Úvod do teorie neuronových sítí

Kamila Lepkova

Biomedicínské inženýrství (ČVUT),

kamila.lepkova(at)cvut.cz

Co je to AI ?

Definice AI podle EU ACT

„Systém umělé inteligence“ (AI systém) znamená systém, který je navržen tak, aby fungoval s určitou úrovní autonomie a který ***na základě dat a vstupů poskytnutých strojem a/nebo člověkem vyvozuje, jak dosáhnout daného lidmi definovaného cíle*** *využívající strojové učení a/nebo přístupy založené na logice a znalostech a vytváří systémově generované výstupy, jako je obsah (generativní systémy umělé inteligence), předpovědi, doporučení nebo rozhodnutí, které ovlivňují prostředí, se kterými systém umělé inteligence interaguje.*

Historie AI

1943 — Vytvořen první matematický model neuronu - Alan Turing

1955 — První použití termínu umělá inteligence John McCarthy

1966 — Chatbot ELIZA a robot Shakey

1989 — První samořídící auto Testlab 1

1996 — Deep Blue poráží Garyho Kasparova

Historie AI

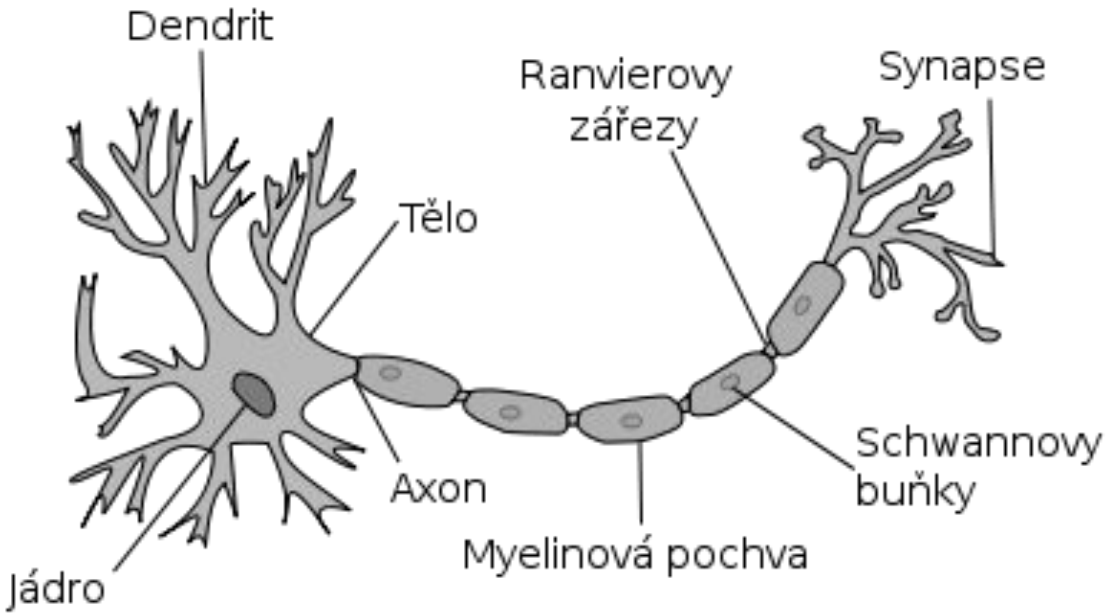

Dataset ImageNet
2006
2010
Watson vyhrává Riskuj!
SIRI a průlom ve zpracování řeči
2011
2017
AlphaGo poráží Lee Sedola v Go
GPT-3: Počítač píše knihu
2020

Neuronová síť

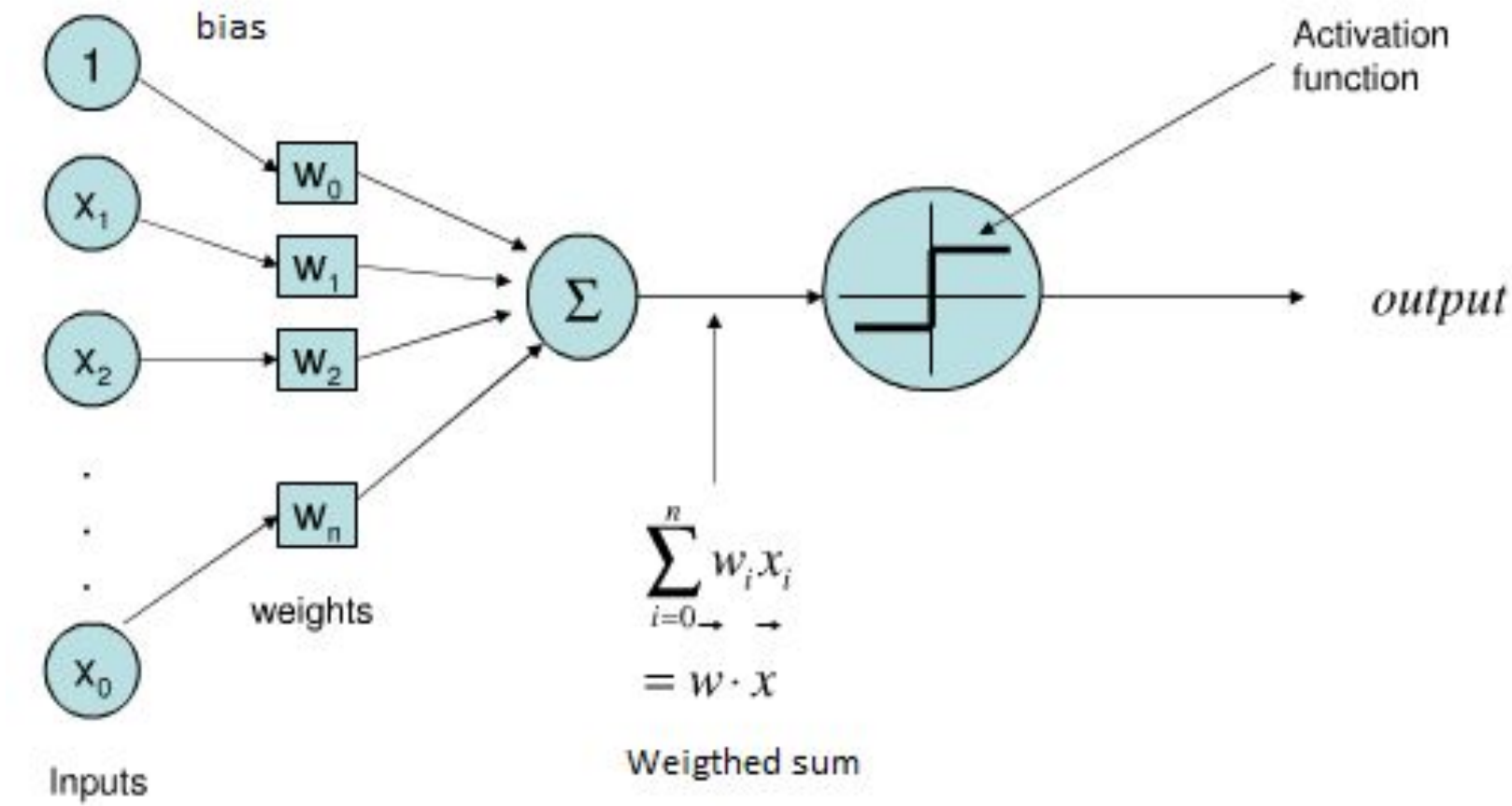
- Struktura / algoritmus / model určený pro paralelní zpracování dat, který se skládá z velkého počtu výkonných prvků - neuronů.
- Inspirován neurofyziologickými poznatky o struktuře a činnosti neuronů živých organismů.
- Schopnost extrahovat a reprezentovat závislost, učit se z předchozí zkušenosti, řešit nelineární úlohy, generalizovat znalosti.
- Aplikace: regrese, klasifikace, shluková analýza, optimalizace, predikce, filtrace, komprese dat
- Využití v reálném světě: lékařství (detekce, segmentace), radiologie, zpracování signálů, zpracování obrazů, modelování

Základní stavební jednotka – Perceptron / Neuron

- Biologický neuron: sběr, zpracování a přenos informací
- vstup (dendrity) / tělo (soma) / výstup (axon)



Perceptron - Neuron



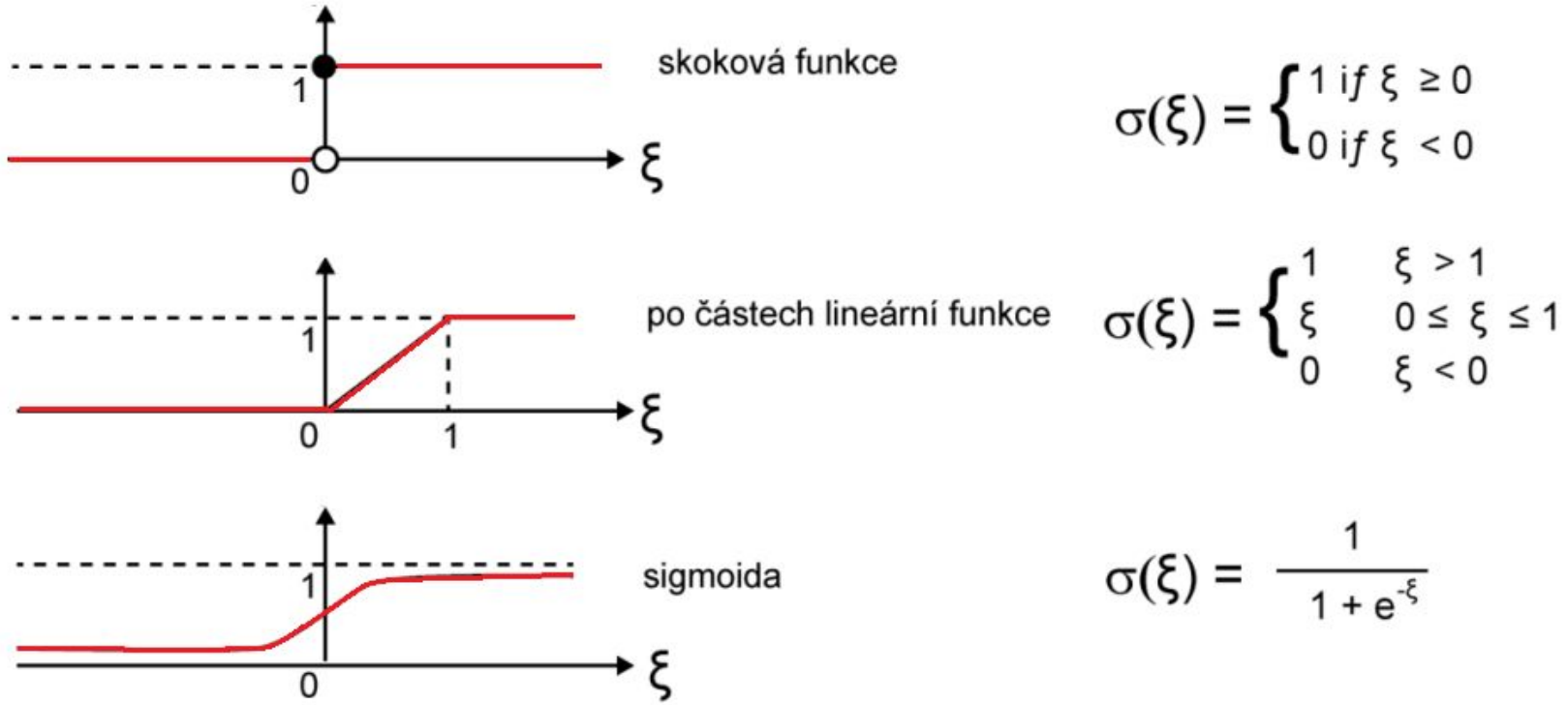
Části perceptronu

- Vstup (Input): specifické informace (reálné číslo: velikost, barva,…)
- Váhy (Weights): důležitost informace (reálné hodnoty)
- Práhová hodnota (Bias): speciální váha s hodnotou 1
- Výstup (Output): třídy, shluky,…

- Aktivační funkce (Activation function)

Aktivační funkce (α)

- Skoková
  (Step function)
- Sigmoidální
  (Sigmid function)
- Po částech lineární
  (Rectified Linear Unit function - ReLU)

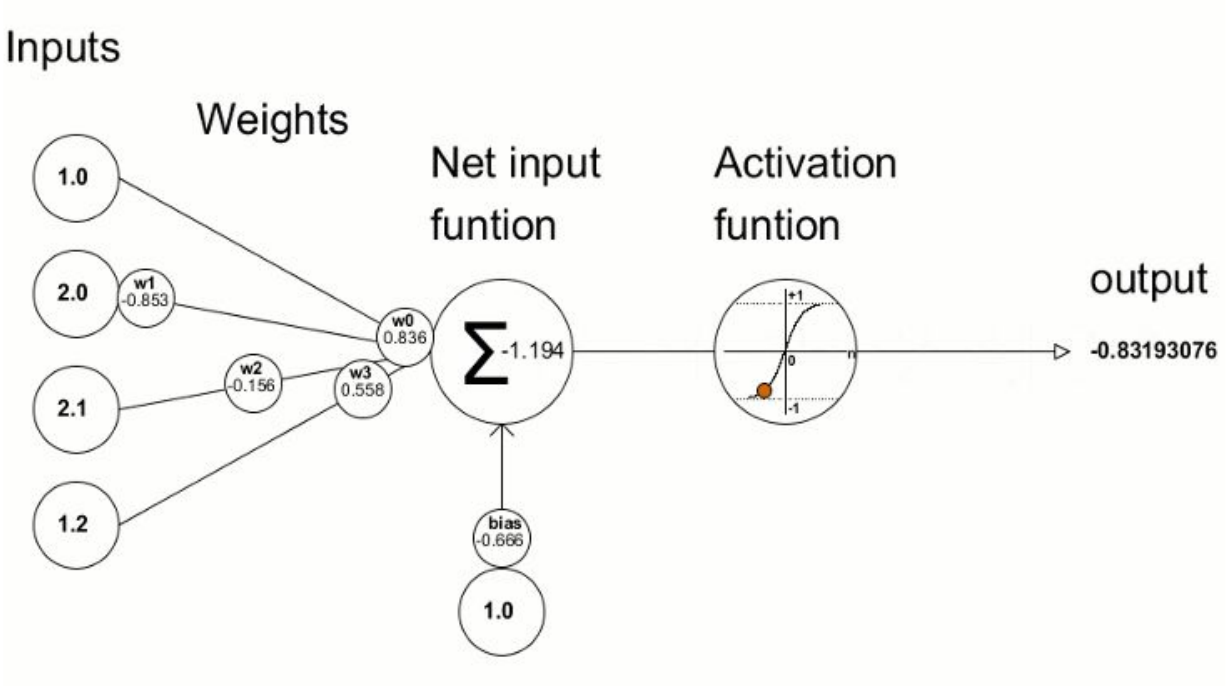


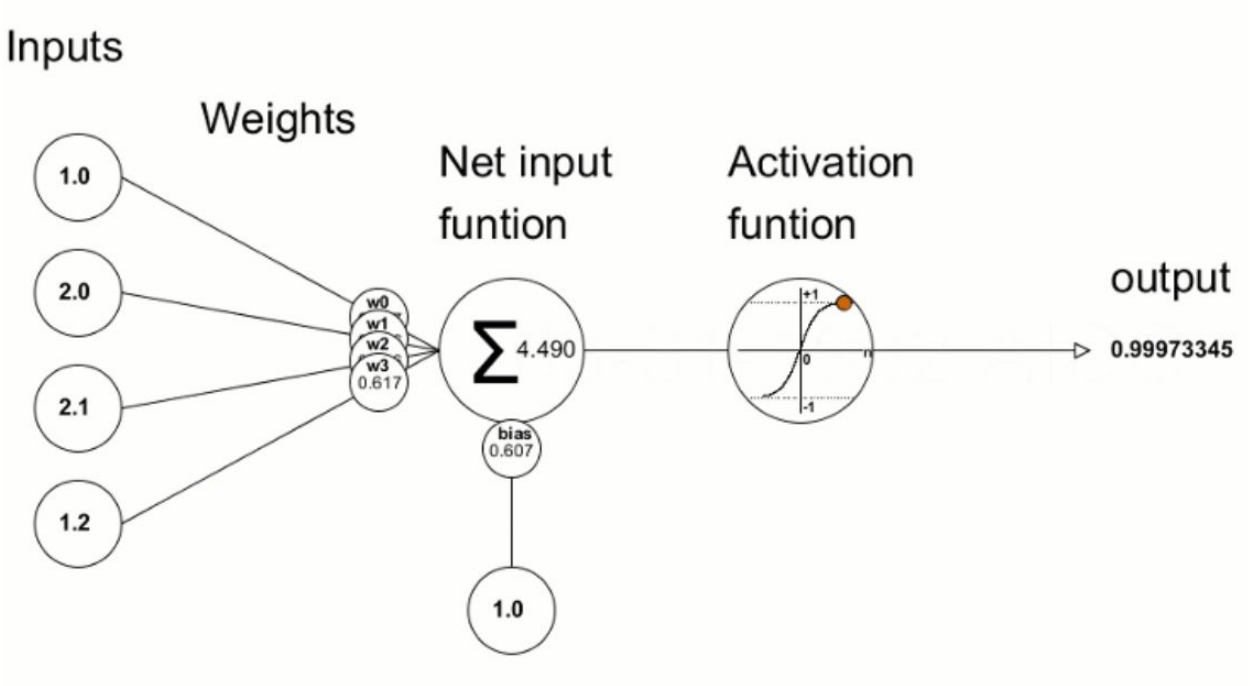
$$\sigma(\xi) = \begin{cases} 1 \text{ if } \xi \geq 0 \\ 0 \text{ if } \xi < 0 \end{cases}$$

