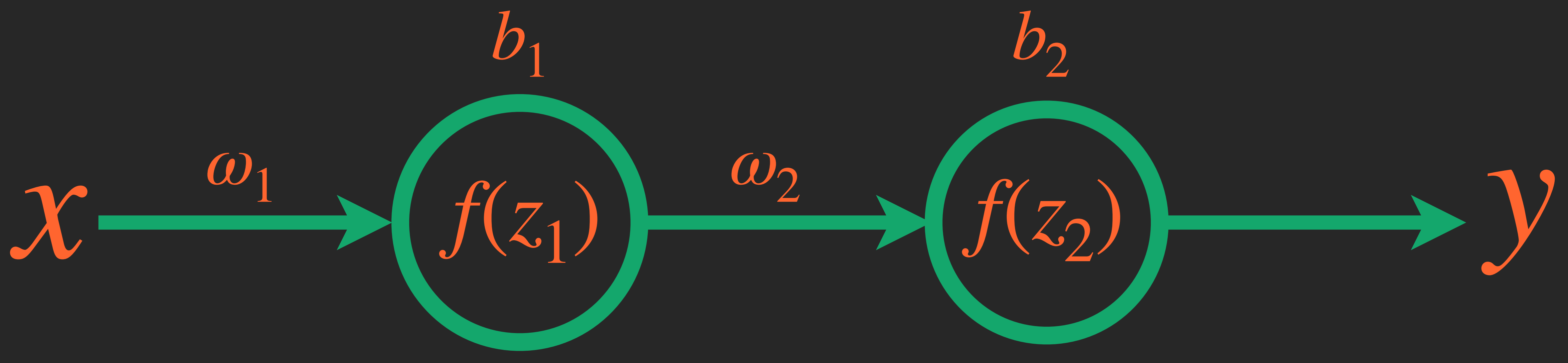




$$y = \begin{cases} 0 & \textbf{if} \quad \omega x < b \\ 1 & \textbf{if} \quad \omega x \geq b \end{cases}$$



$$\vec{\omega} \cdot \vec{x} = \sum_{i=1}^{3} \omega_i x_i$$



The output from the first neuron
is the input to the second neuron

A non-linear function can be used as activation function

$$f(z) = \frac{1}{1 + e^{-z}}$$

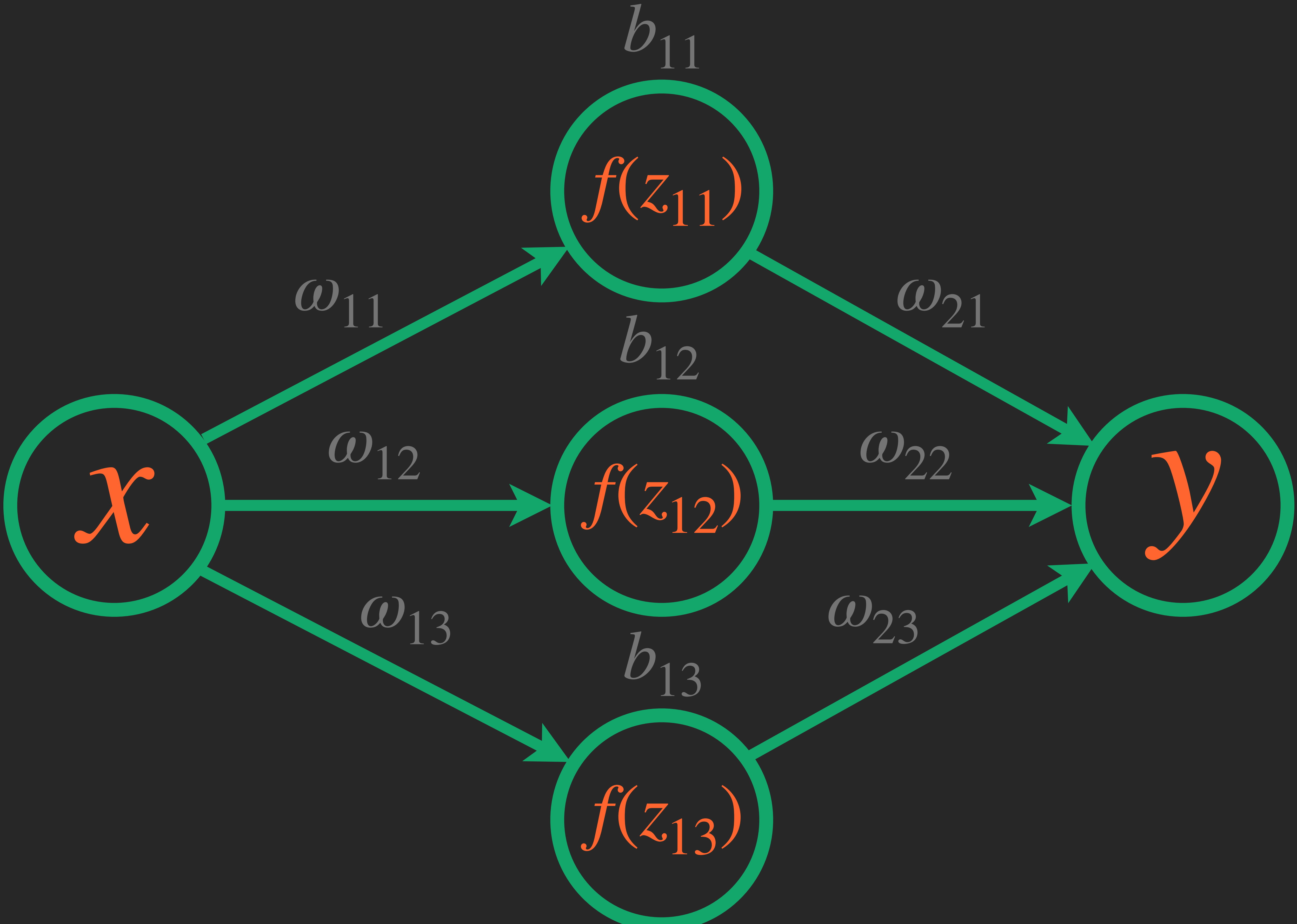
$$f(z) = \tanh(z)$$

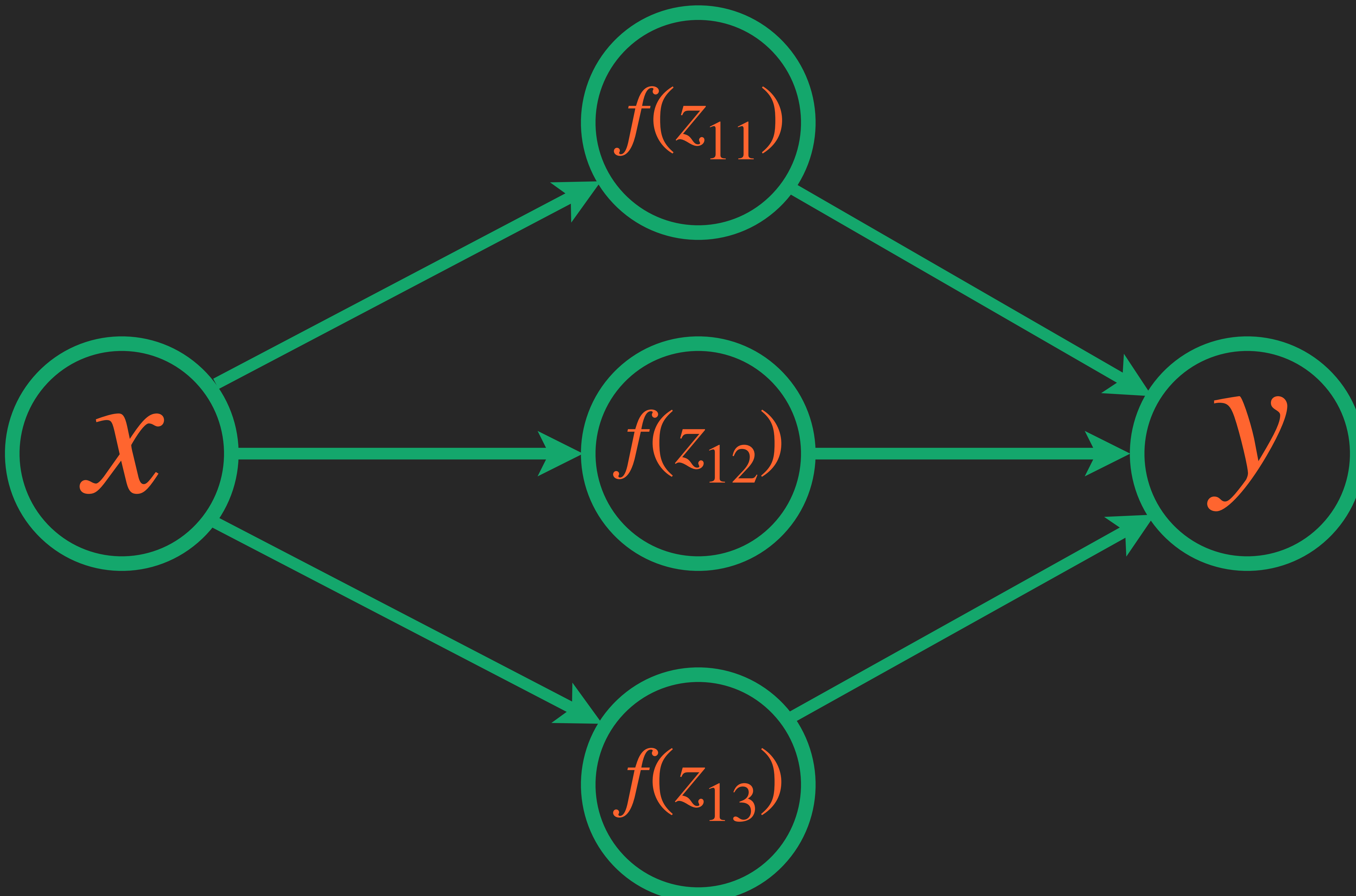
$$f(z) = \begin{cases} 0 & \textbf{if} \quad z < 1 \\ z & \textbf{if} \quad z \geq 1 \end{cases}$$

Why non-linear activation?

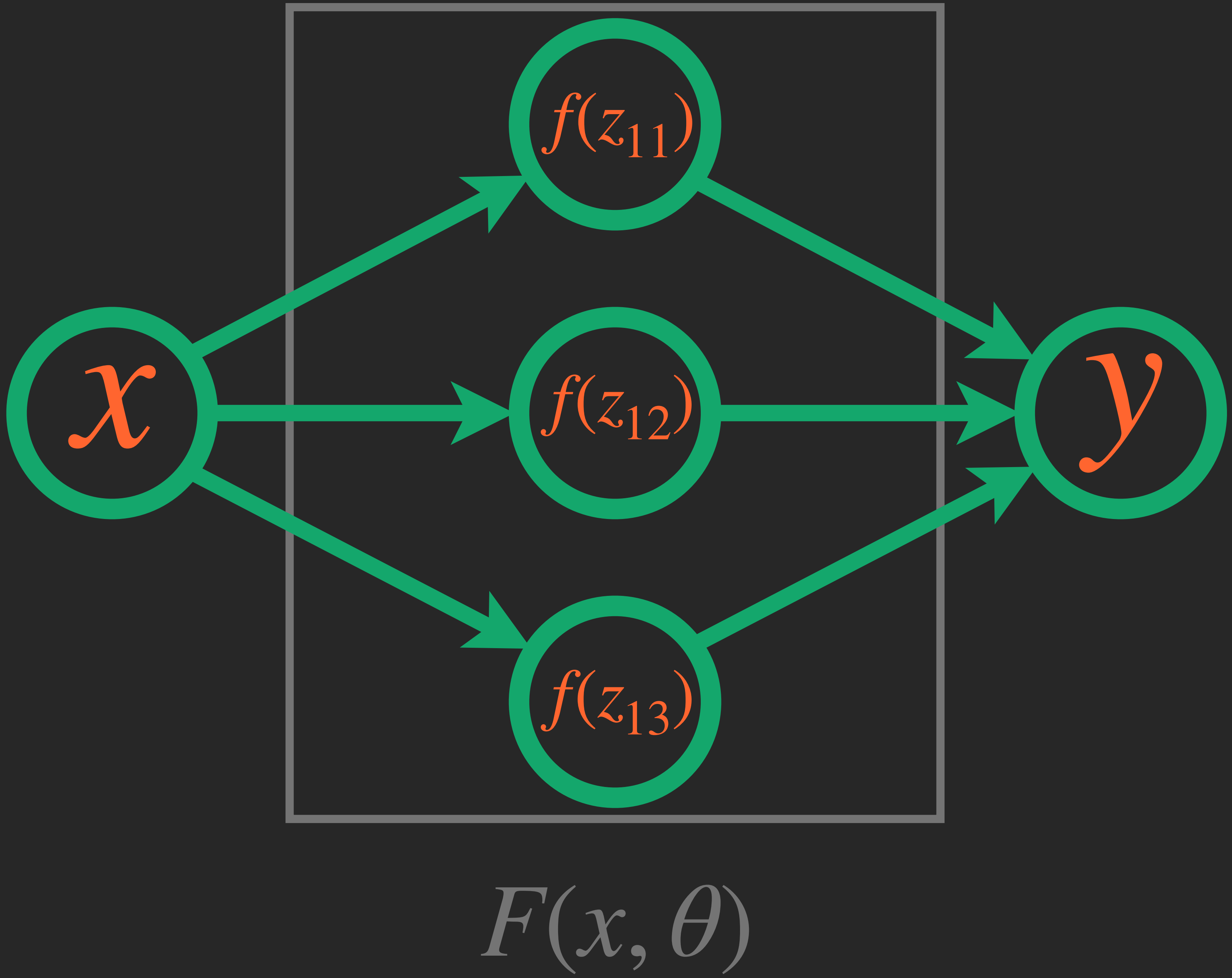
$$f(x) = x$$

$$\begin{aligned} y &= f(z_2) \\ &= f(\omega_2 f(z_1) + b_2) \\ &= f(\omega_2(\omega_1 x + b_1) + b_2) \\ &= \omega_2\omega_1 x + \omega_2 b_1 + b_2 \end{aligned}$$


b_{11}
$f(z_{11})$
ω_{11}
ω_{21}
b_{12}
x
ω_{12}
$f(z_{12})$
ω_{22}
y
ω_{13}
ω_{23}
b_{13}
$f(z_{13})$



$$\theta = \{\omega_{11}, \omega_{12}, \omega_{13}, b_{11}, b_{12}, b_{13}, \omega_{21}, \omega_{22}, \omega_{23}\}$$