$$\sigma(\xi) = \begin{cases} 1 & \xi > 1 \\ \xi & 0 \leq \xi \leq 1 \\ 0 & \xi < 0 \end{cases}$$

$$\sigma(\xi) = \frac{1}{1 + e^{-\xi}}$$

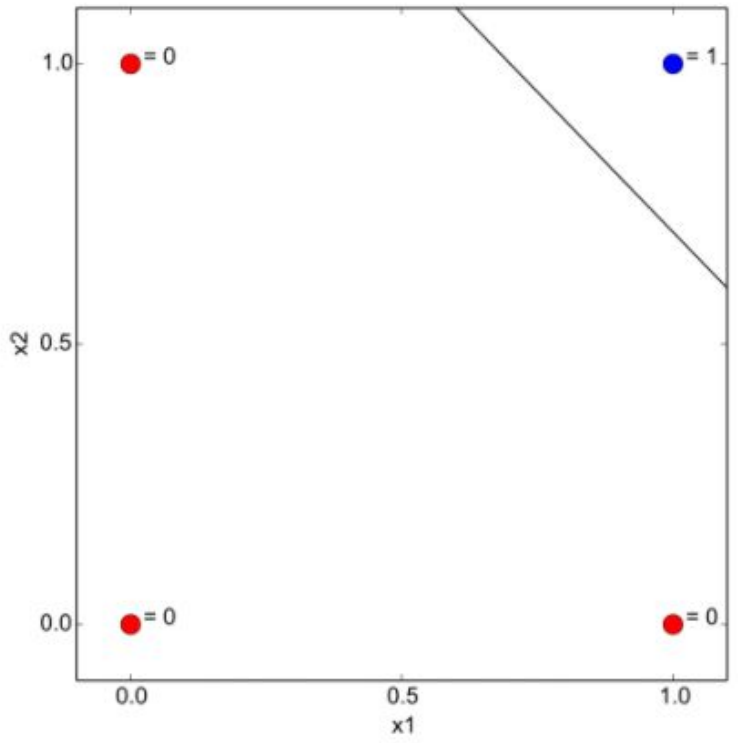
Příklad fungování perceptronu (tan-sigmoid)

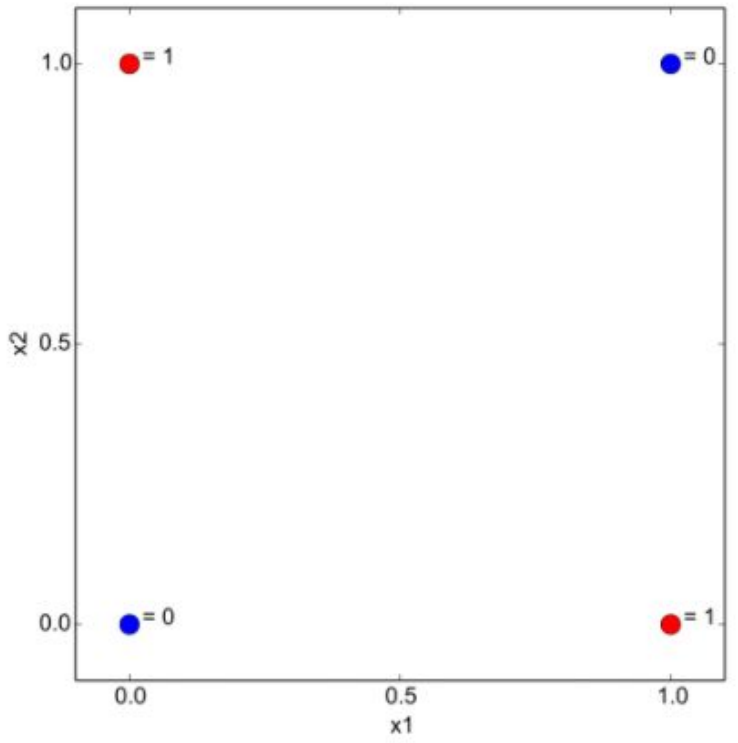




Perceptron

- Schopný klasifikace do dvou tříd – třídy musí být lineárně separované





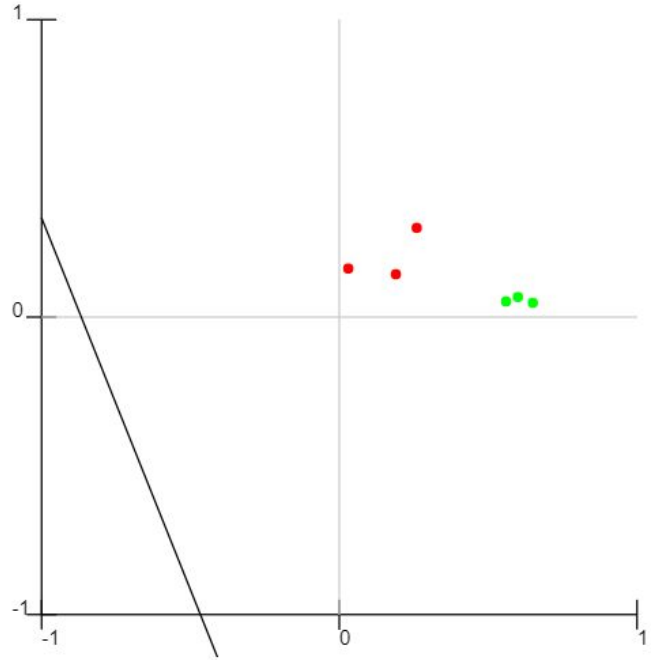
Proces výpočtu (Perceptron)

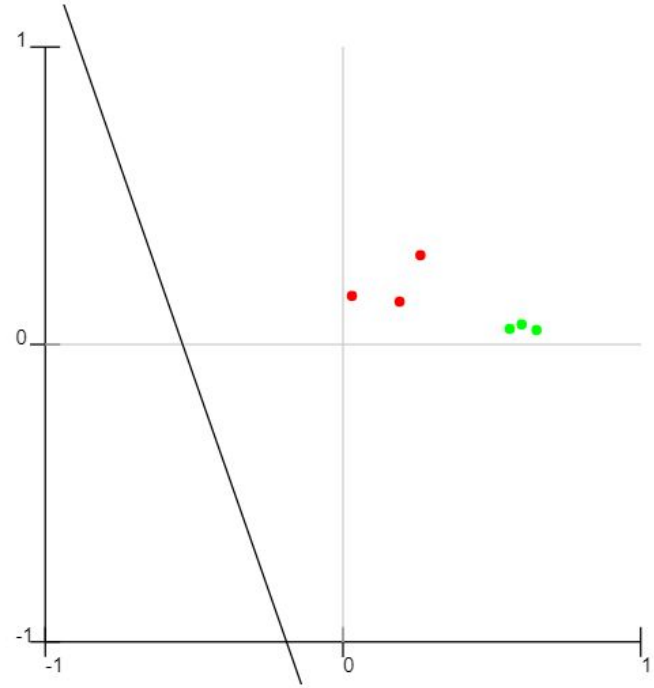
$$y(t) = sgn\left(\sum_{i=0}^{n} w_i(t)x_i(t)\right)$$

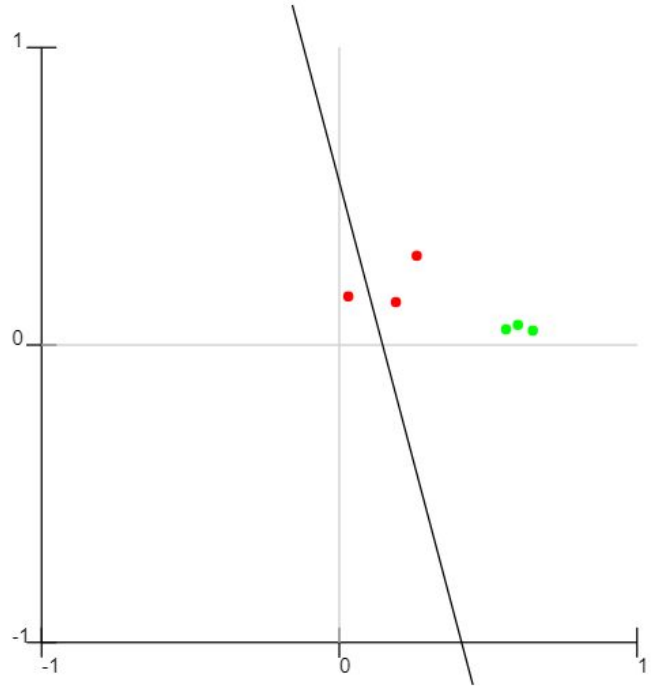
- 1. Inicializace vah $w_i(t)$
- 2. Výpočet sítě $y_i(t)$
- 3. Adaptace vah
  - Výstup je stejný: $w_i(t+1) = w_i(t)$
  - Výstup je 0, měl by být 1: $w_i(t+1) = w_i(t) + \eta x_i(t)$
  - Výstup je 1, měl by být 0: $w_i(t+1) = w_i(t) - \eta x_i(t)$

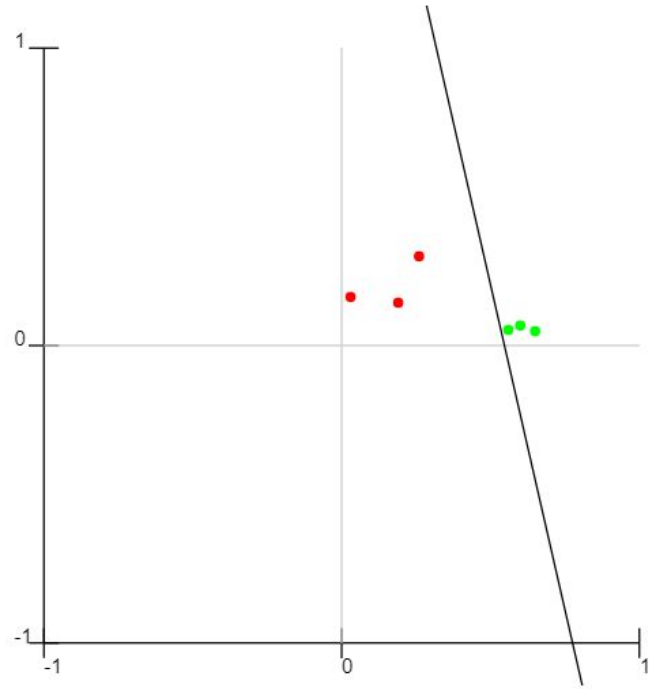
Adaptace vah, zobecnění:

$$\delta = z(t) - y(t)$$
$$w_i(t+1) = w_i(t) - \eta\delta x_i(t)$$
$$z(t) = očekávaný\ výstup$$


1
0
-1
-1
0
1


1
0
-1
-1
0
1

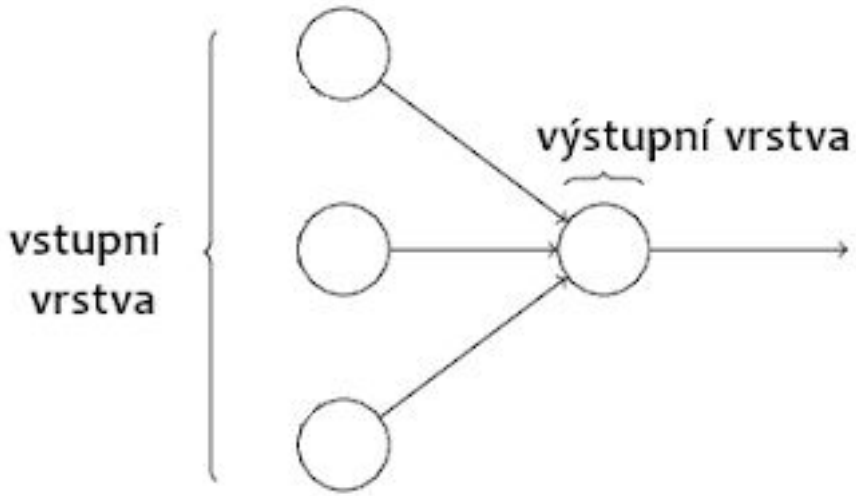

1
0
-1
-1
0
1

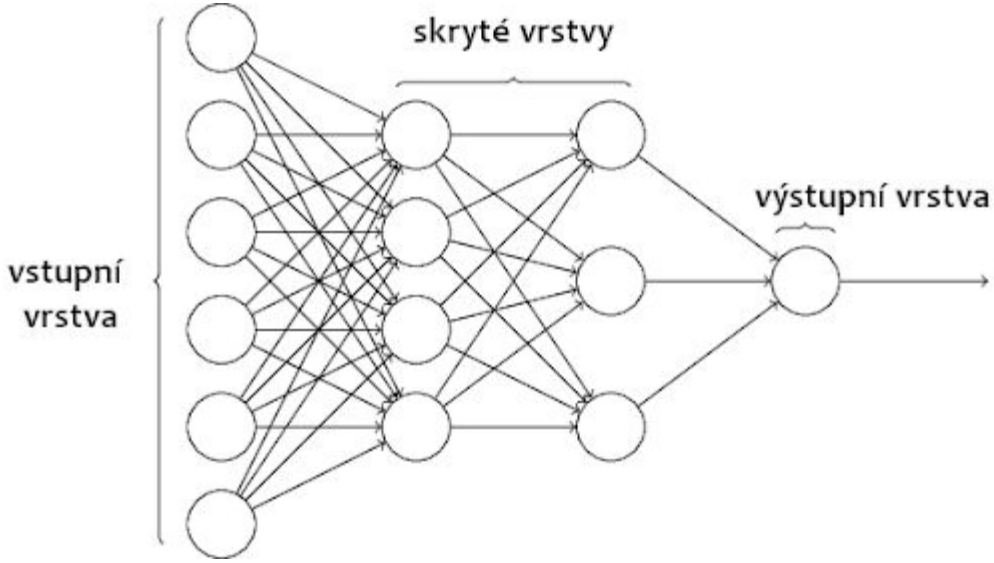

1
0
-1
-1
0
1

Rozdělení neuronových sítí

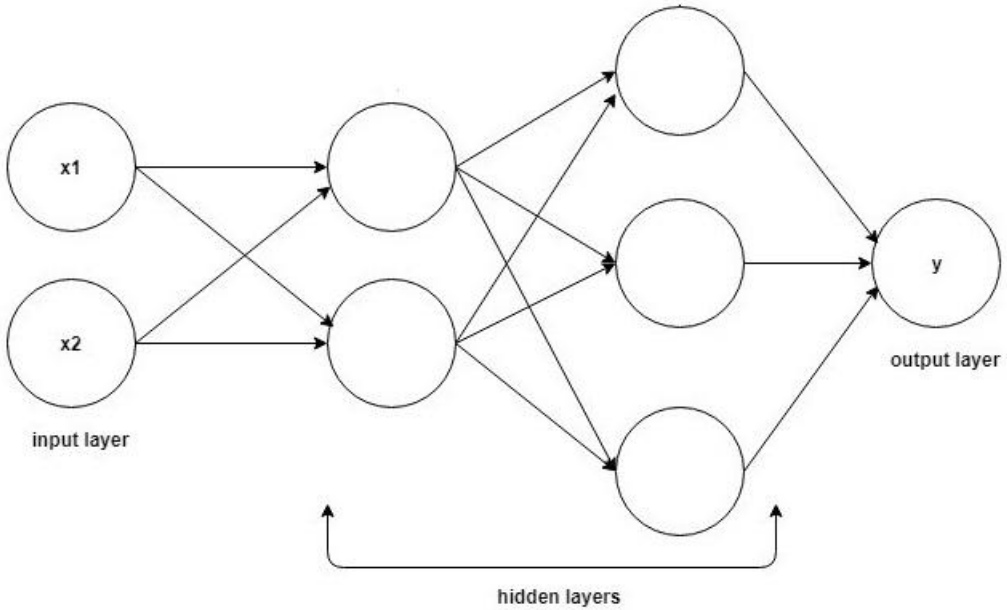
- Podle počtu vrstev:
  - Jednovrstvé (Hopfieldova síť)
  - Vícevrstvé (MLP, Konvoluční síť)
- Podle směru trénování:
  - Dopředné (Feed-forward)
  - Rekurentní (Recurrent)
- Podle algoritmů učení:
  - S učitelem (Supervised)
  - Bez učitele (Unsupervised)
- Podle stylu učení:
  - Deterministické (Backpropagation)
  - Stochastické (Náhodné nastavování vah)