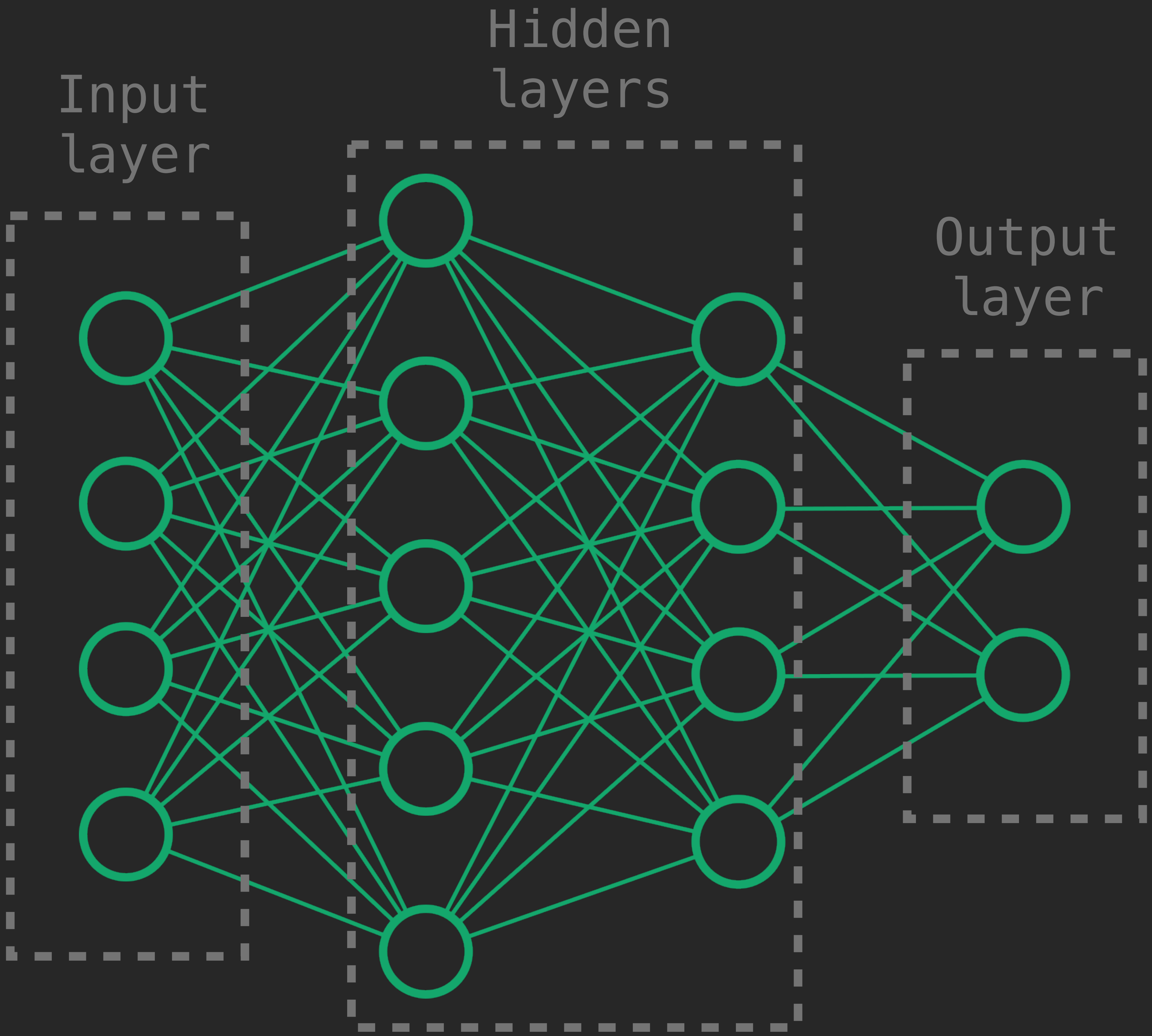


$G : x \rightarrow y$

$F(x, \theta) \simeq G(x)$

A single finite layer can approximate
<u>any</u> continuous function

Deep Neural Network



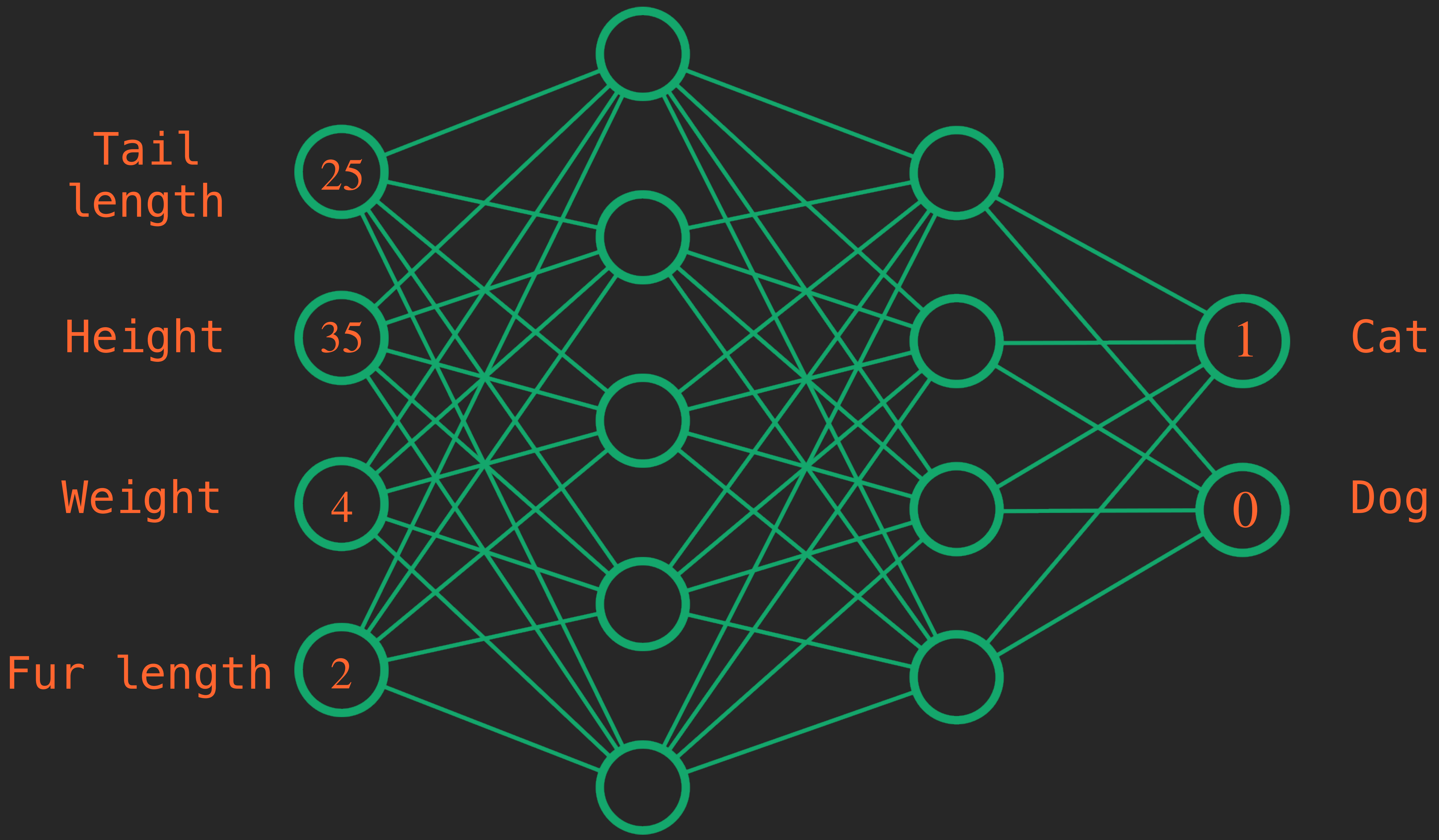
Deep Neural Network

Can be used as function approximators for very abstract functions.

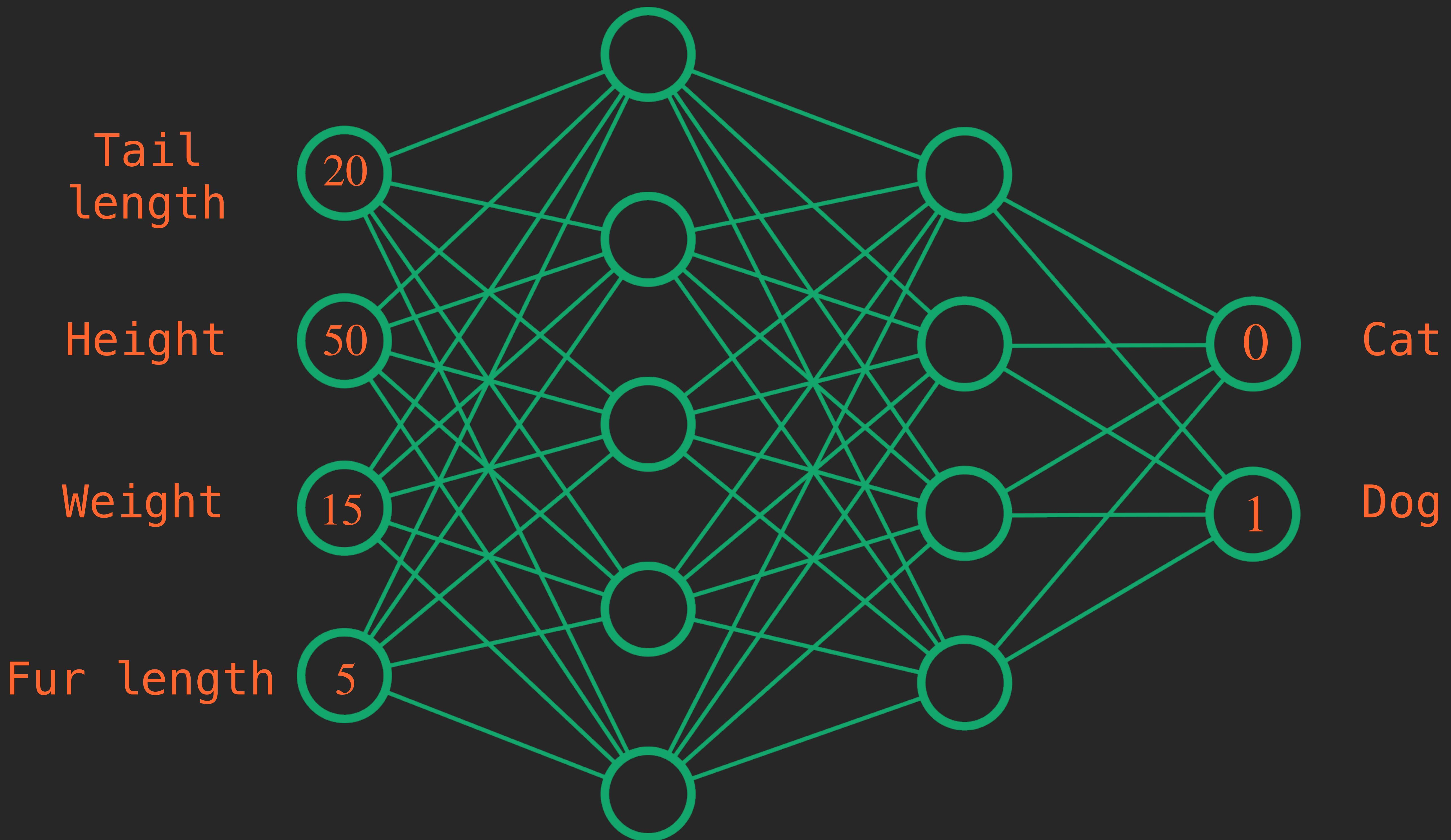
For example functions that predicts whether data comes from measurement on a cat or a dog.

	Tail length (cm)	Height (cm)	Weight (kg)	Fur length (cm)
Dog 1	30	60	15	5
Cat 1	15	20	5	3
Cat 2	20	30	6	8
Dog 1	5	80	40	6
...				