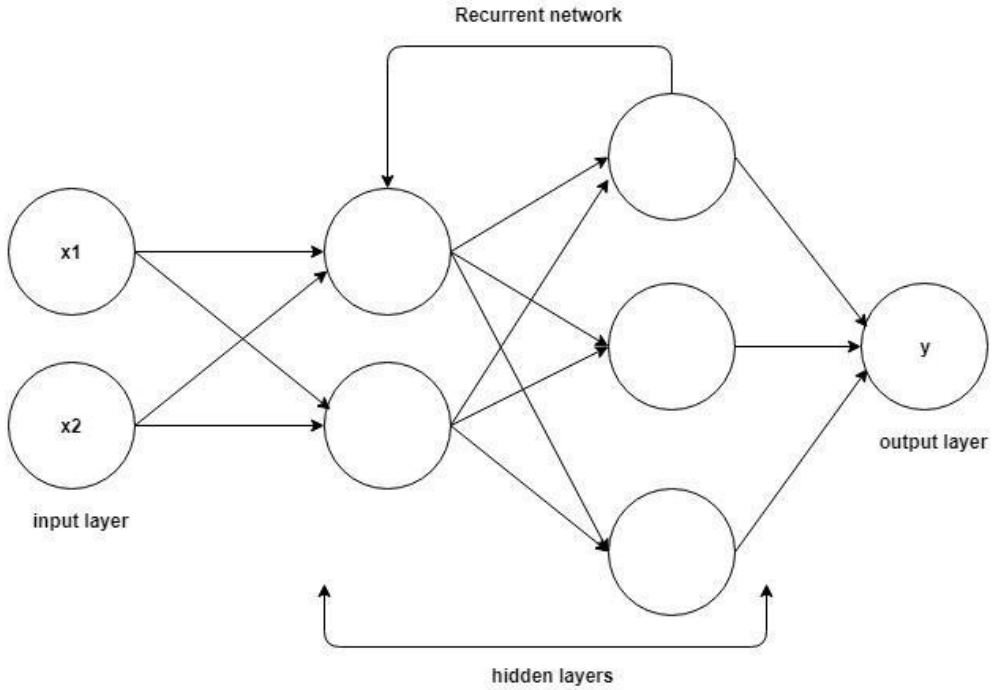
Jednovrstvé / Vícevrstvé





Dopředné / Rekurentní

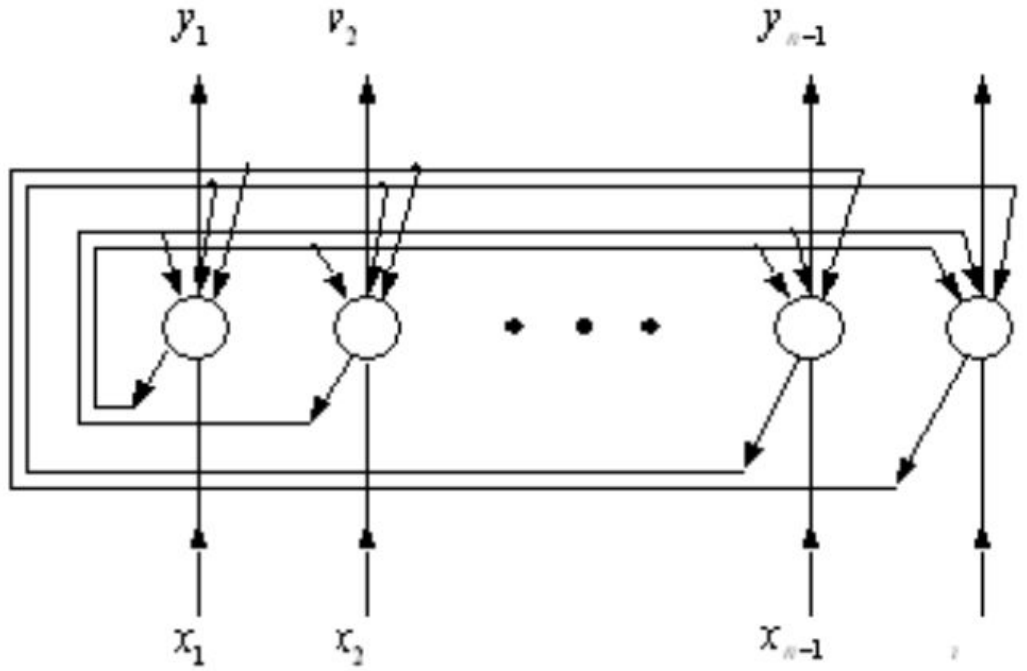




Hopfieldova síť

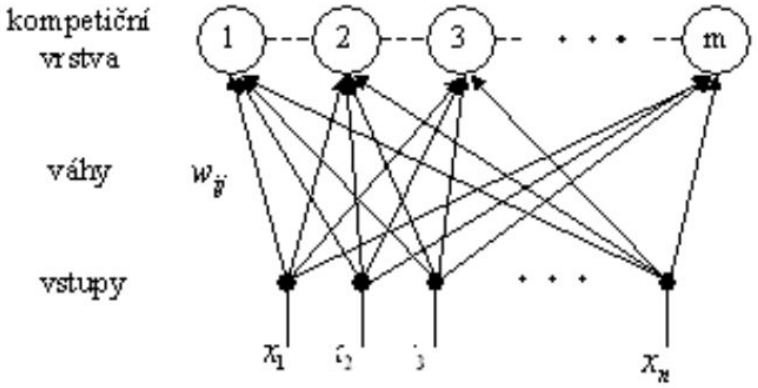
- Jednovrstvá, rekurentní (zpětnovazební) síť
- Zapamatování vstupních vzorů, vybavování
- Výstup neuronu je připojen na vstupy všech ostatních kromě jeho samotného.
- Použití: asociativní paměť,

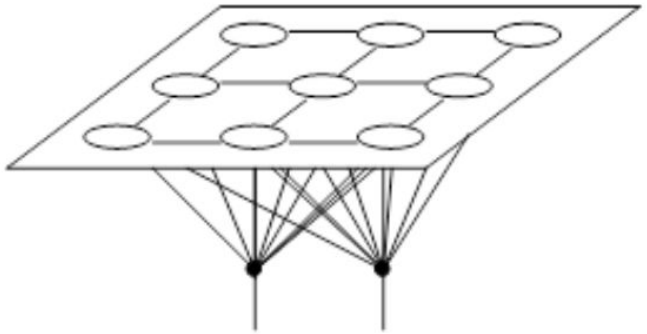
  klasifikátor, optimalizace

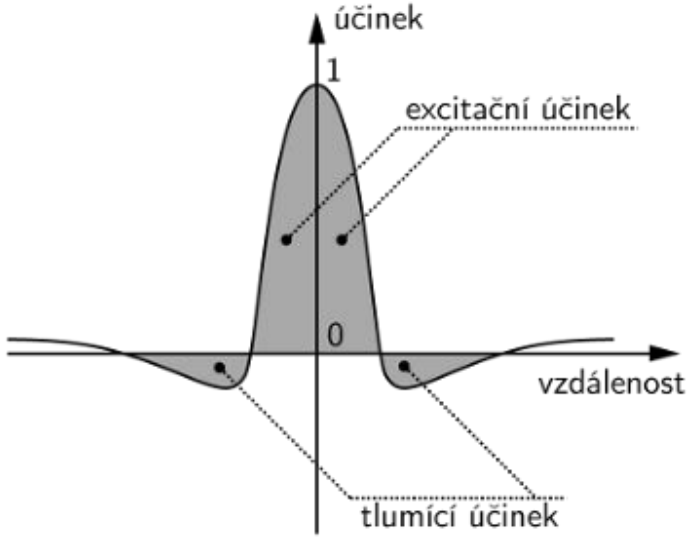


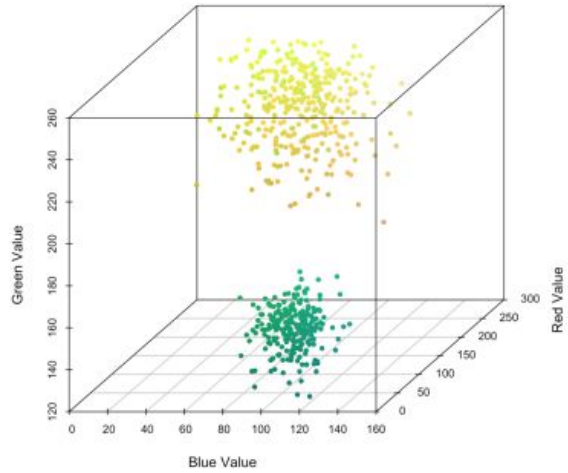
Samoorganizující se neuronové sít / Kohonenova (SOM - Self Organizing Map)

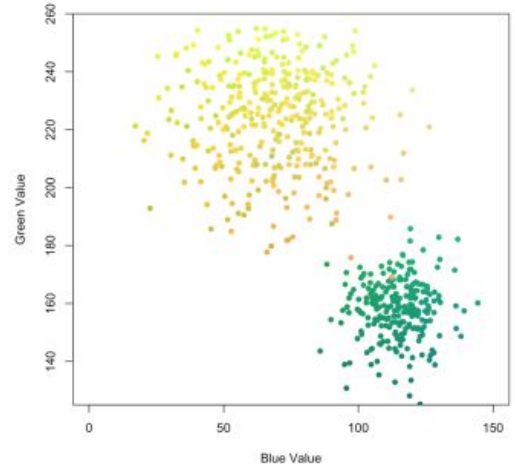
- jednovrstvá síť s dopředným šířením,
- učící algoritmus ve variantě "učení bez učitele" provádí pouze analýzu vstupních dat (shlukovou analýzu)
- neuronová síť obsahuje jedinou vrstvu neuronů (mřížka)