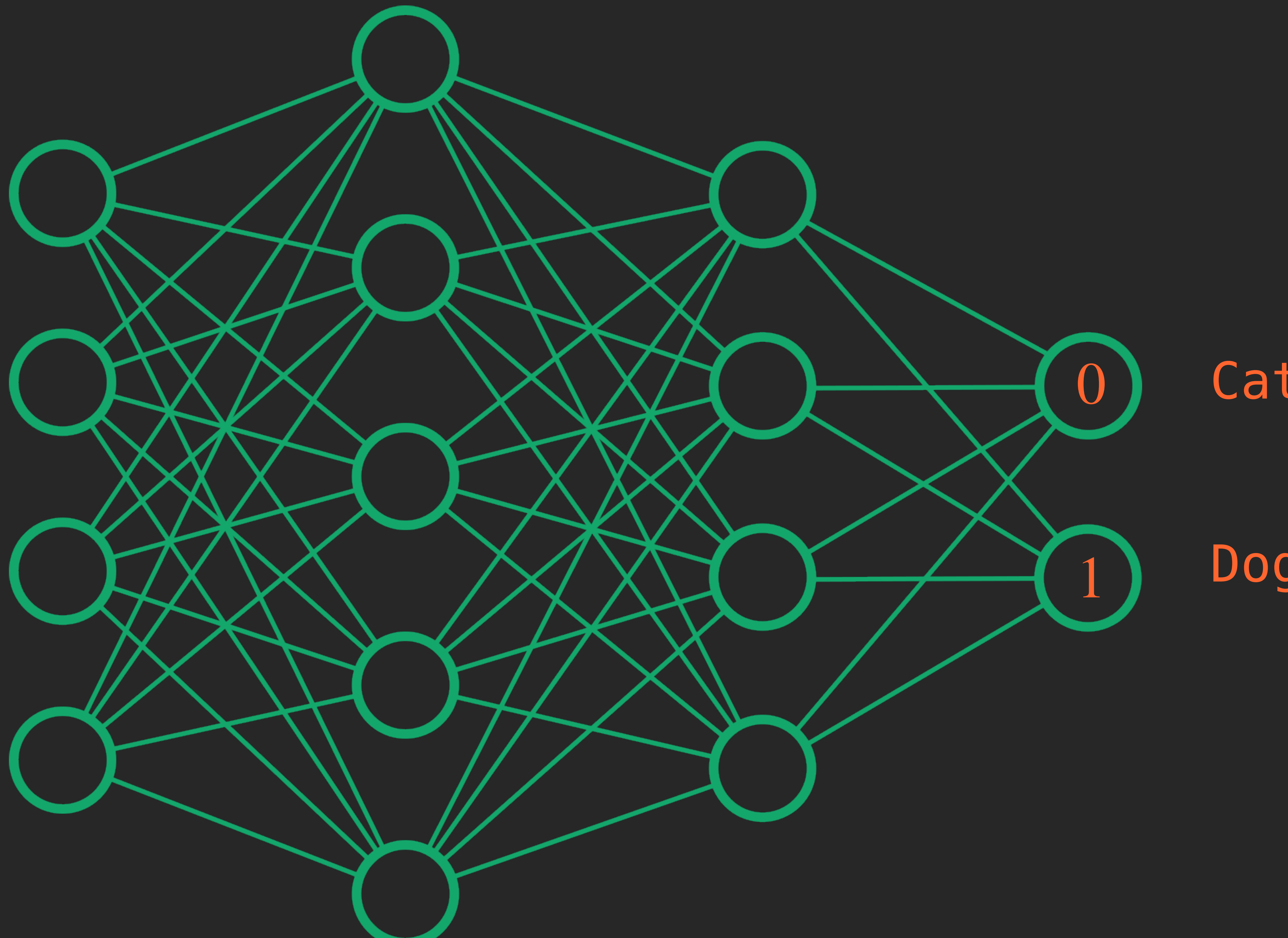
Classification example



Classification example

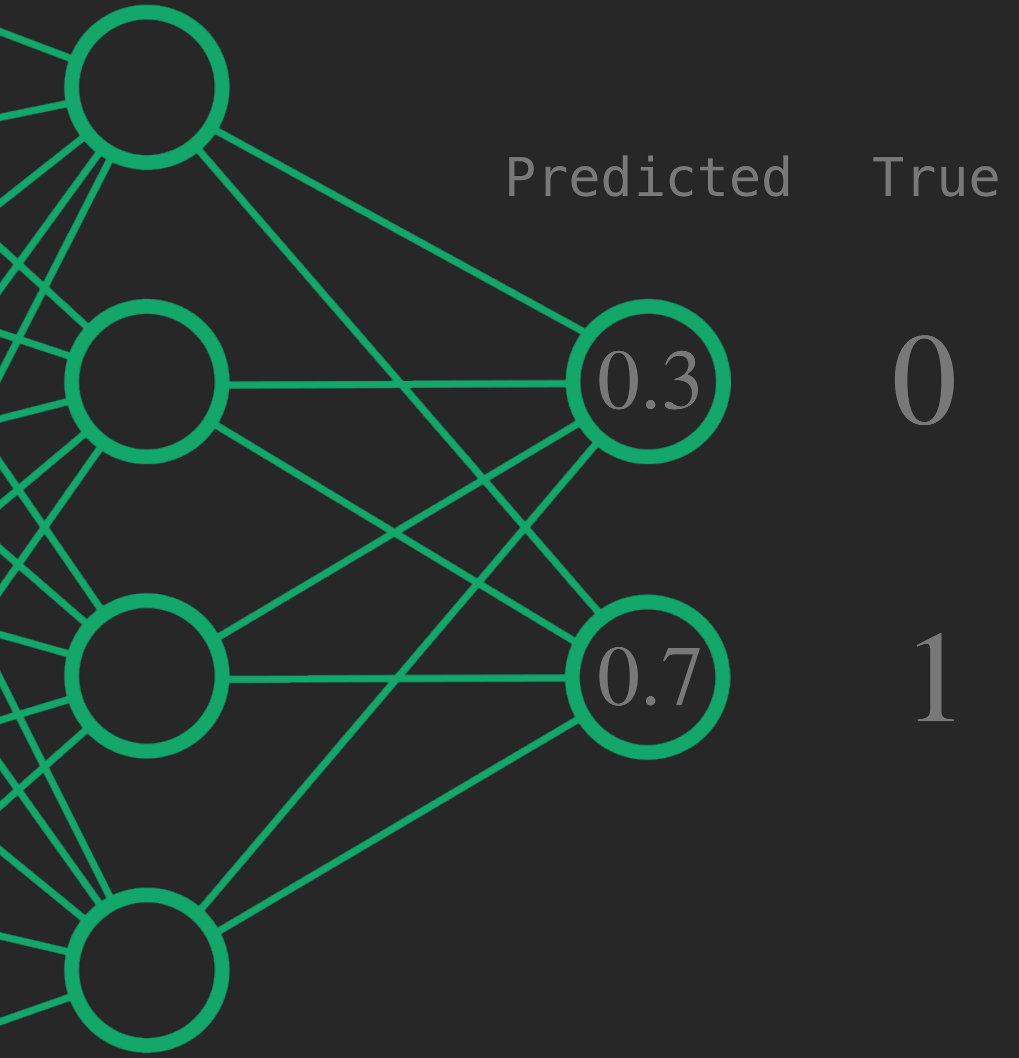


Classification example



$32 \times 32 \rightarrow 1024$ input neurons

Have to tweak θ based on data, so that the network can make accurate predictions



$$\vec{a}_L = \begin{bmatrix} 0.3 \\ 0.7 \end{bmatrix} \qquad \vec{y}_{target} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Some measure of error

$$C(\theta) = \sum_x \| \vec{a}_L(x) - \vec{y}_{target}(x) \|^2$$

Cost function summing over all data points x

Minimized when

$$\frac{\partial C(\theta)}{\partial \theta} = 0$$

Gradient descent

$$\theta_{n+1} = \theta_n - \eta \frac{\partial C}{\partial \theta}$$

Neural network summary

Powerful universal function approximators

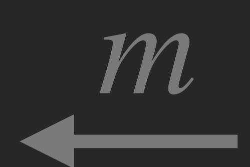
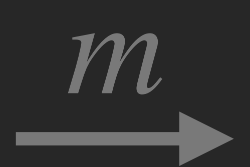
Non-linear activation of nested functions gives deep representative power

Have efficient weight updating methods (back-propagation)

How good the function approximation is depends on how much data you have to train on

Quantum Computing

Classical bit

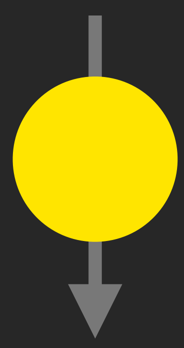
	No		Yes
	0	OR	1
CD	No mirror		Mirror
Hard disk			

Quantum bit

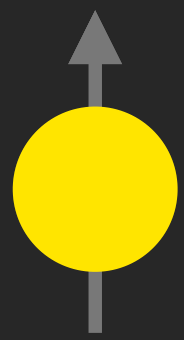
No — Yes

0 ~~AND,OR,BOTH~~ 1

Quantum bit: qubit



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



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

$$|\alpha|^2 + |\beta|^2 = 1$$

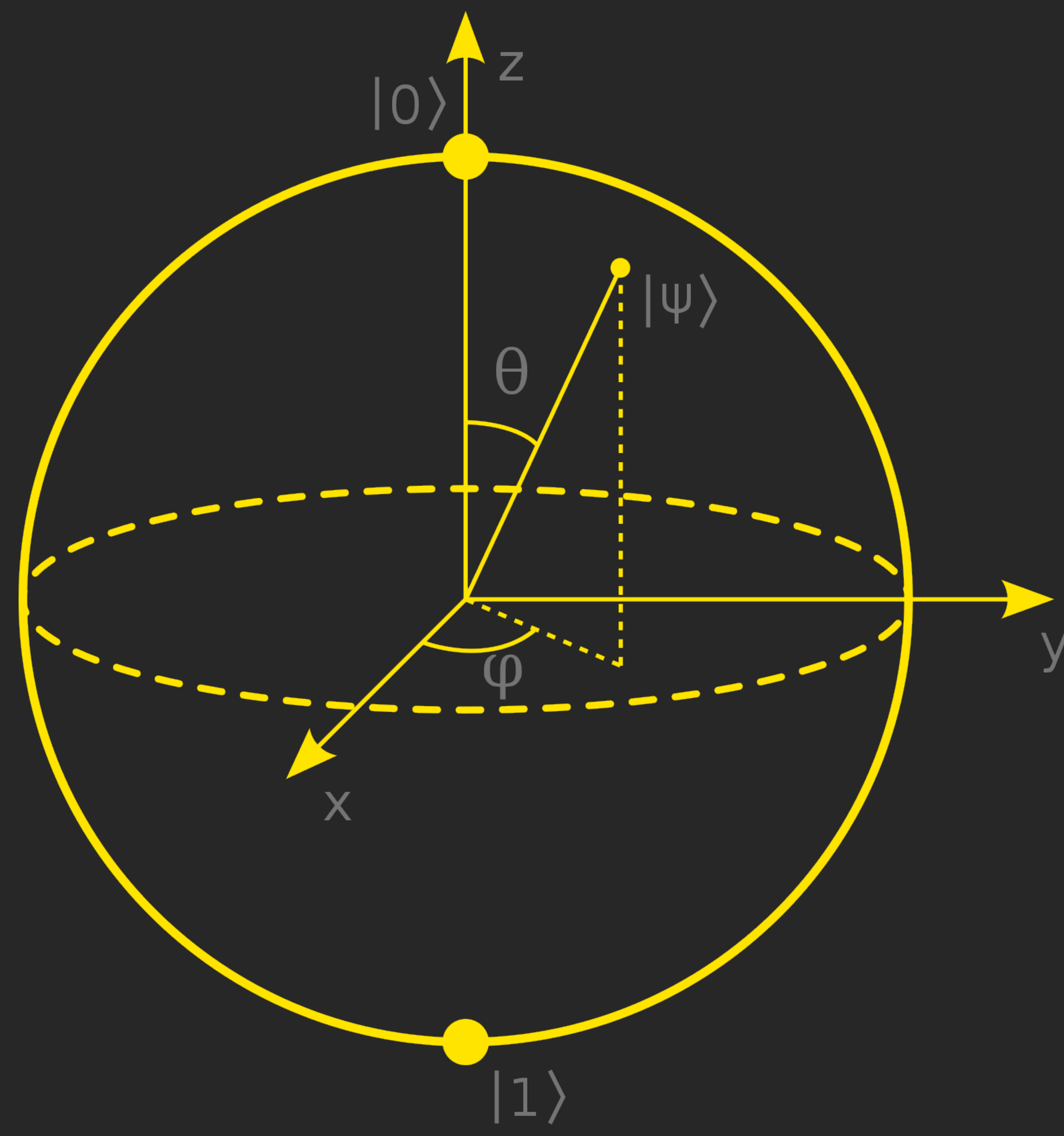
State of system given by

$$|\psi\rangle \in \mathbb{C}^2$$

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Linear combination is <u>not</u> 0 and 1

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$



$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$

Classical computers are based on manipulating bits of information, i.e. 0 and 1

The manipulation is called “logical gates”