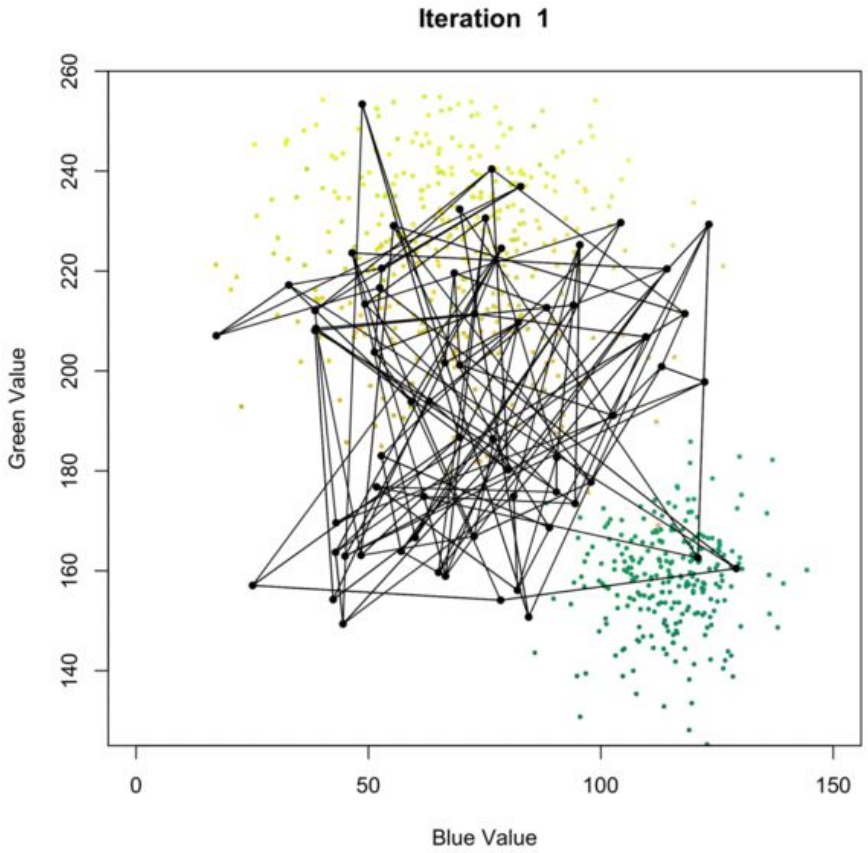


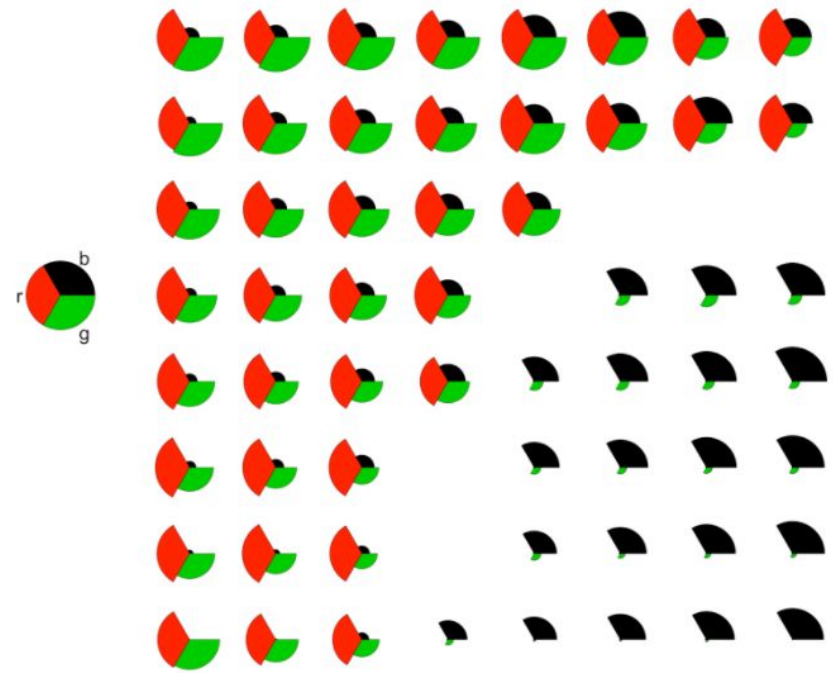





Green Value
Blue Value
Red Value

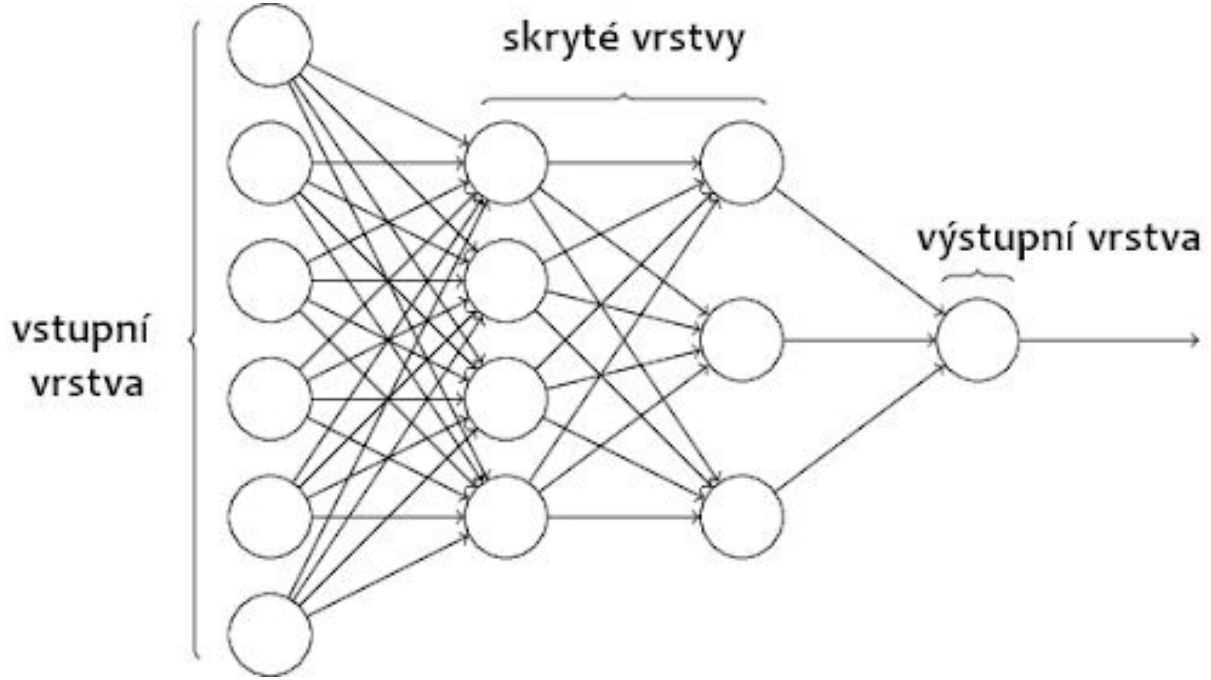

Green Value
Blue Value


Iteration 1
Green Value
Blue Value


b
r
g

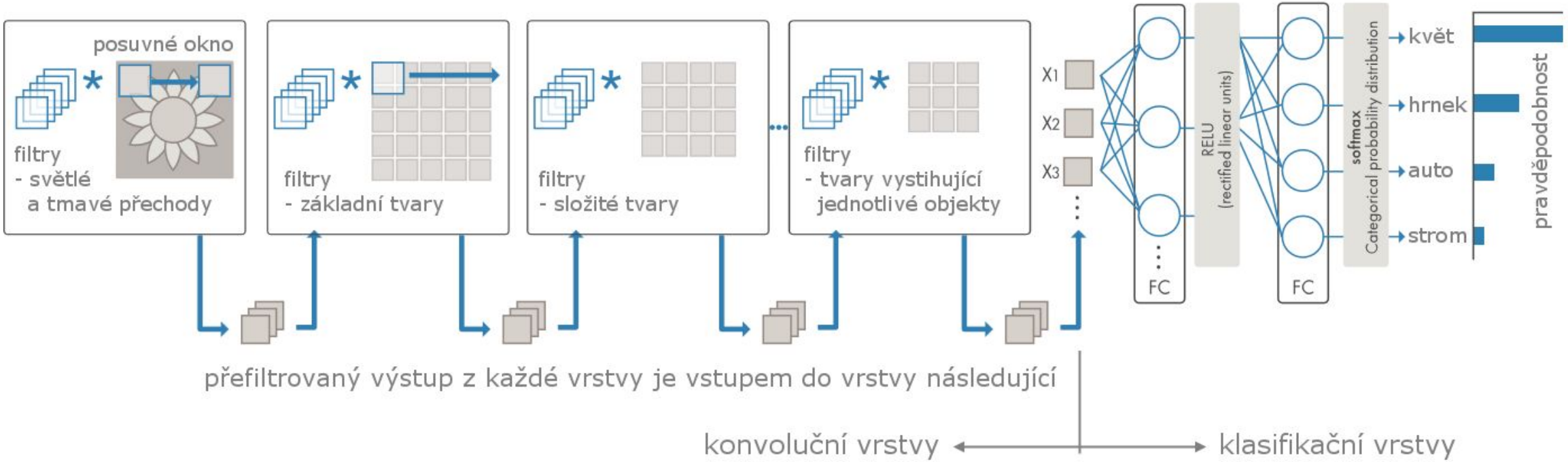
Vícevrstevná neuronová síť (MLP – Multi Layer Perceptron)

- Vstupní – Skrytá – Výstupní vrstva
- Nejpoužívanější neuronová síť
- Použití: predikce, klasifikace, aproximace



Konvoluční neuronová síť

- Vícevrstevná síť
- Vstupní – Skrytá – Výstupní vrstva
- Použití: klasifikace



Konvoluce

Obrázek 14.11. Zaostření

0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0

Konvoluce

Obrázek 14.12. Rozmazání

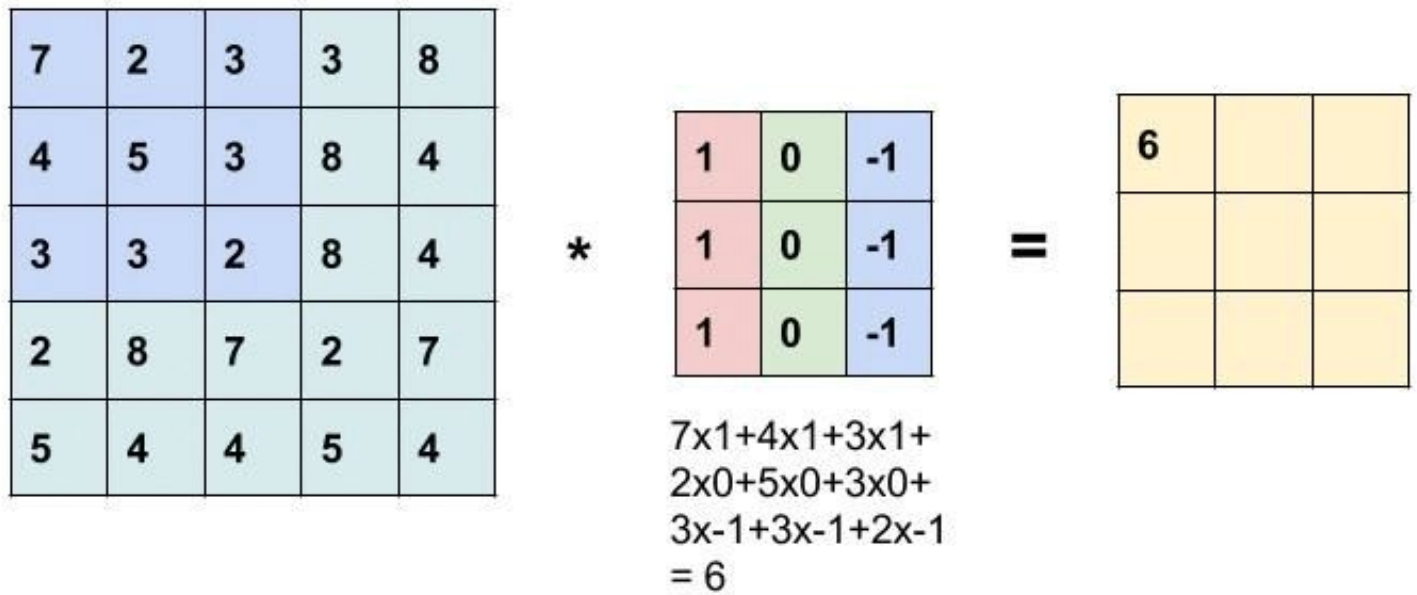
0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

Konvoluce

Obrázek 14.14. Detekce hran

	0	1	0	
	1	-4	1	
	0	1	0	

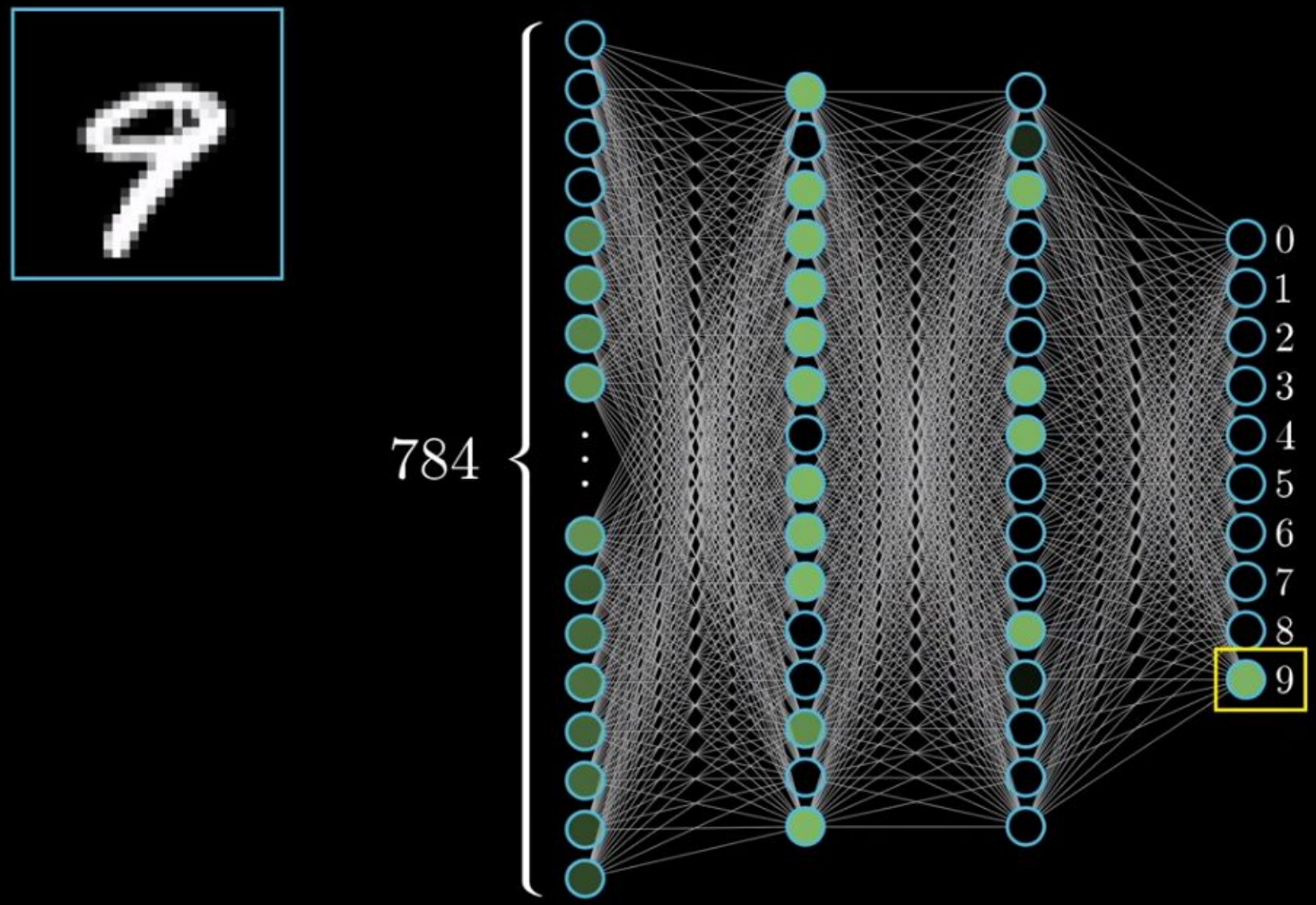
Konvoluce

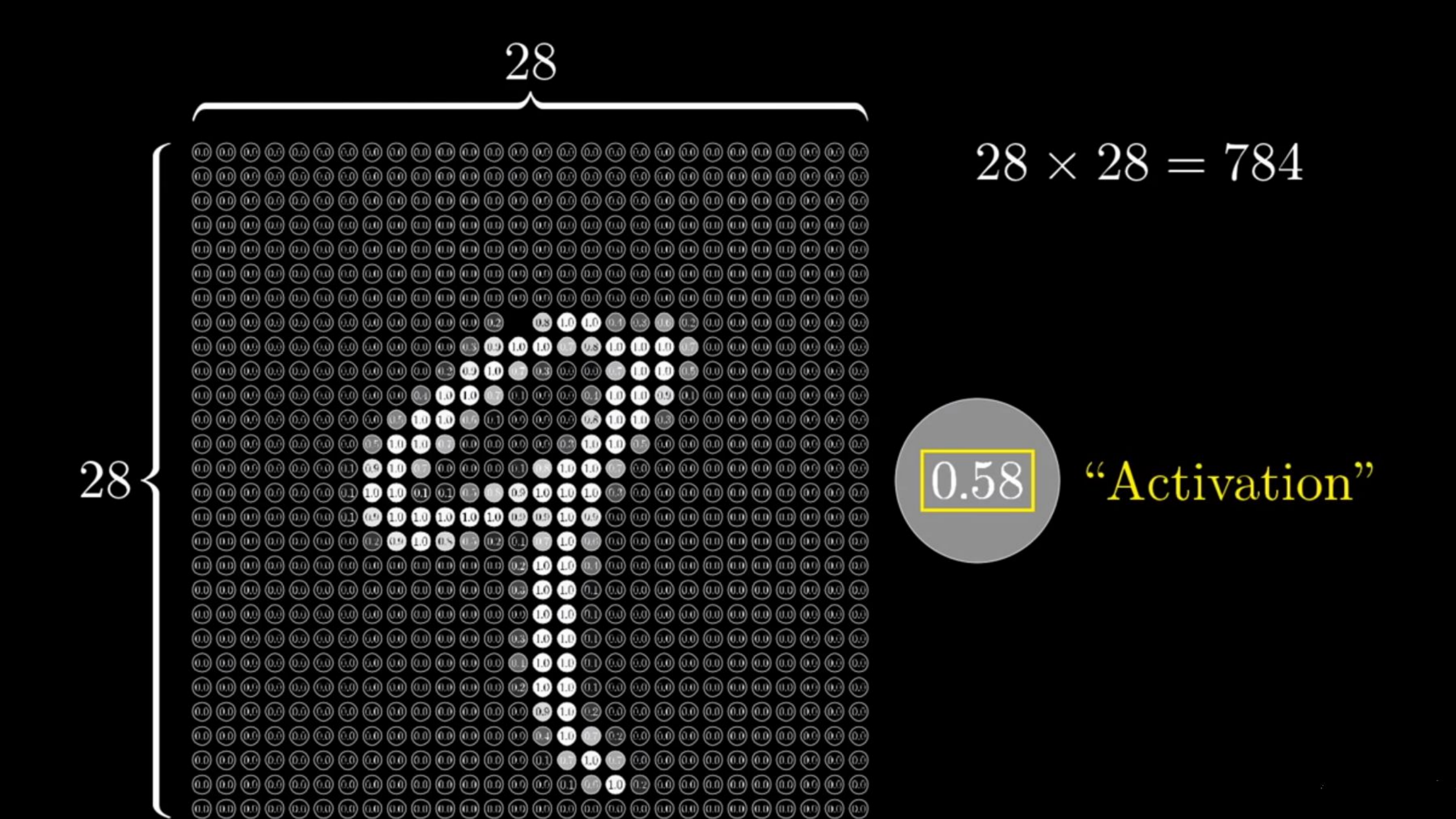

7 2 3 3 8
4 5 3 8 4
3 3 2 8 4
2 8 7 2 7
5 4 4 5 4
*
1 0 -1
1 0 -1
1 0 -1
=
6
7x1+4x1+3x1+
2x0+5x0+3x0+
3x-1+3x-1+2x-1
= 6

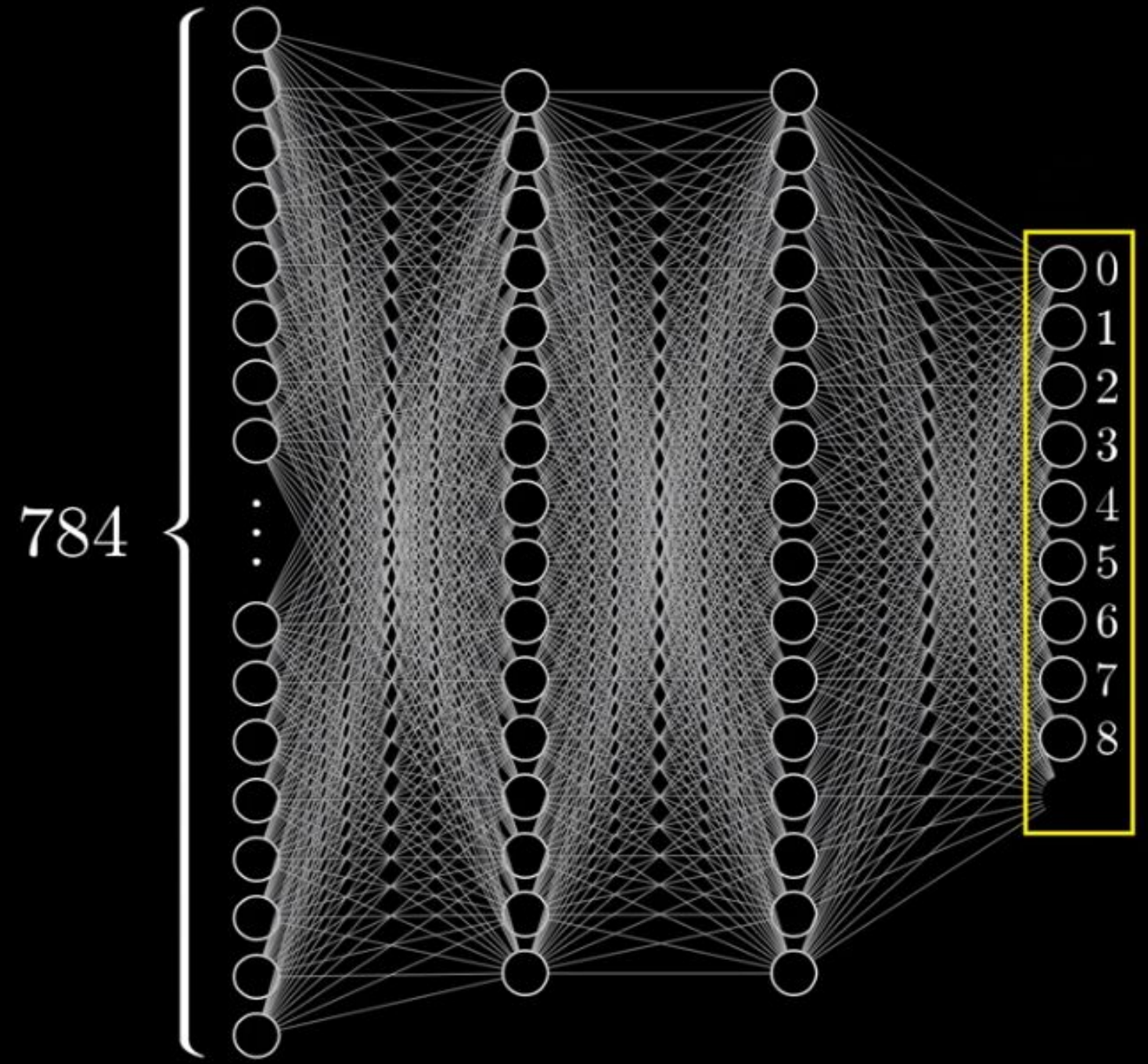
Vytváření neuronové sítě

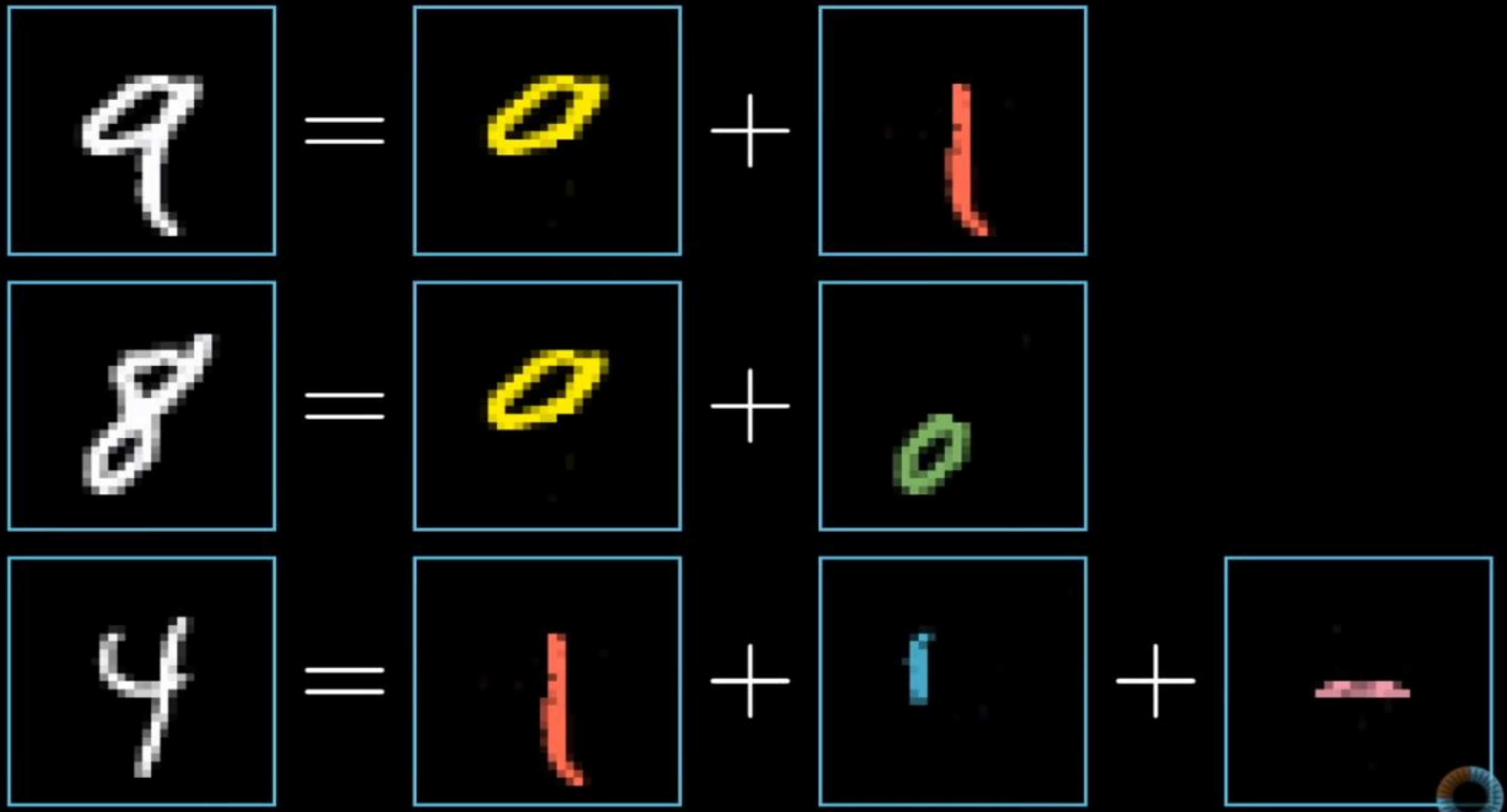
1. Inicializace parametrů (váhy, bias)
2. Definování aktivační funkce (Sigmoid, ReLU)
3. Definování chybové funkce (MSE, CE)
4. Dopředné šíření (Forward Propagation)
5. Počítání chyby
6. Zpětné šíření chyby (Back Propagation)
7. Úprava vah
8. Nová iterace (zpět na krok č. 4)

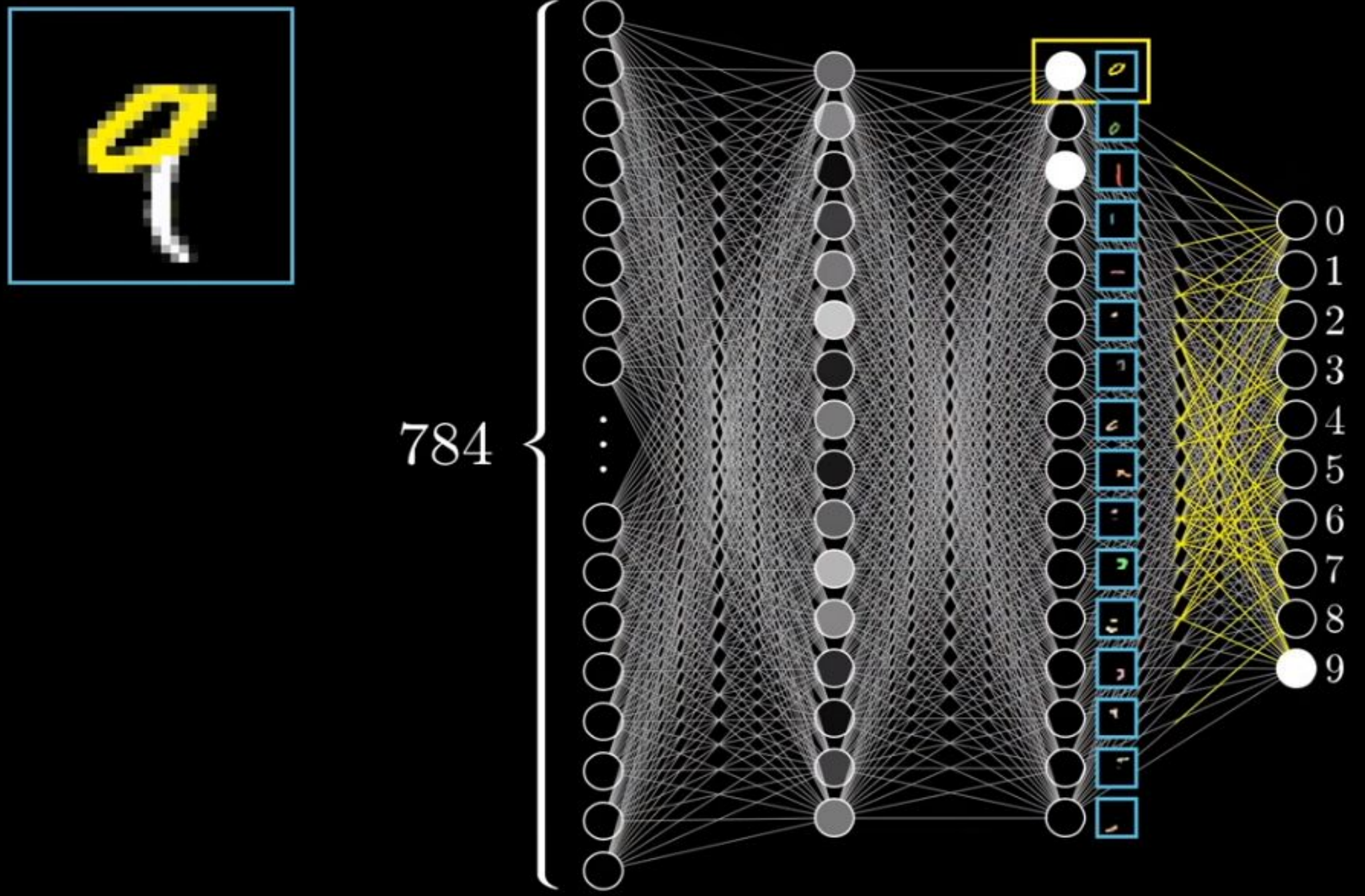
Fungování neuronové sítě

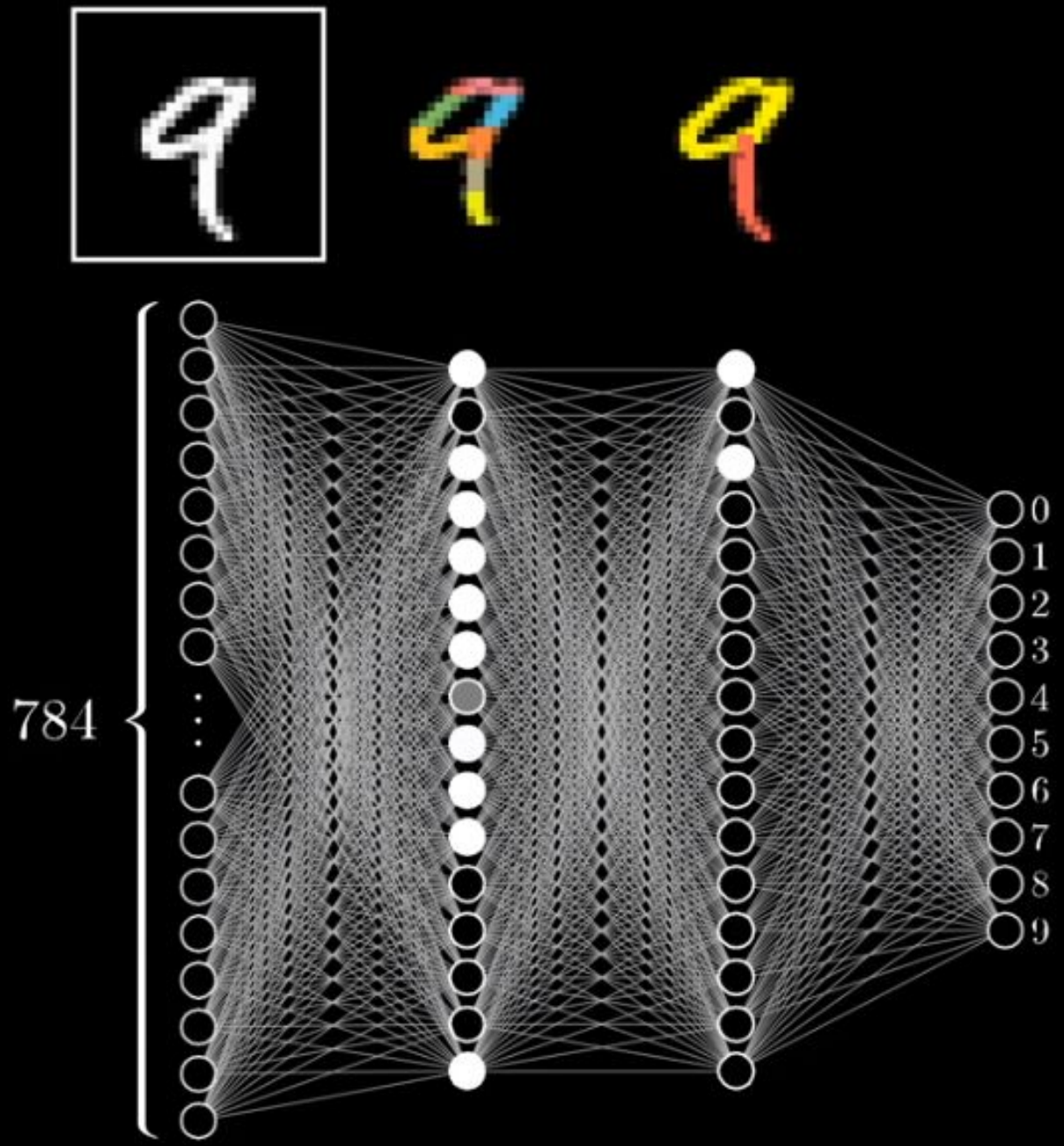



28
28
$28 \times 28 = 784$
0.58
“Activation”