Single bit classical gate

$$NOT: \quad 0 \rightarrow 1 \quad 1 \rightarrow 0$$

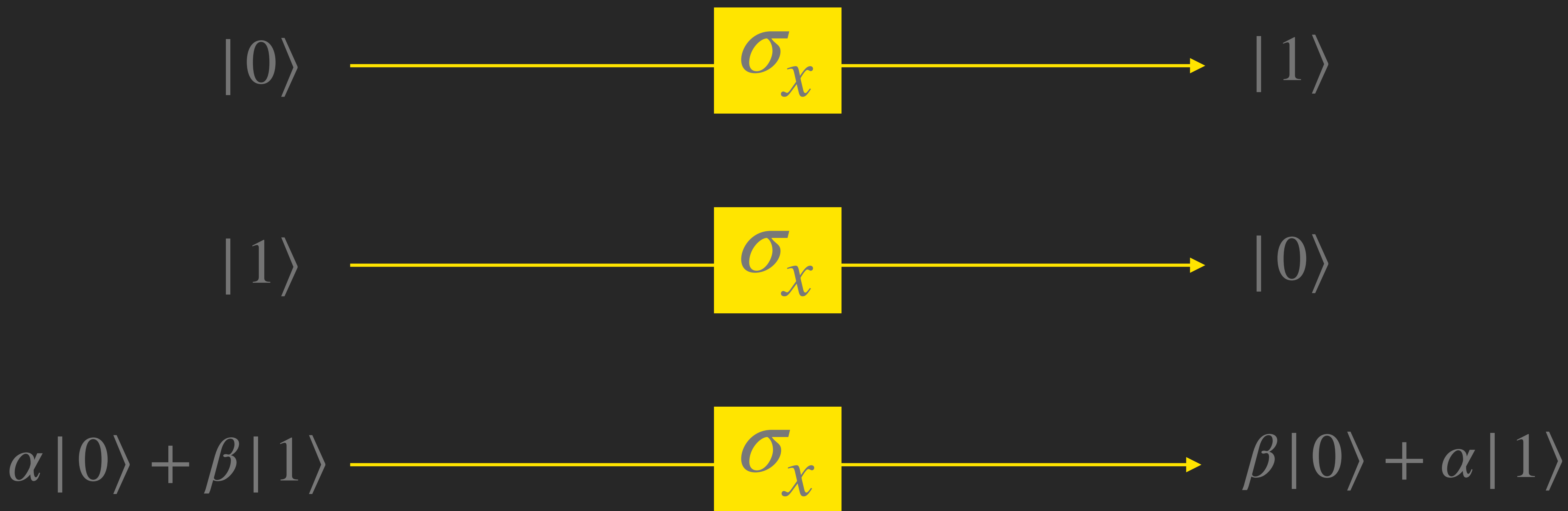
$$ERASE: \quad 0 \rightarrow 0 \quad 1 \rightarrow 0$$

There are also quantum mechanical equivalent operations, called quantum gates

These are the basic operations we can use in a quantum computer

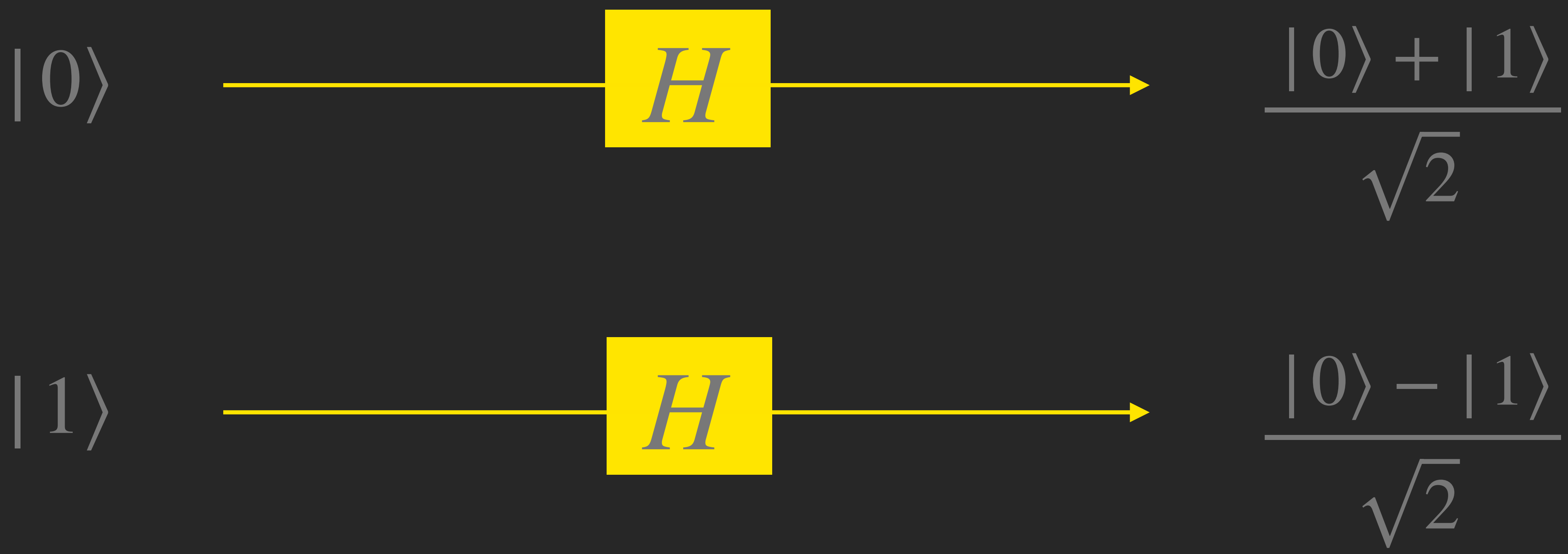
$$NOT: \quad 0 \to 1 \qquad 1 \to 0$$

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

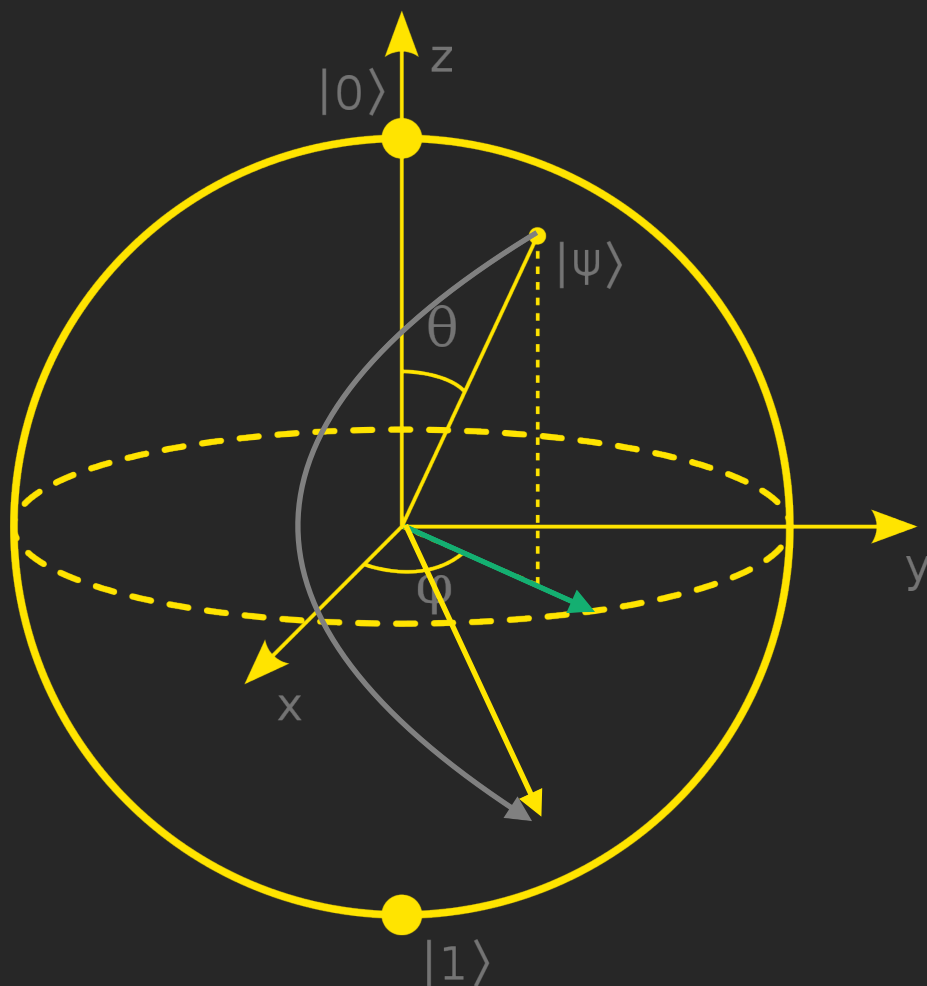


Hadamard gate

$$H = \frac{1}{\sqrt{2}}\left(\sigma_x + \sigma_z\right) = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$



General rotation on Bloch sphere



$$U = e^{i\alpha} R_{\hat{n}}(\theta)$$

$$R_{\hat{n}}(\theta) = \cos(\theta/2) I - i \sin(\theta/2) \Big(n_x \sigma_x + n_y \sigma_y + n_z \sigma_z \Big)$$

What about classical gates for two bits?

Classical CNOT

Flip target bit if control bit is 1

ct		$c\hat{t}$
00	$\rightarrow$	00
01	$\rightarrow$	01
10	$\rightarrow$	11
11	$\rightarrow$	10

What about operation on two qubits?

Tensor product states

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad |10\rangle = |1\rangle \otimes |0\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad |11\rangle = |1\rangle \otimes |1\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

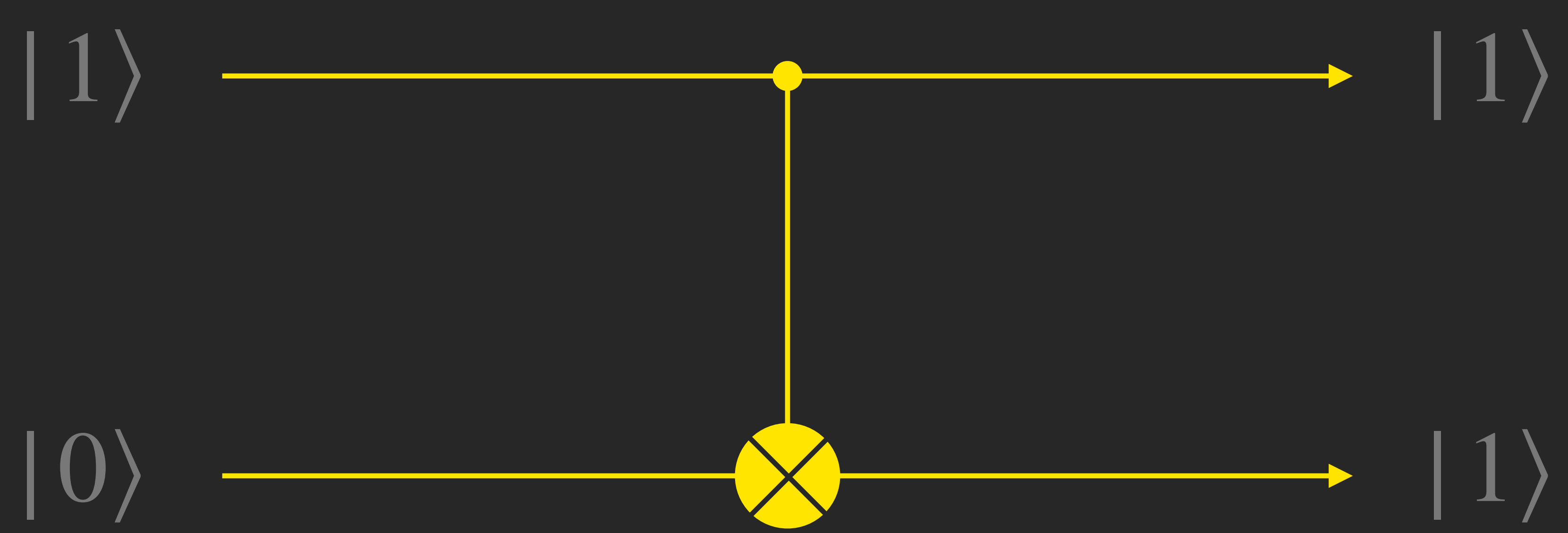
$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$$

Quantum CNOT

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

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

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



Generate entanglement

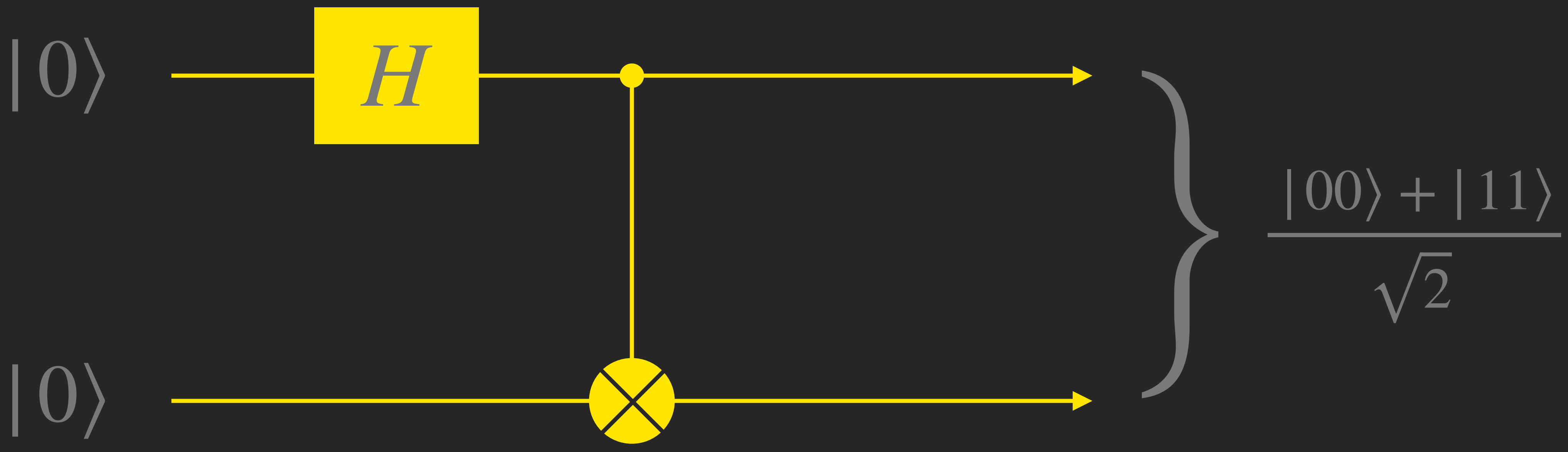
$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$|00\rangle \rightarrow \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)|0\rangle = \frac{|00\rangle + |10\rangle}{\sqrt{2}}$$