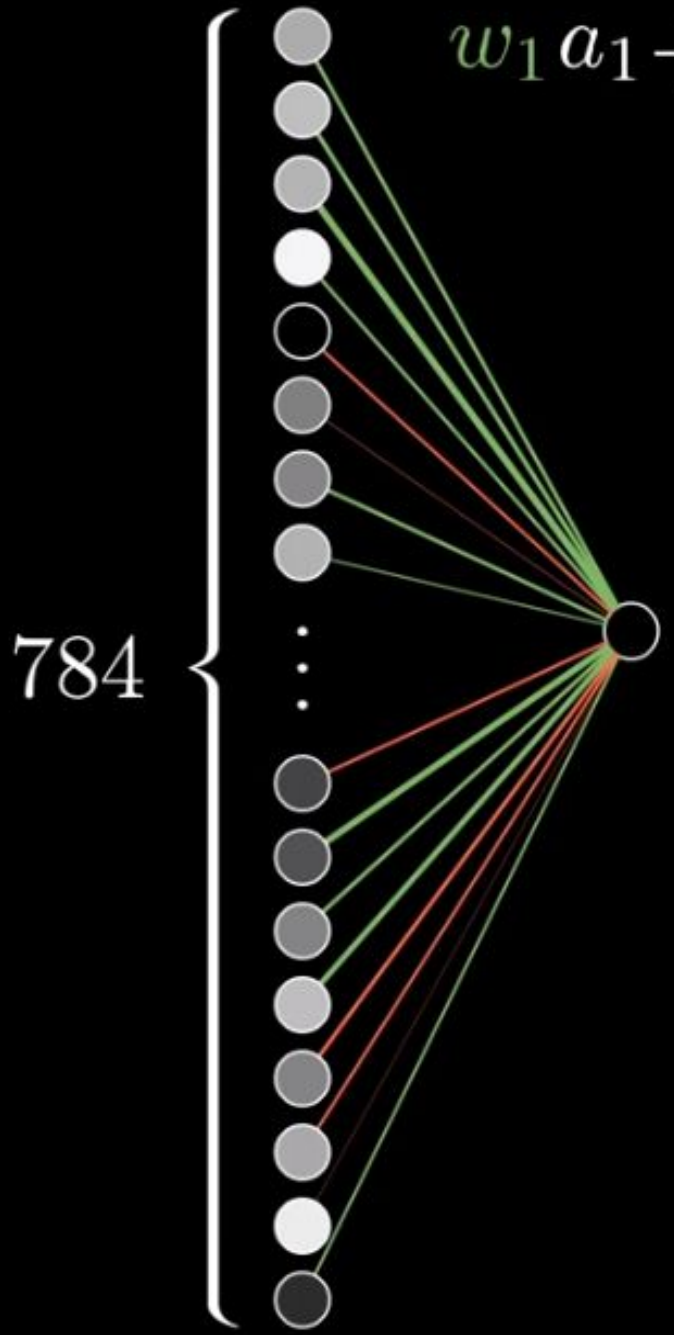

784
0
1
2
3
4
5
6
7
8
0.97
9



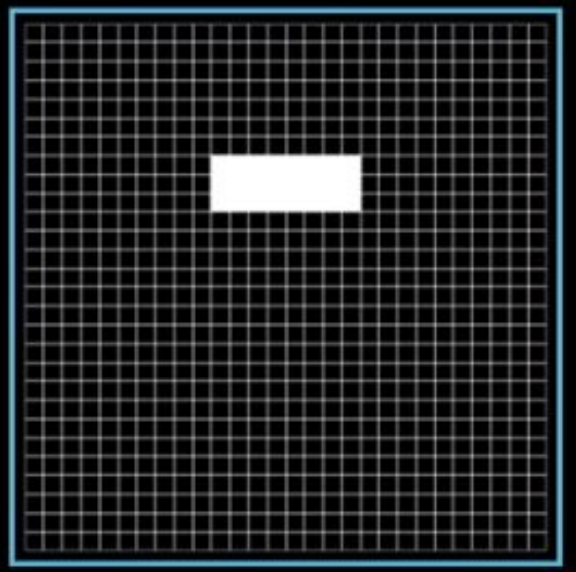

784
0
1
2
3
4
5
6
7
8
9


784
0
1
2
3
4
5
6
7
8
9

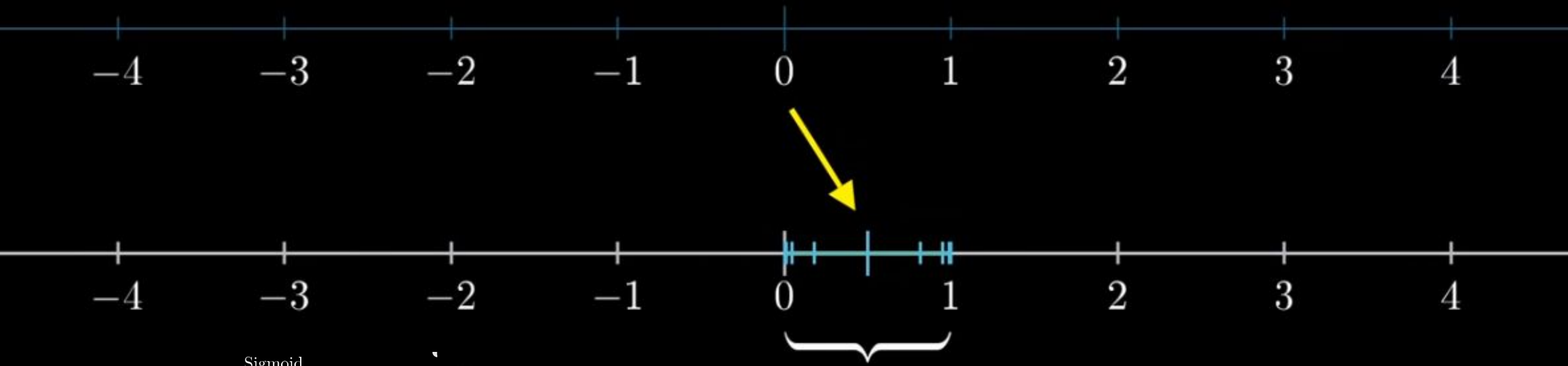
$$w_1 a_1 + w_2 a_2 + w_3 a_3 + w_4 a_4 + \cdots + w_n a_n$$



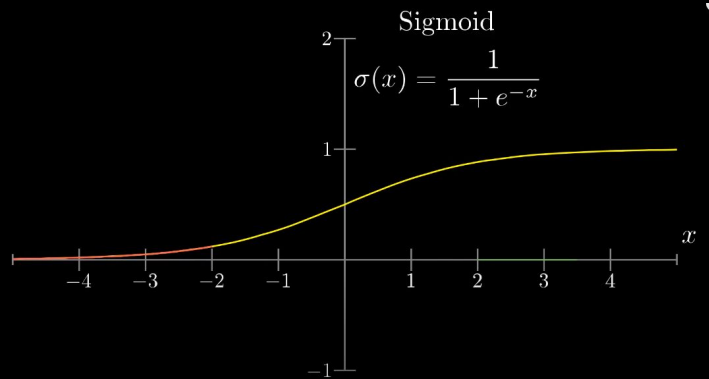
w_1: 2.07
w_2: 2.31
w_3: 3.64
w_4: 1.87
w_5: −1.51
w_6: −0.43
w_7: 2.01
w_8: 1.07
⋮

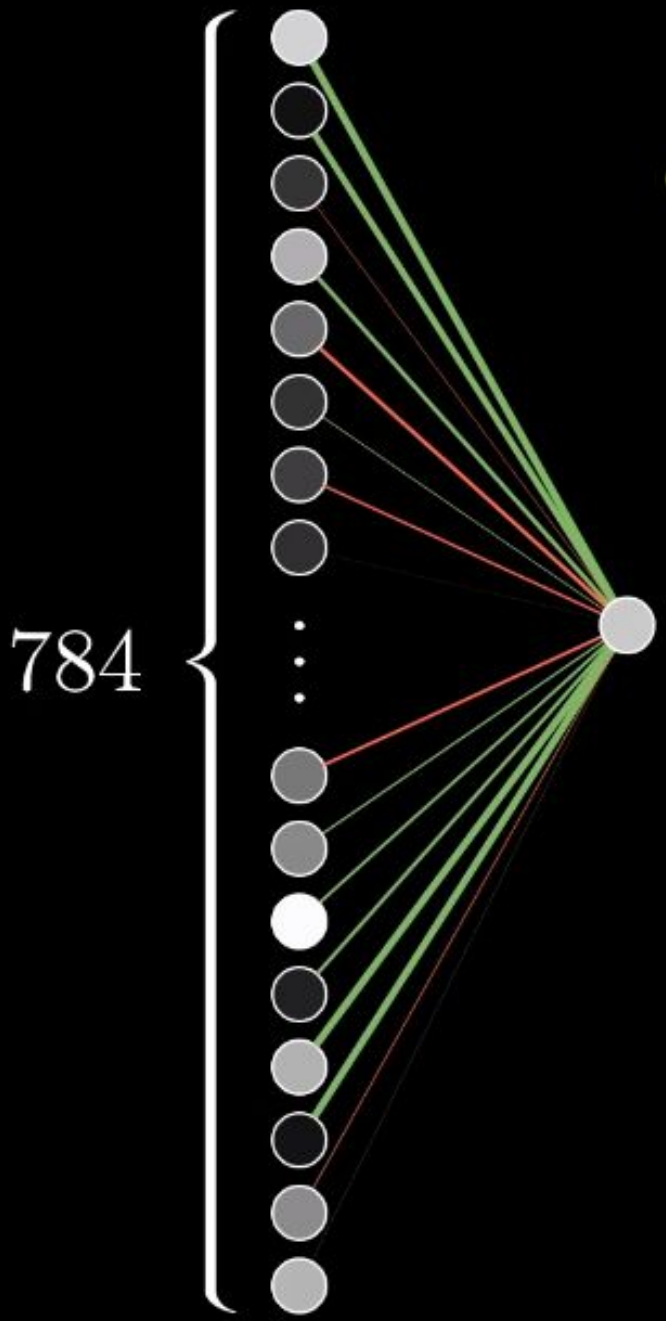


$$w_1 a_1 + w_2 a_2 + w_3 a_3 + w_4 a_4 + \cdots + w_n a_n$$



Activations should be in this range

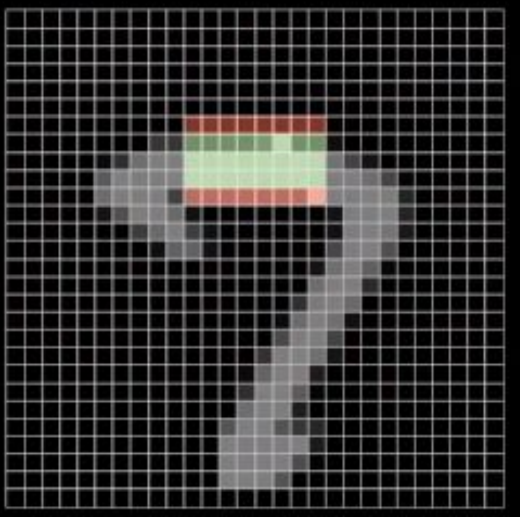


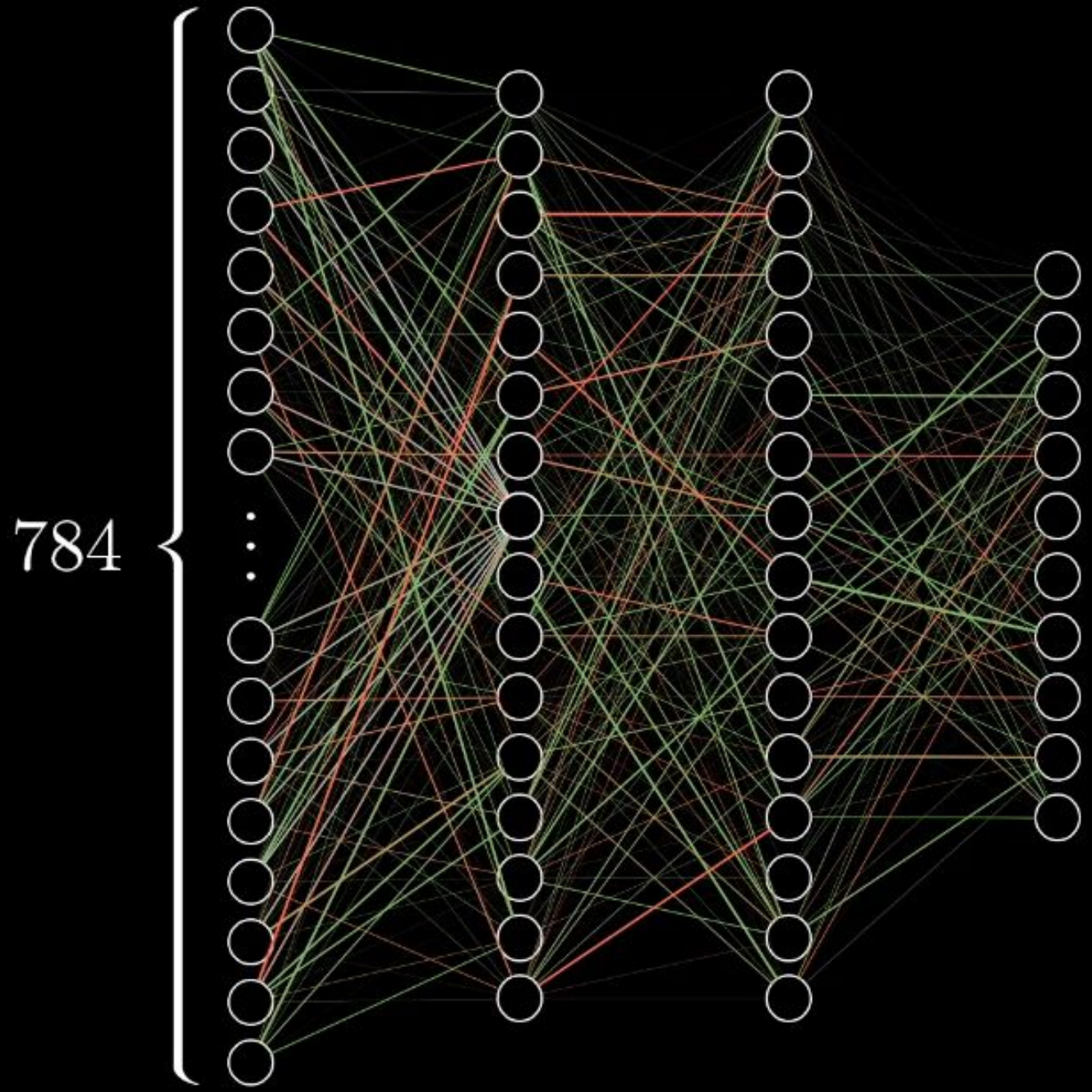


$$\sigma\left(w_1 a_1 + w_2 a_2 + w_3 a_3 + \cdots + w_n a_n \boxed{-10}\right)$$

"bias"