Generate entanglement

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

$$|00\rangle \rightarrow \frac{|00\rangle + |10\rangle}{\sqrt{2}} \rightarrow \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$



N qubits lives in 2^N

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

$$|\psi\rangle \in \mathbb{C}^2$$

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

$$|\psi_1 \psi_2\rangle \in \mathbb{C}^4$$

$$|01101001\ldots\rangle$$

$$|\psi_1 \psi_2 \ldots \psi_n\rangle \in \mathbb{C}^{2^n}$$

Classical data

$$\vec{v} = \begin{bmatrix} 0.54 \\ 0.83 \end{bmatrix}$$

$$\vec{v} = \begin{bmatrix} 0.54 \\ 0.44 \\ 0.31 \\ 0.63 \end{bmatrix}$$

Quantum data

$$|\psi\rangle = 0.54\,|0\rangle + 0.83\,|1\rangle$$

$$|\psi\phi\rangle = 0.54\,|00\rangle + 0.44\,|01\rangle + 0.31\,|10\rangle + 0.66\,|11\rangle$$

n qubits can store 2^n numbers

2^n numbers can be compressed into n qubits

$$\vec{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{2^n} \end{bmatrix} \in \mathbb{R}^{2^n} \longrightarrow |q_1 q_2 \ldots q_n\rangle = \sum_{i=1}^{2^n} v_i |i\rangle$$

$O(n)$: how the number of operations in an algorithm scales with the input

n = number of classical data points	Classical	Quantum
FFT	$O(n \log_2 n)$	$O\left(\log_2(n)^2\right)$
Eigenvalues Eigenvectors	$O(n^3)$	$O\left(\log_2(n)^2\right)$
Matrix inversion	$O(n \log_2 n)$	$O\left(\log_2(n)^3\right)$

1 GB classical data $n = 10^9 \to 30$ qubits

	Classical	Quantum
FFT	10^{10}	900
Eigenvalues Eigenvectors	10^{27}	900
Matrix inversion	10^{10}	26000

Three big caveats

1. Encoding classical information into qubits also have computational cost.

2. Measurement collapses wave function: if final state is superposition we need quantum tomography.

3. Environmental noise destroys quantum effects.

Quantum computing summary

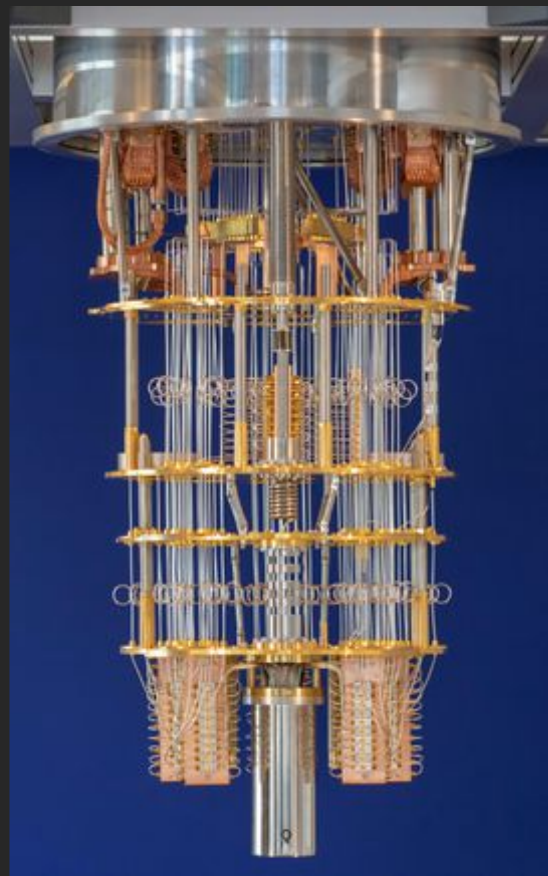
1. Quantum computers can take advantage of dimensional compression of classical data

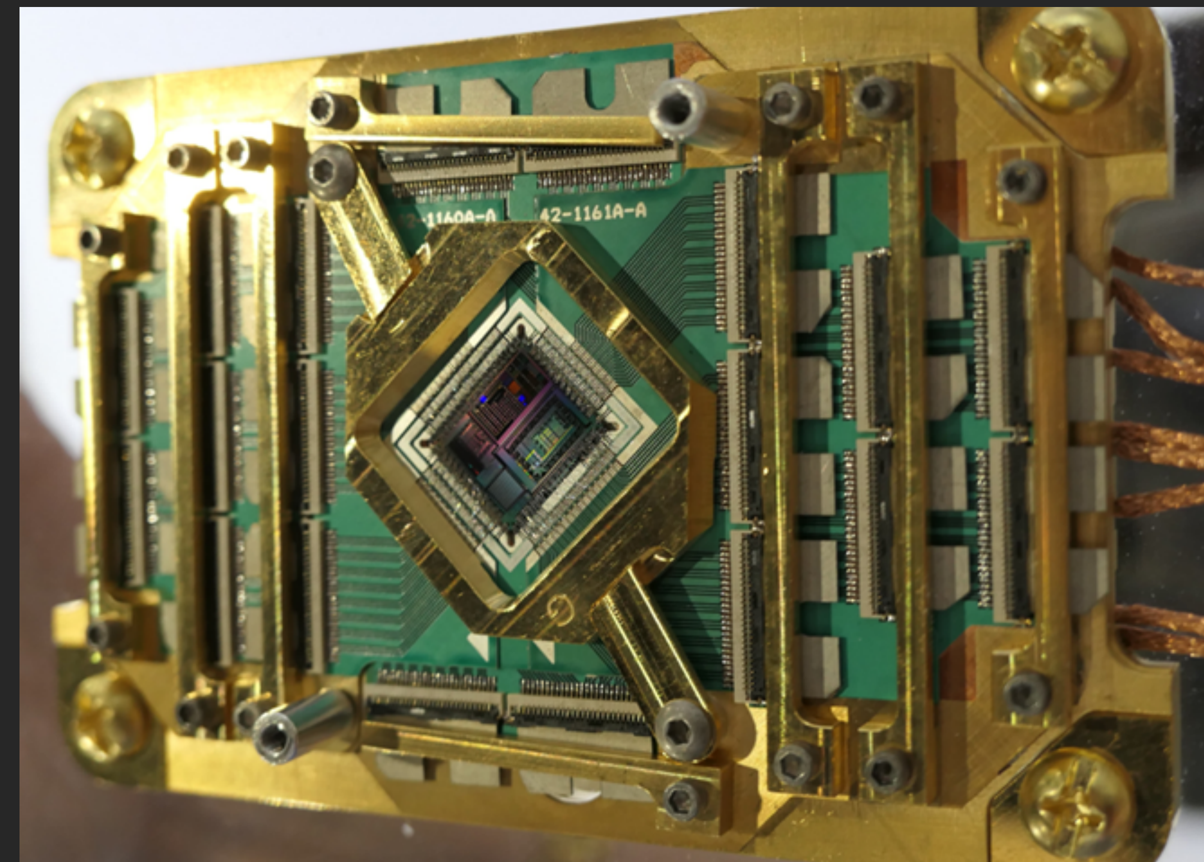
2. As well as quantum effects like superposition and entanglement

3. Able to perform some computations exponentially faster than classical counterpart

4. Importing and exporting classical data is non-trivial

5. Experimentally difficult to build due to sensitivity to environment





Built by commercial companies: IBM, Google, Intel, etc.

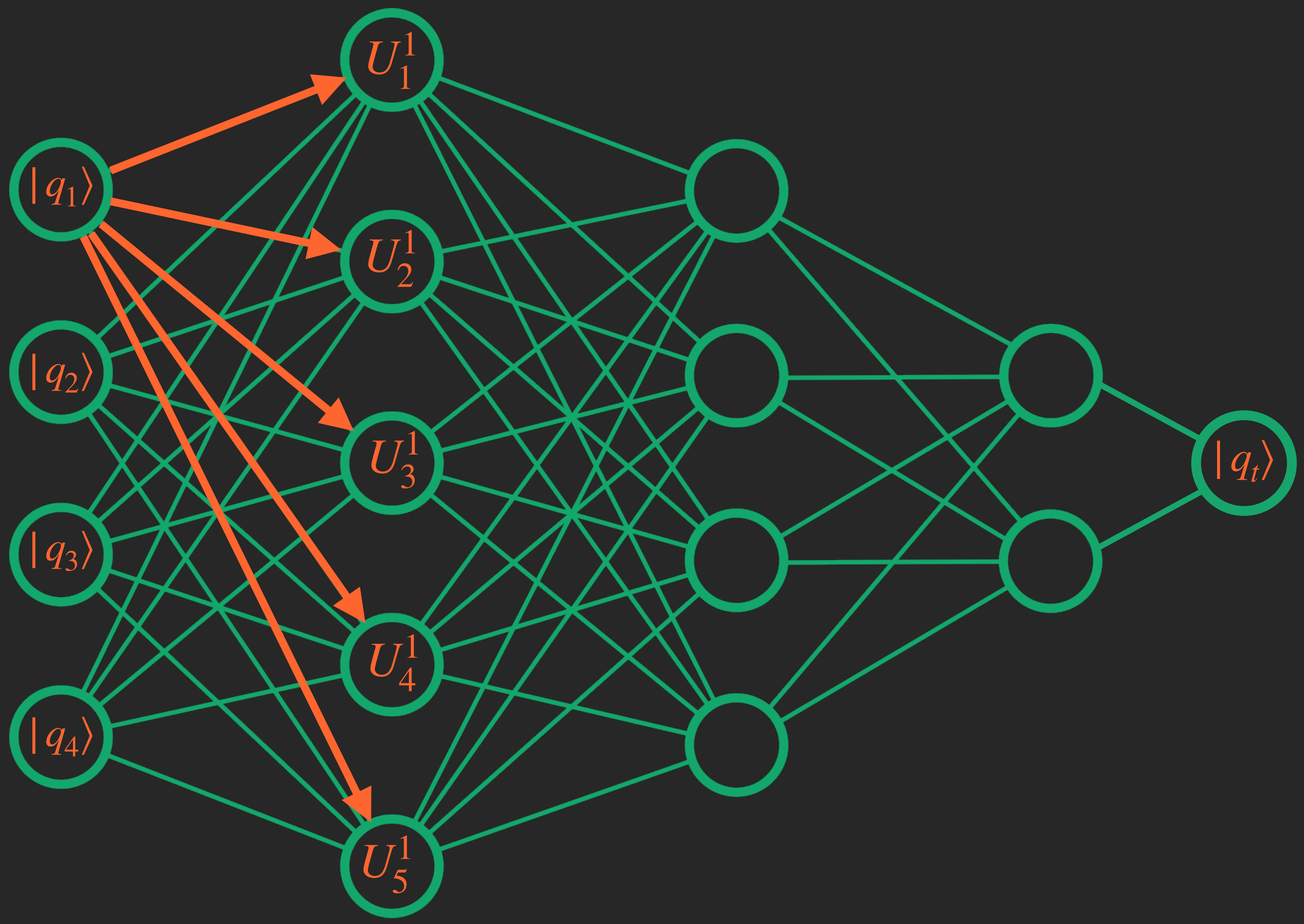
Shor’s algorithm: prime number factorization

Grovers’s algorithm: search unstructured database

Quantum computers could break many of modern classical encryption methods

Quantum Networks

Naive quantum network



No cloning: cannot make a copy of a quantum state

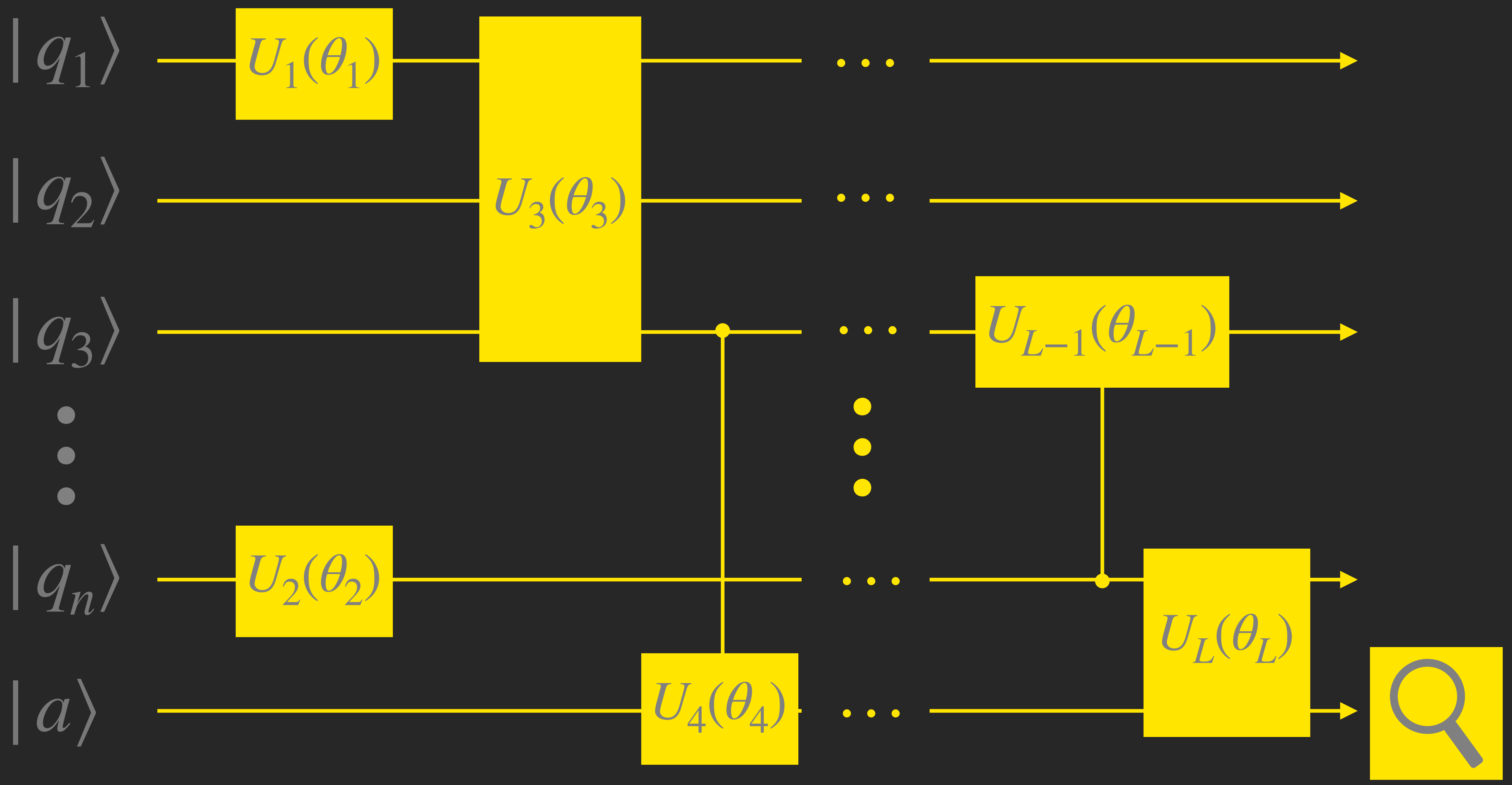
QNN requirements

1. Produces outputs that is closest to the target by some distance measure (i.e. minimized some cost function)

2. The QNN reflect one or more basic neural computing mechanisms

3. Must be based on quantum effects: superposition, entanglement and/or interference

Schuld, Maria, et al. "The quest for a quantum neural network." *Quantum Information Processing* 13.11 (2014): 2567-2586.



n qubits and one "special" qubit $|a\rangle$

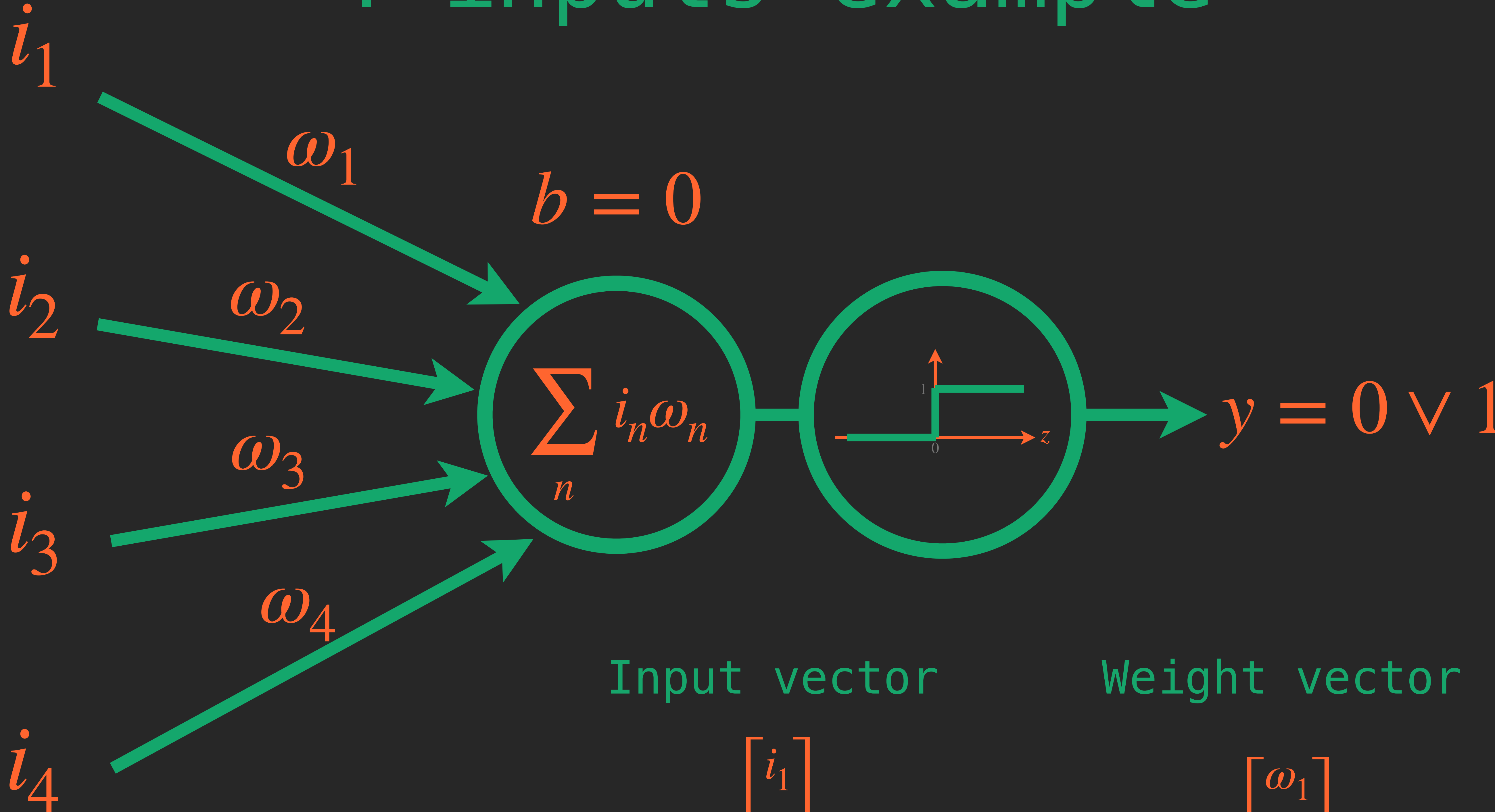
Tune parameters $\theta = \{\theta_1, \theta_2, \ldots, \theta_L\}$ such that measurement outcome is the desired one

Key power of neural networks comes from
non-linear activation functions

All operators in quantum mechanics are linear
except measurement

Focus on one particular implementation using measurement as activation

4 inputs example



Input vector

$$\vec{i} = \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix}$$

Weight vector

$$\vec{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}$$

We will now implement a quantum equivalent to this classical neuron

Input vector

$$\vec{i} = \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix}$$

Weight vector

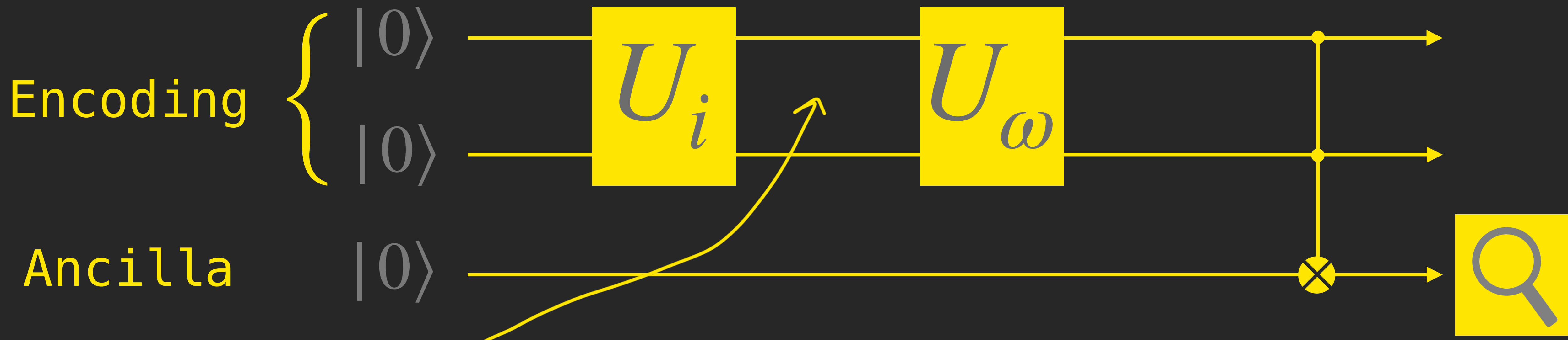
$$\vec{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}$$

McCulloch-Pitts neuron: $i_n, \omega_n \in \{-1, 1\}$

$$|\psi_i\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} i_n |n\rangle \qquad |\psi_\omega\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} \omega_n |n\rangle$$

$$|n\rangle \in \{ |00\rangle, |01\rangle, |10\rangle, |11\rangle \}$$

$$\left.\begin{aligned} |\psi_i\rangle &= \frac{1}{\sqrt{4}} \left(i_1 |00\rangle + i_2 |01\rangle + i_3 |10\rangle + i_4 |11\rangle \right) \\ |\psi_\omega\rangle &= \frac{1}{\sqrt{4}} \left(\omega_1 |00\rangle + \omega_2 |01\rangle + \omega_3 |10\rangle + \omega_4 |11\rangle \right) \end{aligned}\right\} \quad 4\langle \psi_\omega | \psi_i \rangle = \vec{i} \cdot \vec{\omega}$$



$$|\psi_i\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} i_n |n\rangle$$

$$|\psi_\omega\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} \omega_n |n\rangle$$

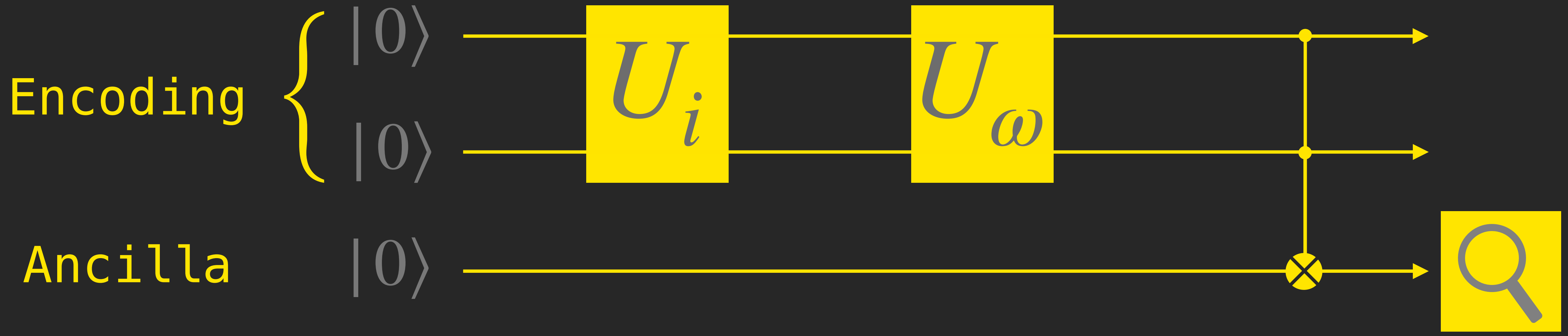
U_i prepares input vector

$$U_i|00\rangle = |\psi_i\rangle$$

Any unitary of the form

$$U_i = \begin{bmatrix} i_1 & \cdots & \cdots & \cdots \\ i_2 & \cdots & \cdots & \cdots \\ i_3 & \cdots & \cdots & \cdots \\ i_4 & \cdots & \cdots & \cdots \end{bmatrix} |00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Tacchino, F, et al. "An artificial neuron implemented on an actual quantum processor." *npj Quantum Information* 5.1 (2019): 26.



$$|\psi_i\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} i_n |n\rangle$$

$$|\psi_\omega\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} \omega_n |n\rangle$$