Only activate meaningfully when weighted sum > 10



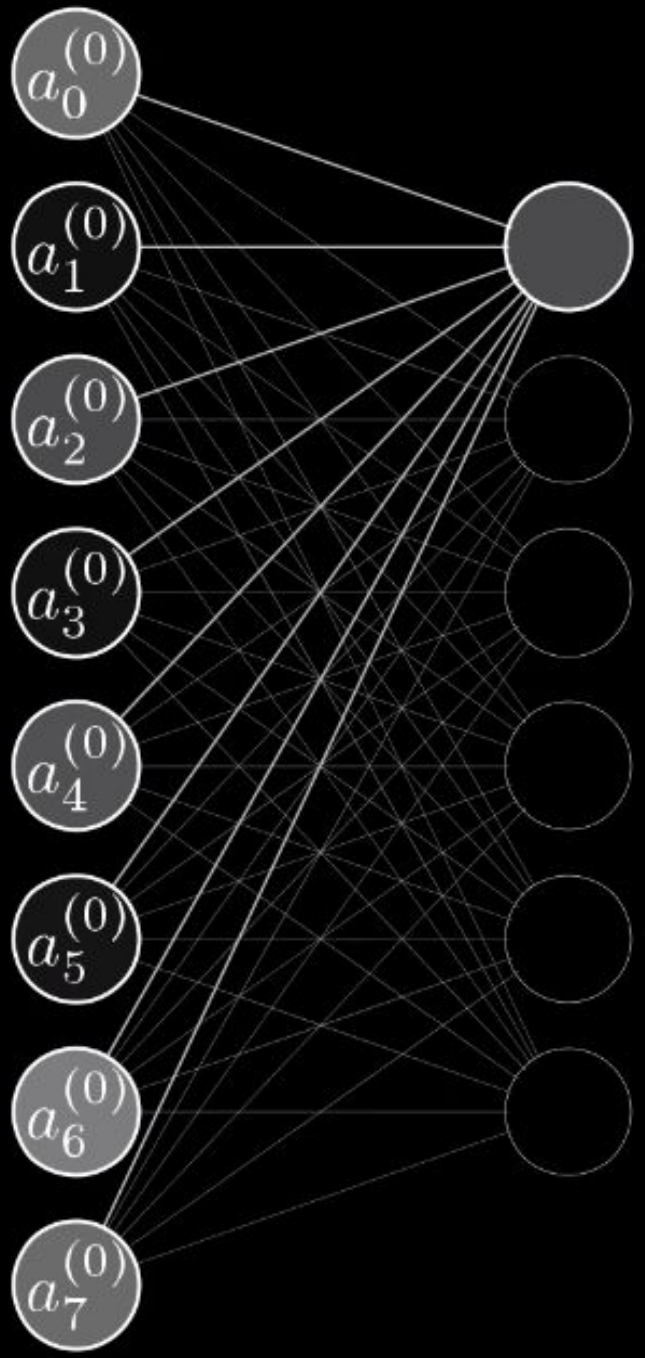


$$784 \times 16 + 16 \times 16 + 16 \times 10$$
weights

$$16 + 16 + 10$$
biases

$$13{,}002$$

Learning $\rightarrow$ Finding the right weights and biases



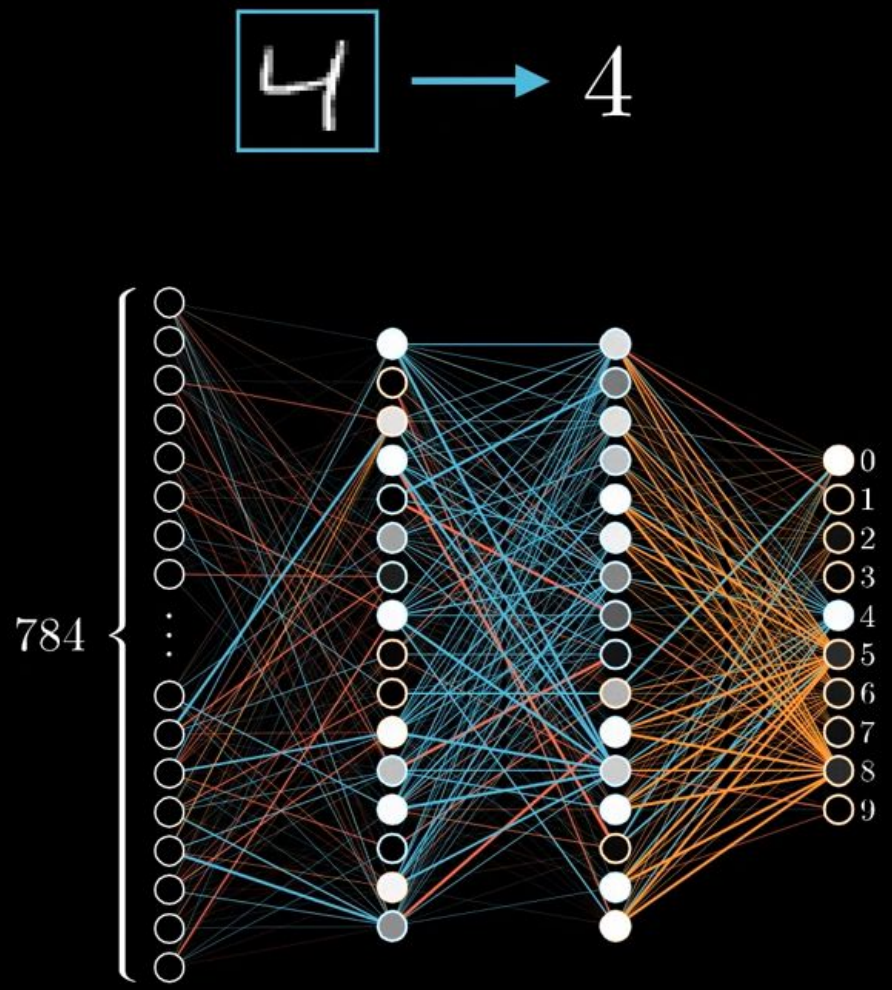
Sigmoid

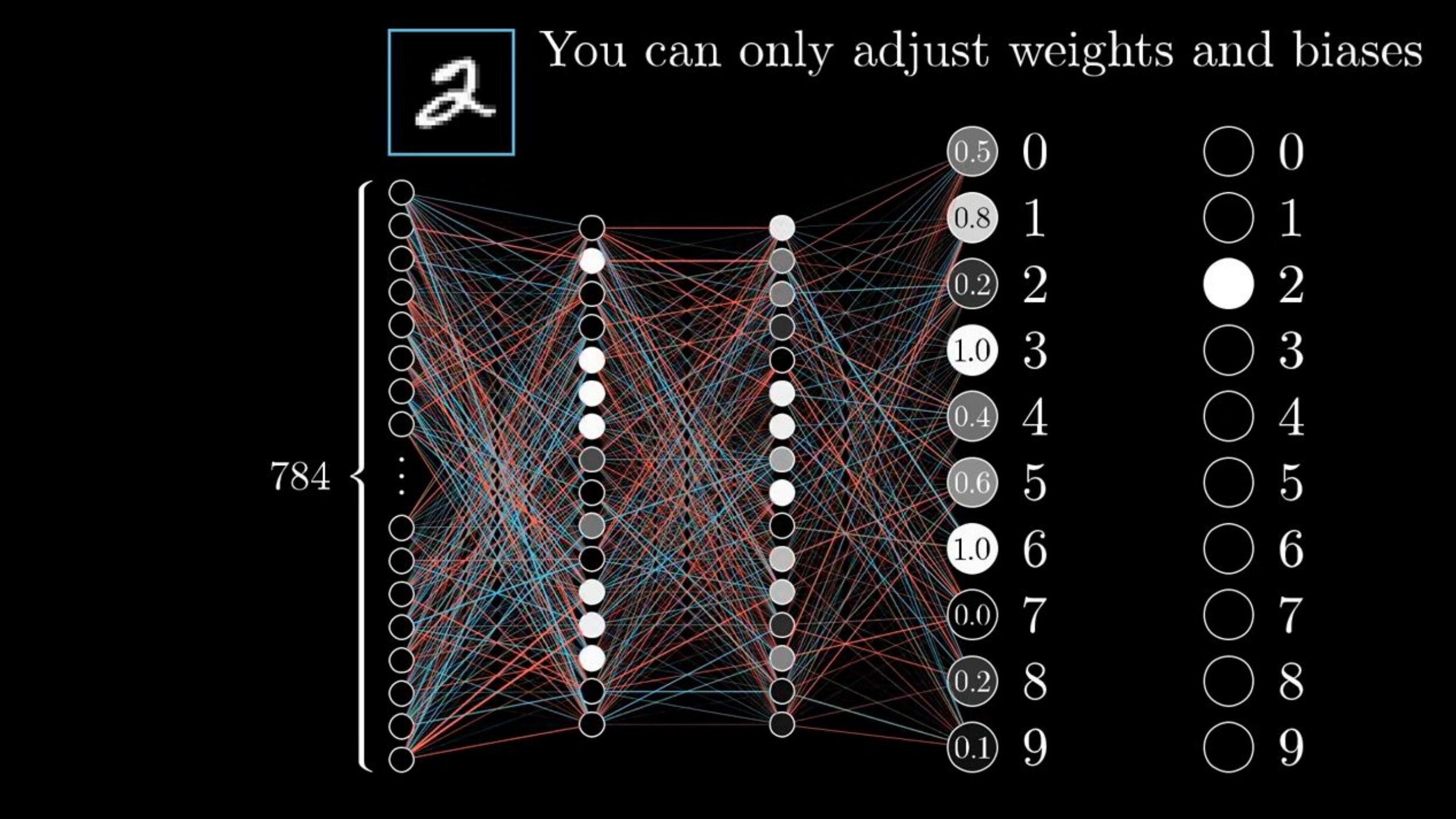
$$a_0^{(1)} = \sigma\left(w_{0,0}\, a_0^{(0)} + w_{0,1}\, a_1^{(0)} + \cdots + w_{0,n}\, a_n^{(0)} + b_0 \right)$$

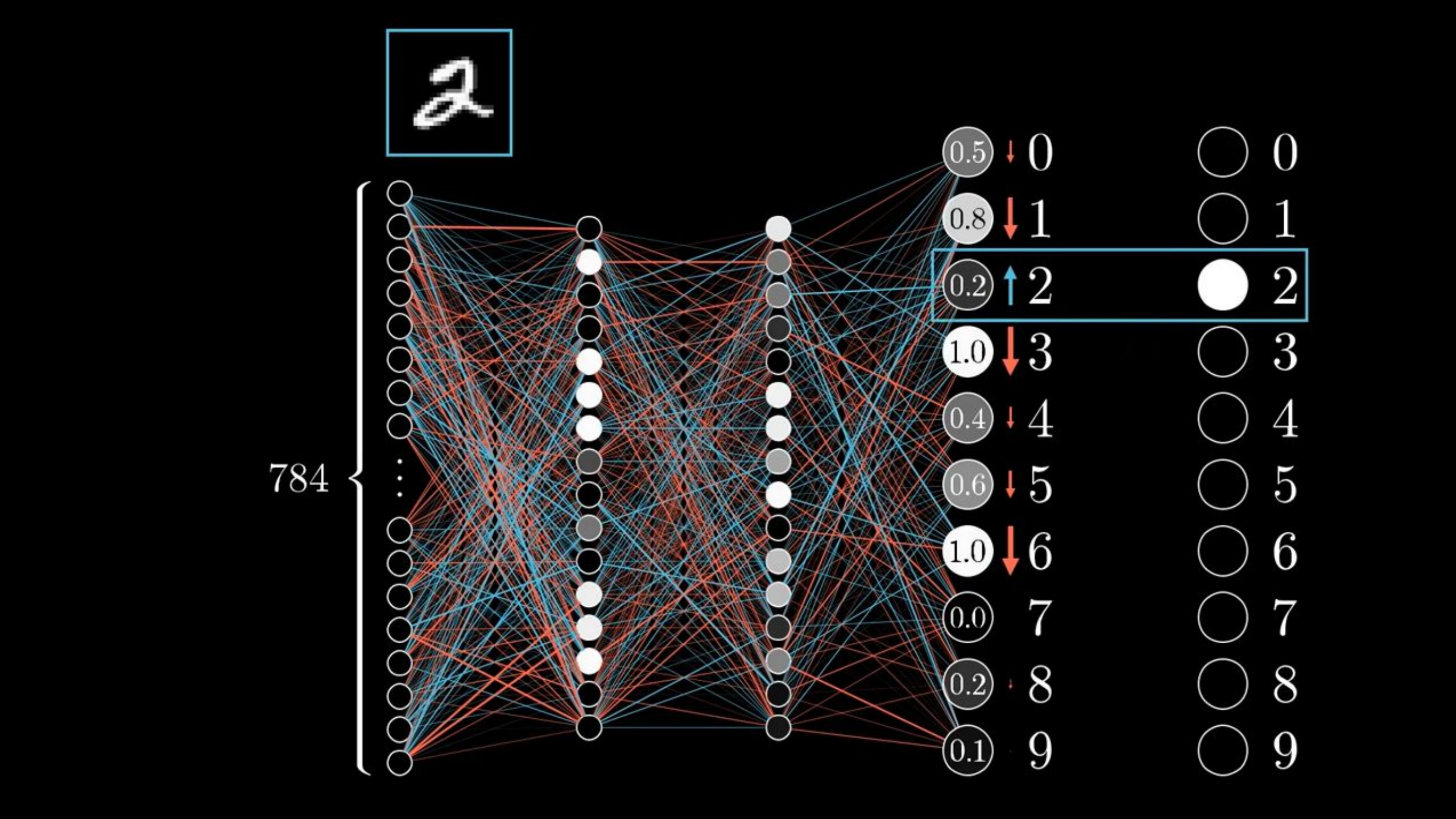
Bias

$$\sigma\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix} \right)$$

Učení Neuronové sítě




You can only adjust weights and biases
784
0.5
0.8
0.2
1.0
0.4
0.6
1.0
0.0
0.2
0.1
0
1
2
3
4
5
6
7
8
9
0
1
2
3
4
5
6
7
8
9

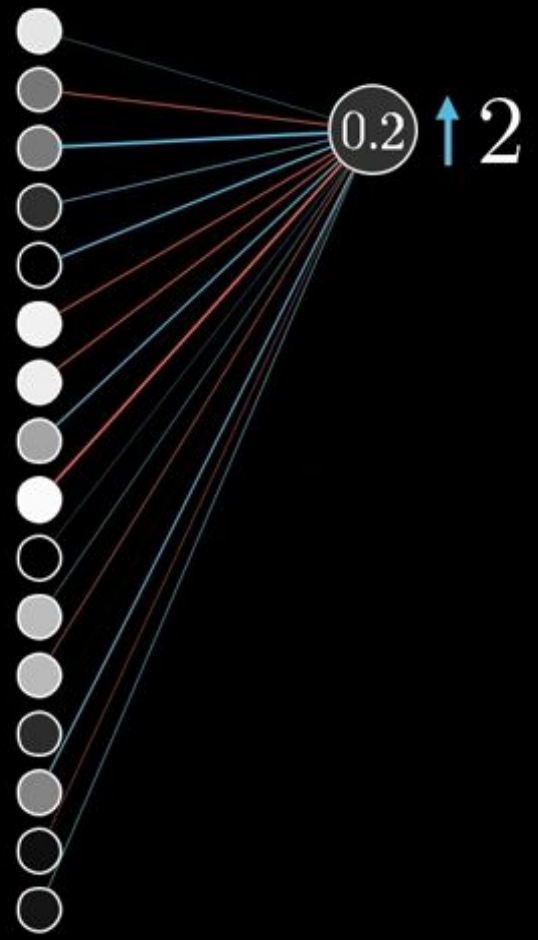

784
0.5 0
0.8 1
0.2 2
1.0 3
0.4 4
0.6 5
1.0 6
0.0 7
0.2 8
0.1 9
0
1
2
3
4
5
6
7
8
9

$$0.2 = \sigma(w_0 a_0 + w_1 a_1 + \cdots + w_{n-1} a_{n-1} + b)$$

Increase b

Increase w_i

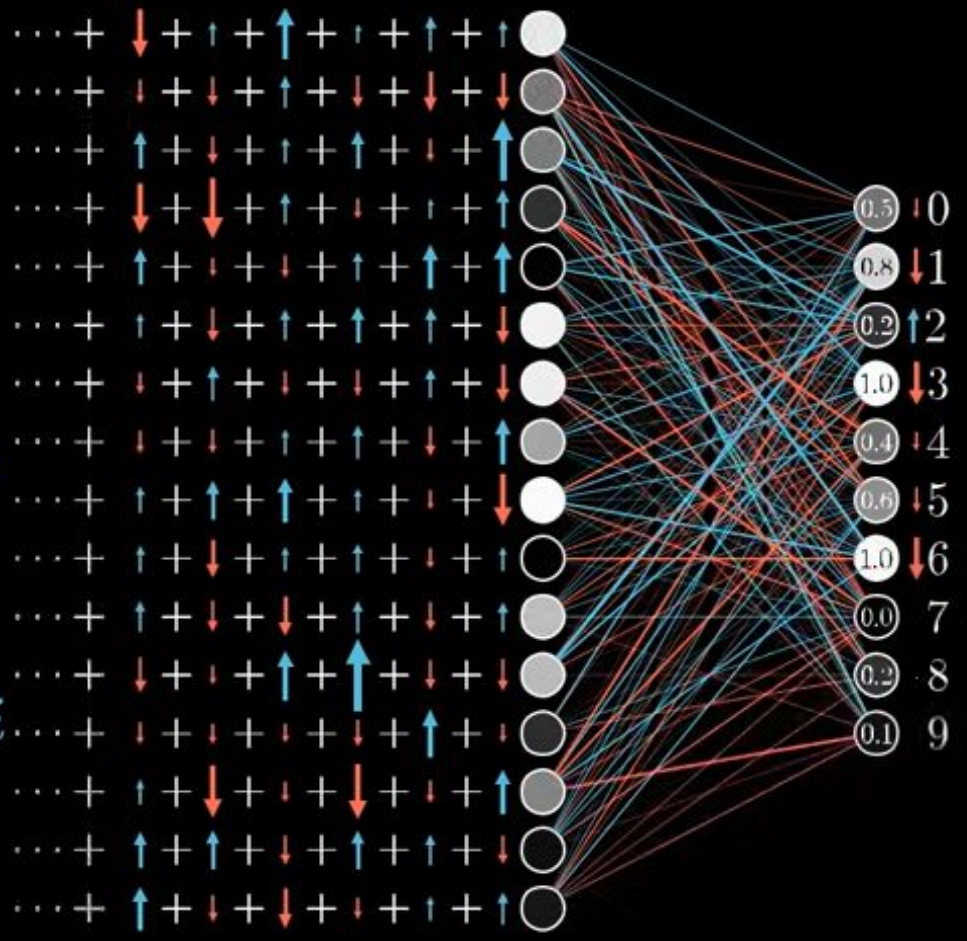
Change a_i



Increase b

Increase w_i
in proportion to a_i

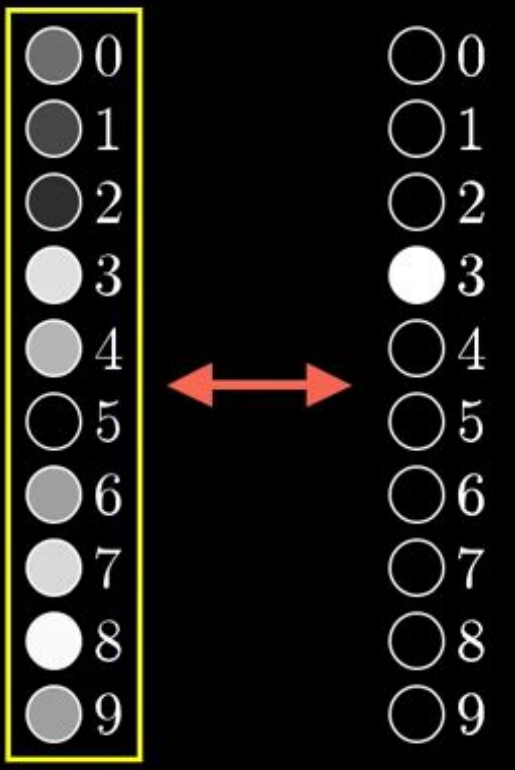
Change a_i
in proportion to w_i



Cost of 

$$
\begin{aligned}
3.37 \Big\{ \quad & 0.1863 \leftarrow (0.43 - 0.00)^2 + \\
& 0.0809 \leftarrow (0.28 - 0.00)^2 + \\
& 0.0357 \leftarrow (0.19 - 0.00)^2 + \\
& 0.0138 \leftarrow (0.88 - 1.00)^2 + \\
& 0.5242 \leftarrow (0.72 - 0.00)^2 + \\
& 0.0001 \leftarrow (0.01 - 0.00)^2 + \\
& 0.4079 \leftarrow (0.64 - 0.00)^2 + \\
& 0.7388 \leftarrow (0.86 - 0.00)^2 + \\
& 0.9817 \leftarrow (0.99 - 0.00)^2 + \\
& 0.3998 \leftarrow (0.63 - 0.00)^2
\end{aligned}
$$

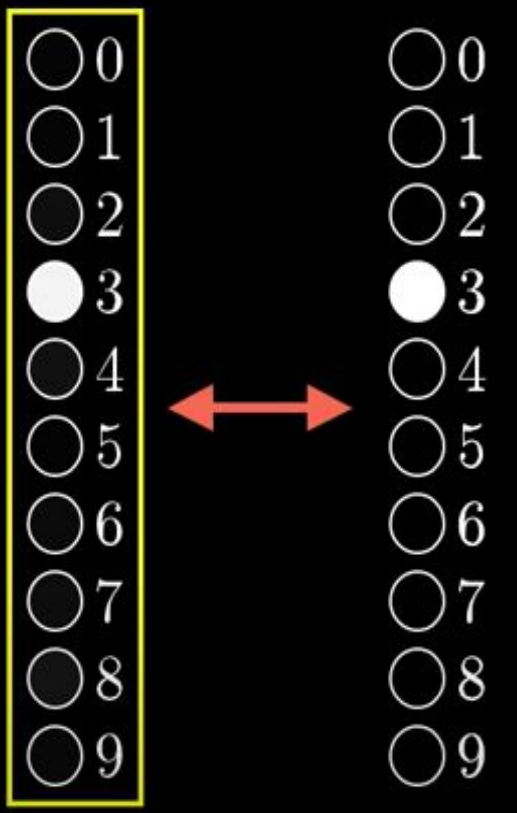
What's the "cost" of this difference?