U_i prepares input vector

$$U_i|00\rangle = |\psi_i\rangle$$

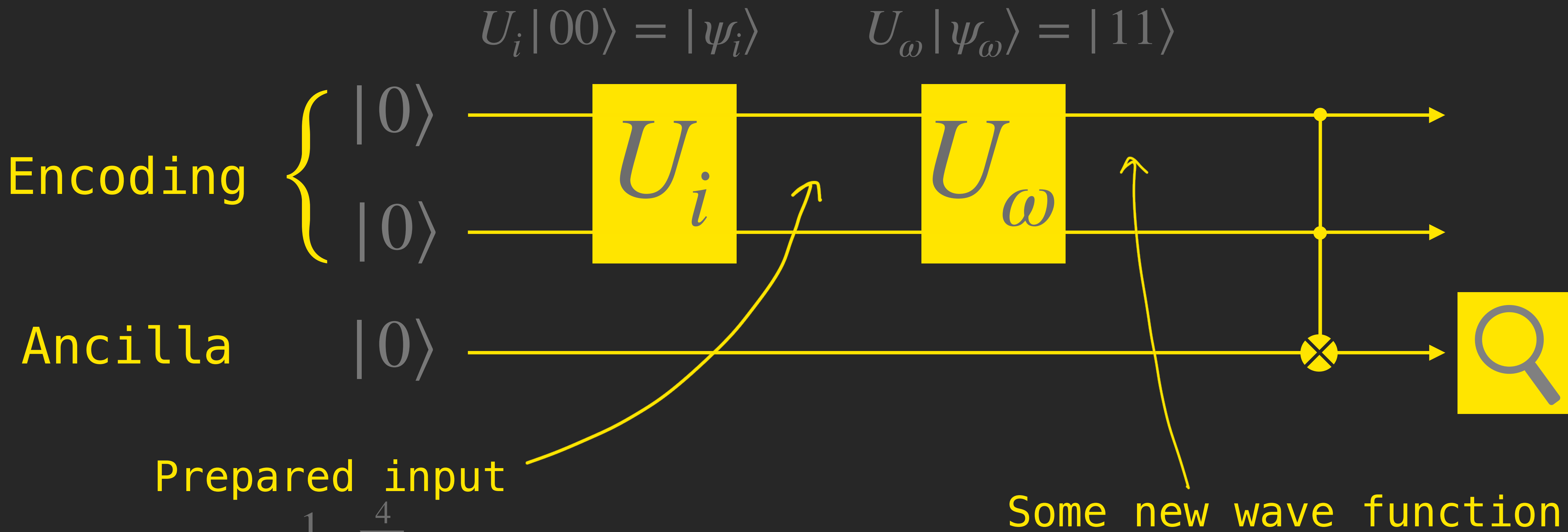
Any unitary of the form

$$U_\omega = \begin{bmatrix} \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \omega_1 & \omega_2 & \omega_3 & \omega_4 \end{bmatrix}$$

U_ω projects weight vector

$$U_\omega|\psi_\omega\rangle = |11\rangle$$

Tacchino, F, et al. "An artificial neuron implemented on an actual quantum processor." *npj Quantum Information* 5.1 (2019): 26.



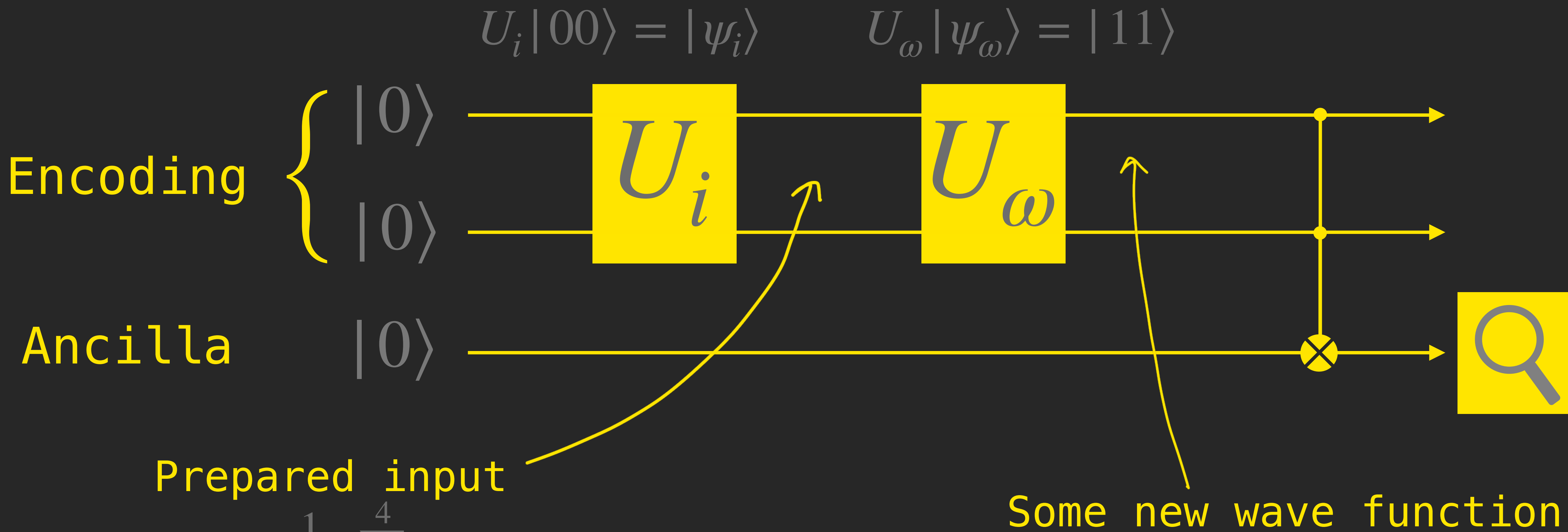
$$|\psi_i\rangle = \frac{1}{\sqrt{4}}\sum_{n=1}^{4} i_n|n\rangle$$

$$U_\omega|\psi_i\rangle \equiv |\phi_{i,\omega}\rangle = \sum_{n=1}^{4} c_n|n\rangle$$

$$|\phi_{i,\omega}\rangle = c_1|00\rangle + c_2|01\rangle + c_3|10\rangle + c_4|11\rangle$$

$$\frac{\vec{i}\cdot\vec{\omega}}{4} = \langle\psi_\omega|\psi_i\rangle = \langle\psi_\omega|U_\omega^\dagger U_\omega|\psi_i\rangle = \langle 11|\phi_{i,\omega}\rangle = c_4$$

Inner product of input and weight vector has been encoded in c_4



$$|\psi_i\rangle = \frac{1}{\sqrt{4}} \sum_{n=1}^{4} i_n |n\rangle$$

$$U_\omega |\psi_i\rangle \equiv |\phi_{i,\omega}\rangle = \sum_{n=1}^{4} c_n |n\rangle$$

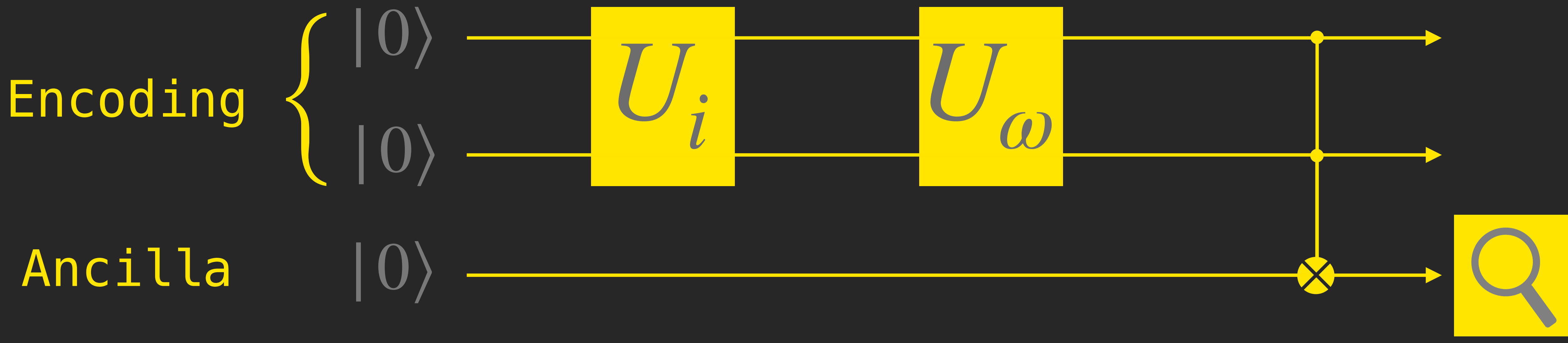
$$|\phi_{i,\omega}\rangle|0\rangle = c_1|000\rangle + c_2|010\rangle + c_3|100\rangle + c_4|110\rangle$$

$$CNOT \downarrow$$

$$|out\rangle = c_1|000\rangle + c_2|010\rangle + c_3|100\rangle + c_4|111\rangle$$

Tacchino, F, et al. "An artificial neuron implemented on an actual quantum processor." *npj Quantum Information* 5.1 (2019): 26.

$U_i|00\rangle = |\psi_i\rangle \qquad U_\omega|\psi_\omega\rangle = |11\rangle$



$$|out\rangle = c_1|000\rangle + c_2|010\rangle + c_3|100\rangle + c_4|111\rangle$$

Classical neuron is **activated** if the weighted sum of input is larger than some bias.

Probability to measure ancilla in state 1 (**activated neuron**) is proportional to the weighted sum of input

$$P_{act} = |c_4|^2 = \frac{|\sum_n^4 i_n \omega_n|^2}{4^2}$$

$$\vec{i} \cdot \vec{\omega} = 4\langle\psi_\omega|\psi_i\rangle = 4c_4$$

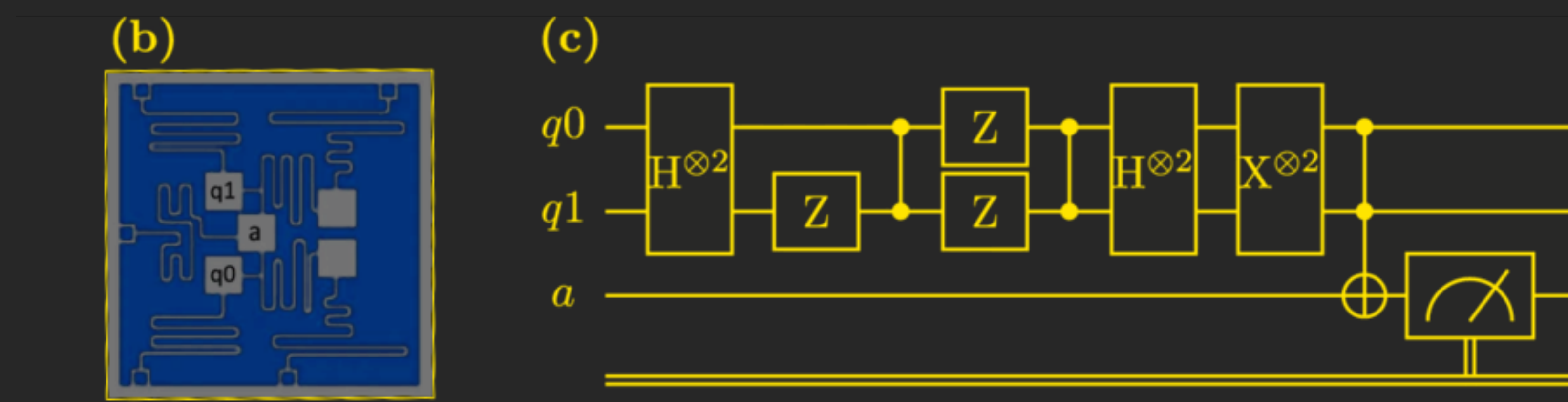
If $\vec{i} \parallel \vec{\omega}$ the probability of activation is 1

$$\vec{i} = \vec{\omega} \qquad \vec{i} = -\vec{\omega}$$

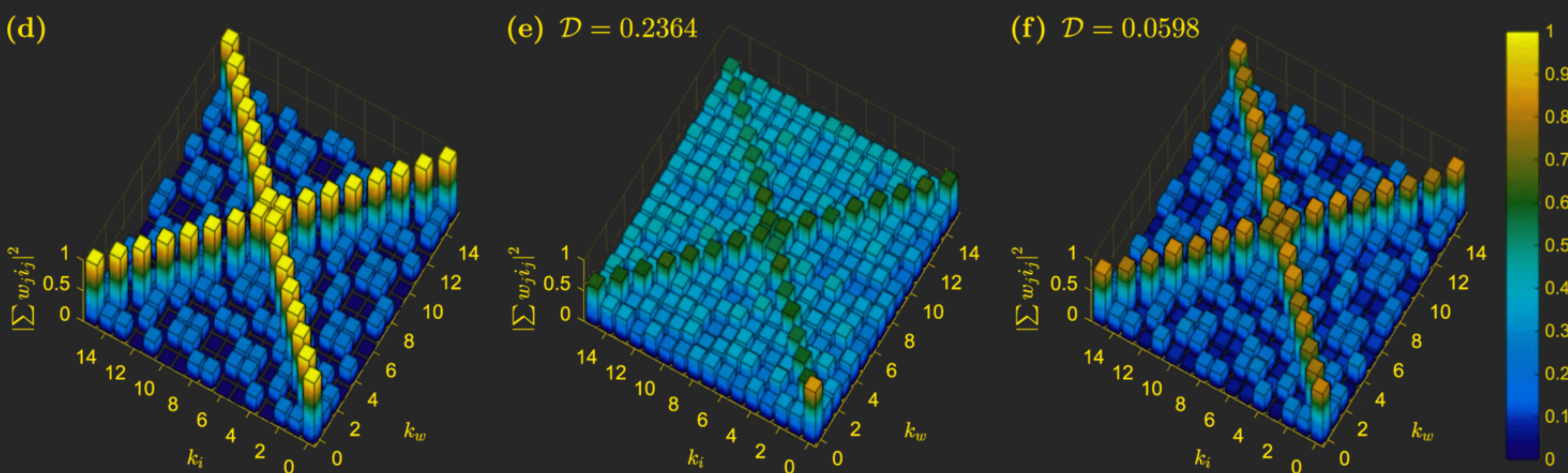
Tacchino, F, et al. "An artificial neuron implemented on an actual quantum processor." *npj Quantum Information* 5.1 (2019): 26.

Pattern recognition

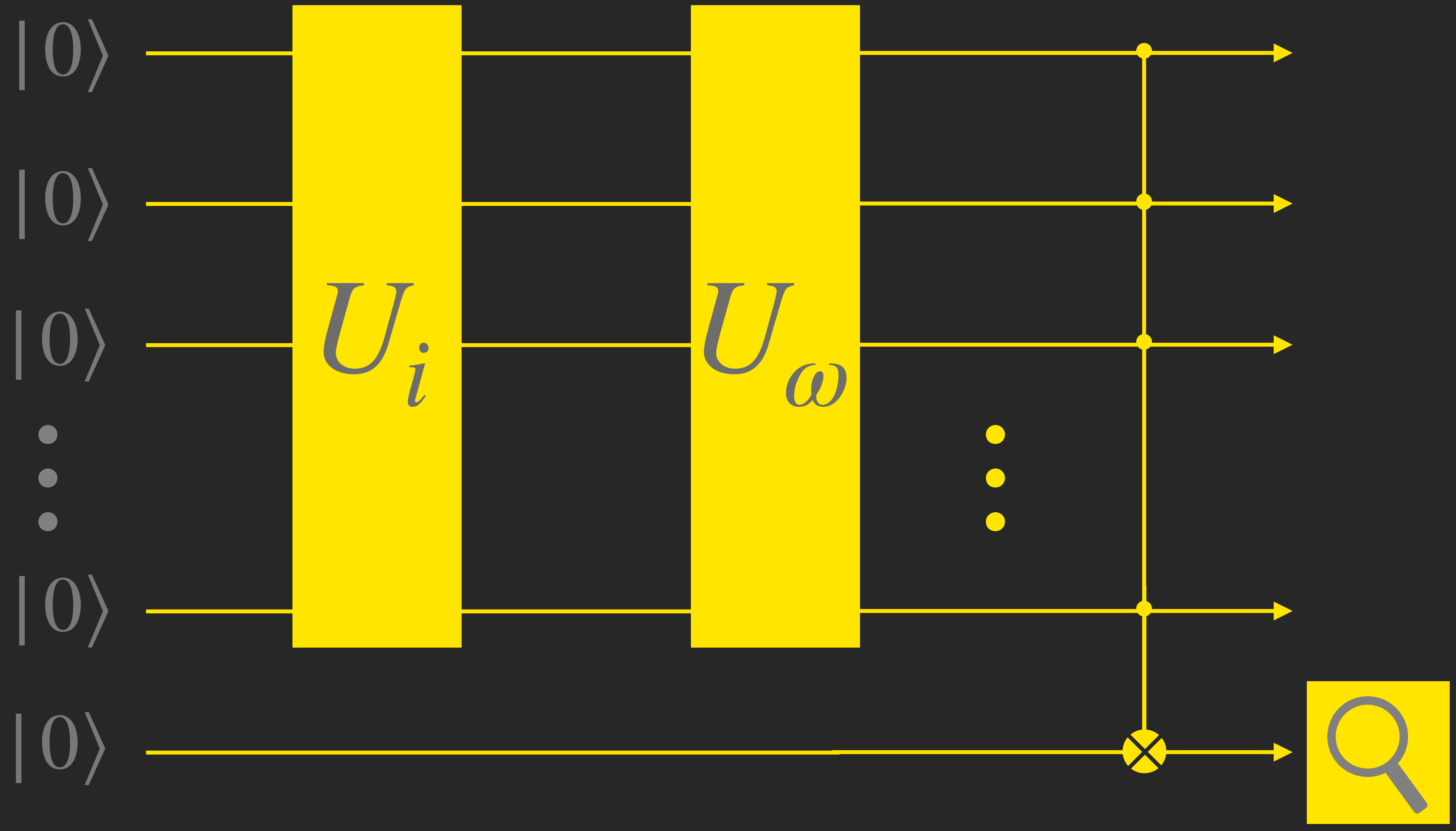
$$\vec{i} = \begin{bmatrix} \pm 1 \\ \pm 1 \\ \pm 1 \\ \pm 1 \end{bmatrix} \quad \vec{\omega} = \begin{bmatrix} \pm 1 \\ \pm 1 \\ \pm 1 \\ \pm 1 \end{bmatrix}$$



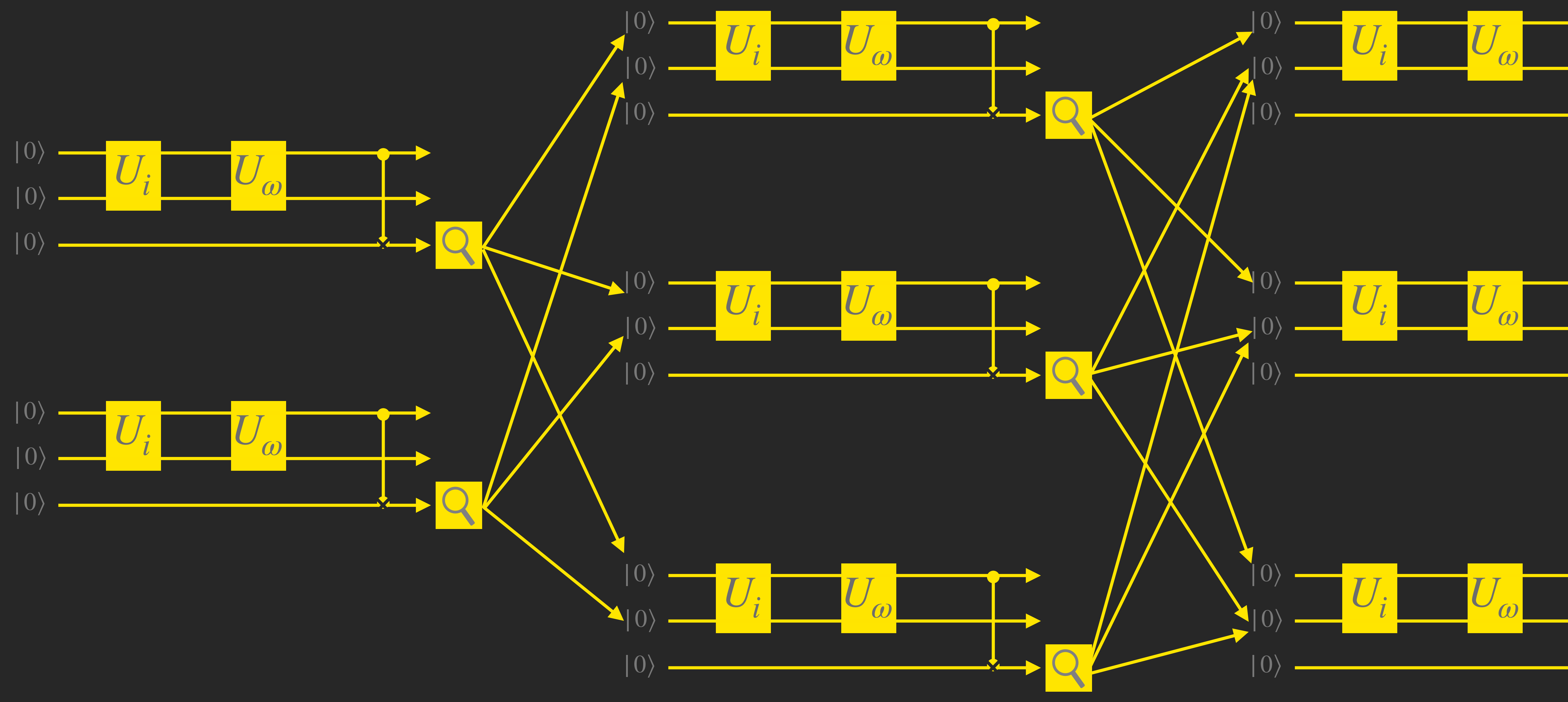
$4^2 = 16$ different possible vectors



Tacchino, F, et al. "An artificial neuron implemented on an actual quantum processor." *npj Quantum Information* 5.1 (2019): 26.

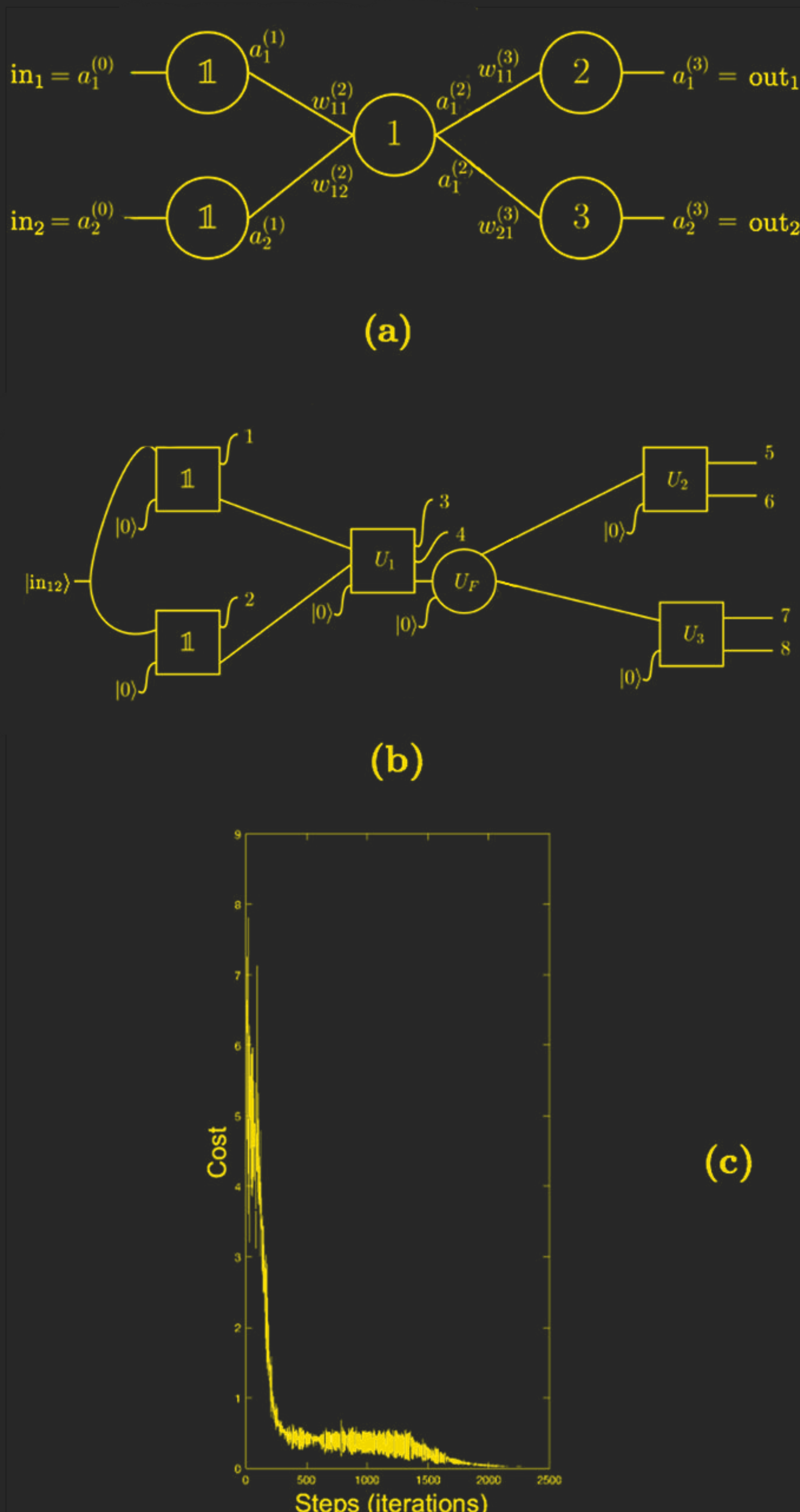


Tacchino, F, et al. "An artificial neuron implemented on an actual quantum processor." *npj Quantum Information* 5.1 (2019): 26.



Every measurement destroys quantum effects
→ classical propagation of probabilities

So this essentially becomes a classical network
→ no quantum benefits



Another QNN

Implemented on simulator

Distribute quantum states using “fan-out” operator

Classically updates operators by using gradient decent

$$\frac{\partial C}{\partial \theta}$$

This gradient is very difficult to experimentally access

And θ grows polynomially with number of qubits

“A key type of quantum supremacy is that the quantum network can take and process quantum inputs: it can for example process $|+\rangle$ and $|-\rangle$ differently.”

Wan, Kwok Ho, et al. "Quantum generalisation of feedforward neural networks." *npj Quantum Information* 3.1 (2017): 36.

“Based on our numerics we cannot make a case for any quantum advantage over classical competitors for supervised learning.”

Farhi, Edward, et al. "Classification with quantum neural networks on near term processors." *arXiv preprint arXiv:1802.06002* (2018).

“As a conclusion, QNN research has not found a coherent approach yet and none of the competing ideas can fully claim to be a QNN model according to the requirements set here.”

Schuld, Maria, et al. "The quest for a quantum neural network." *Quantum Information Processing* 13.11 (2014): 2567-2586.

Conclusion and outlook

Neural networks and quantum computers are individually powerful concepts

No clear and natural reason to merge the two

Fundamental difference between non-linear NN and linear quantum mechanics difficult to reconcile

More work required to find eventual benefit: very young field