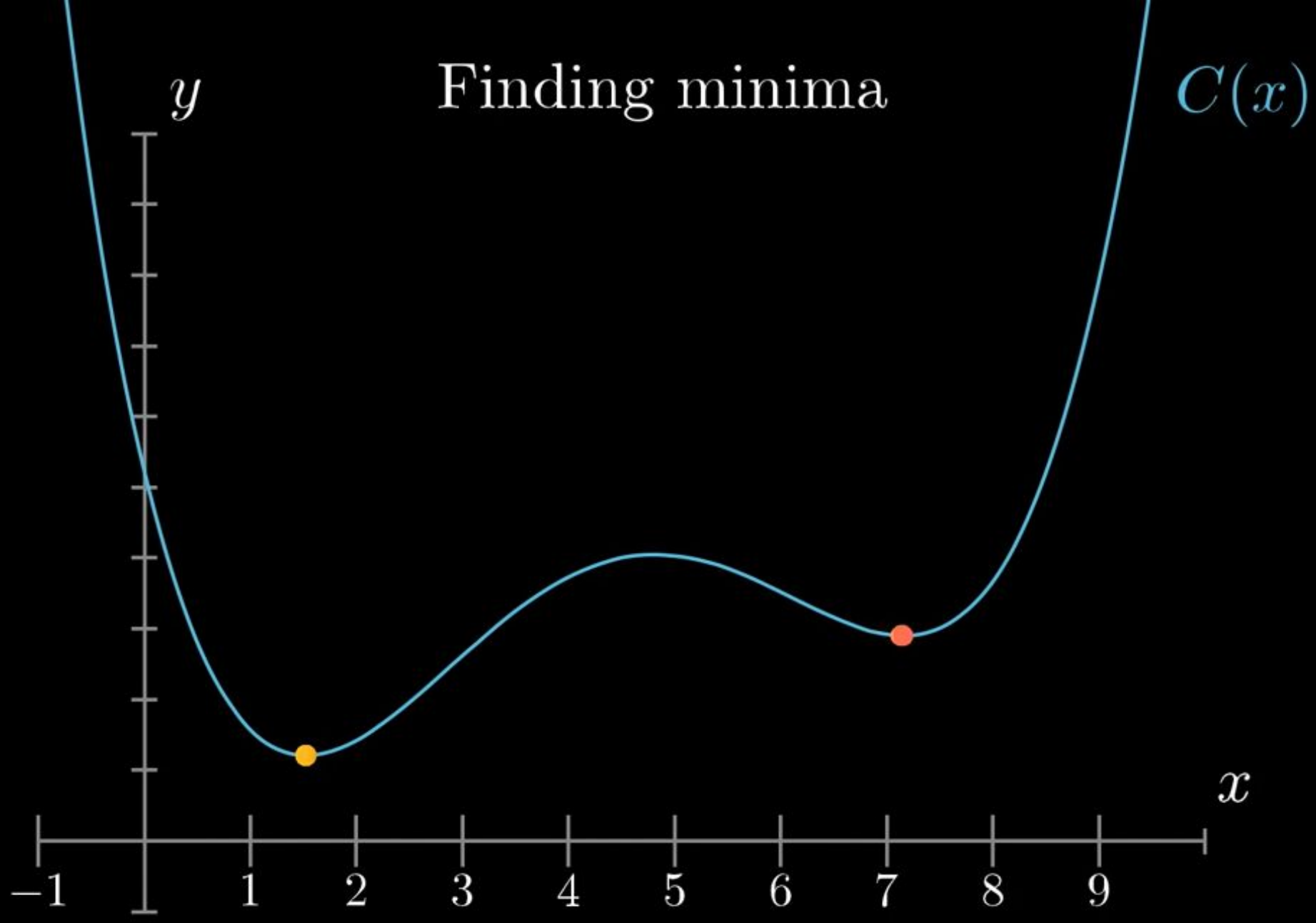


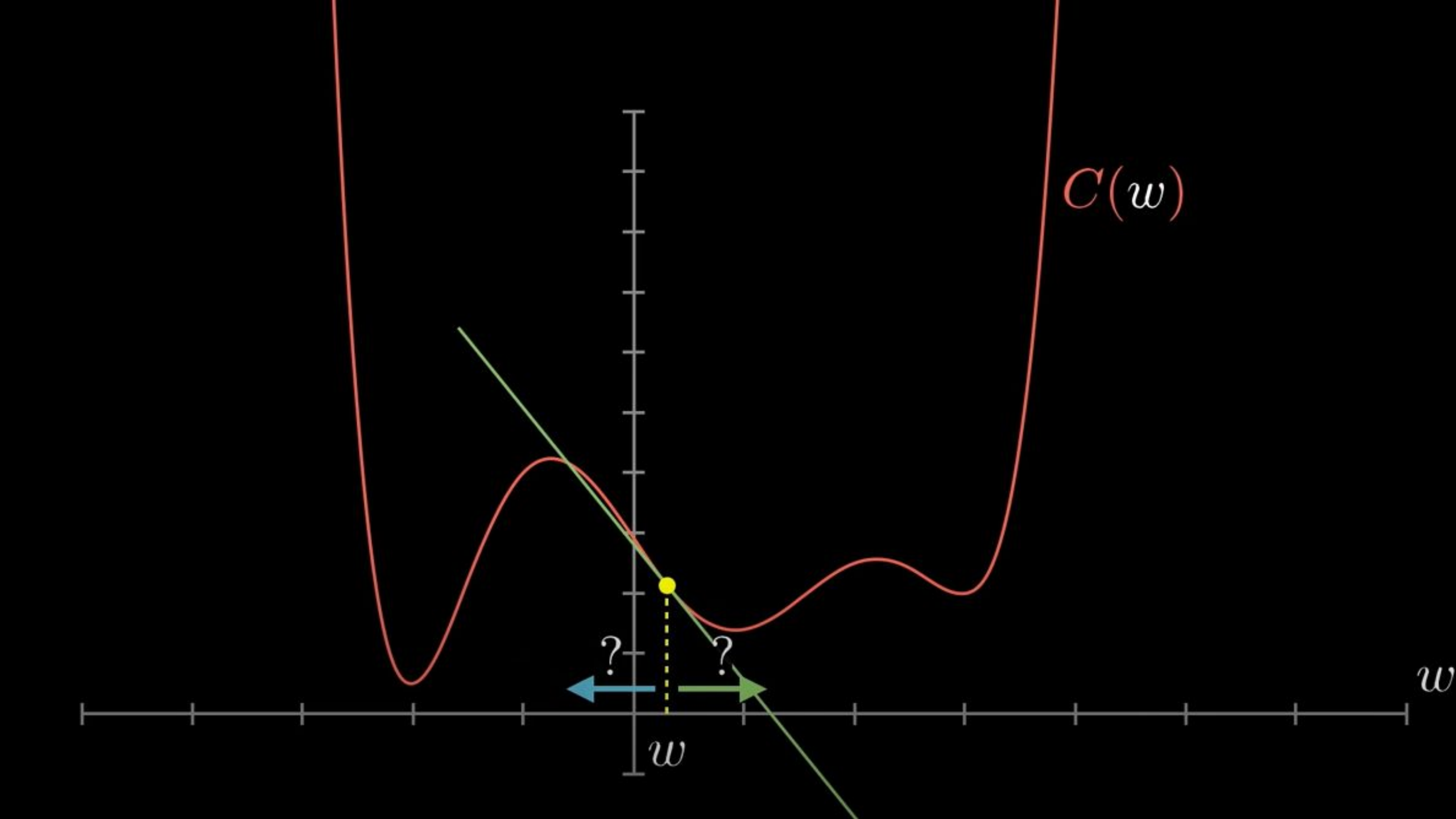
Cost of 

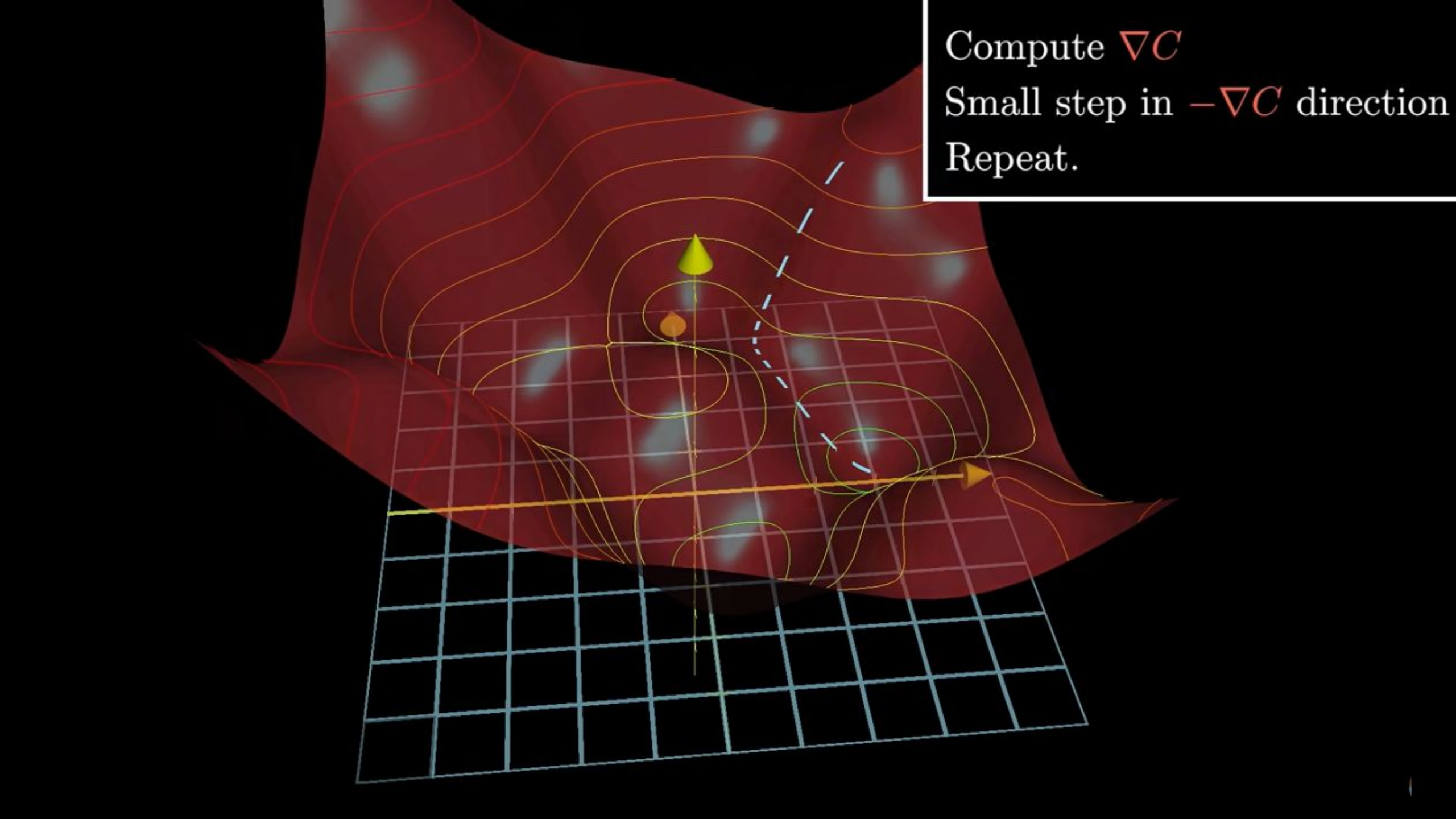
$$
\begin{aligned}
0.0006 &\leftarrow (0.02 - 0.00)^2 + \\
0.0007 &\leftarrow (0.03 - 0.00)^2 + \\
0.0039 &\leftarrow (0.06 - 0.00)^2 + \\
0.0009 &\leftarrow (0.97 - 1.00)^2 + \\
0.0055 &\leftarrow (0.07 - 0.00)^2 + \\
0.0004 &\leftarrow (0.02 - 0.00)^2 + \\
0.0022 &\leftarrow (0.05 - 0.00)^2 + \\
0.0033 &\leftarrow (0.06 - 0.00)^2 + \\
0.0072 &\leftarrow (0.08 - 0.00)^2 + \\
0.0018 &\leftarrow (0.04 - 0.00)^2
\end{aligned}
\quad \Big\} \; 0.03
$$

What's the "cost" of this difference?




Finding minima
y
$C(x)$
x
-1
1
2
3
4
5
6
7
8
9


$C(w)$
?
?
w
w


Compute ∇C
Small step in $-\nabla C$ direction
Repeat.

$$\vec{\mathbf{W}} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{13,000} \\ w_{13,001} \\ w_{13,002} \end{bmatrix}$$

$$-\nabla C(\vec{\mathbf{W}}) = \begin{bmatrix} 0.31 \\ 0.03 \\ -1.25 \\ \vdots \\ 0.78 \\ -0.37 \\ 0.16 \end{bmatrix}$$

w_0 should increase somewhat
w_1 should increase a little
w_2 should decrease a lot

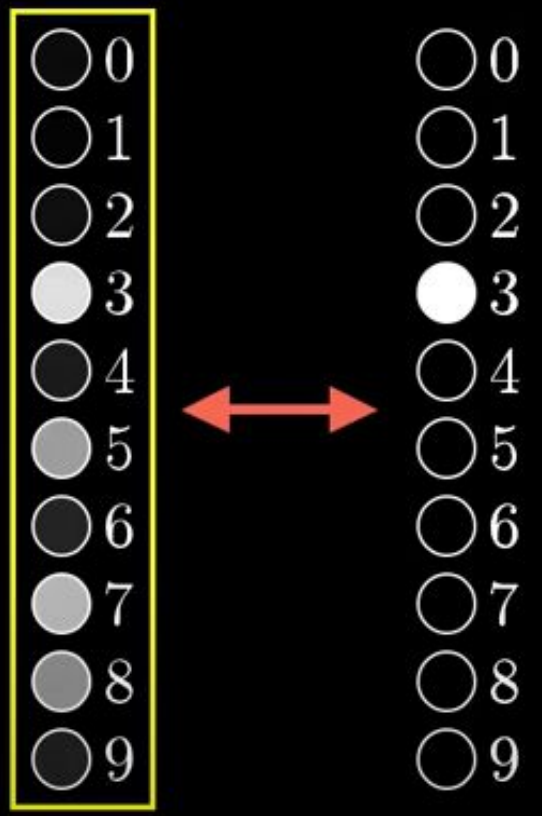
$w_{13,000}$ should increase a lot
$w_{13,001}$ should decrease somewhat
$w_{13,002}$ should increase a little

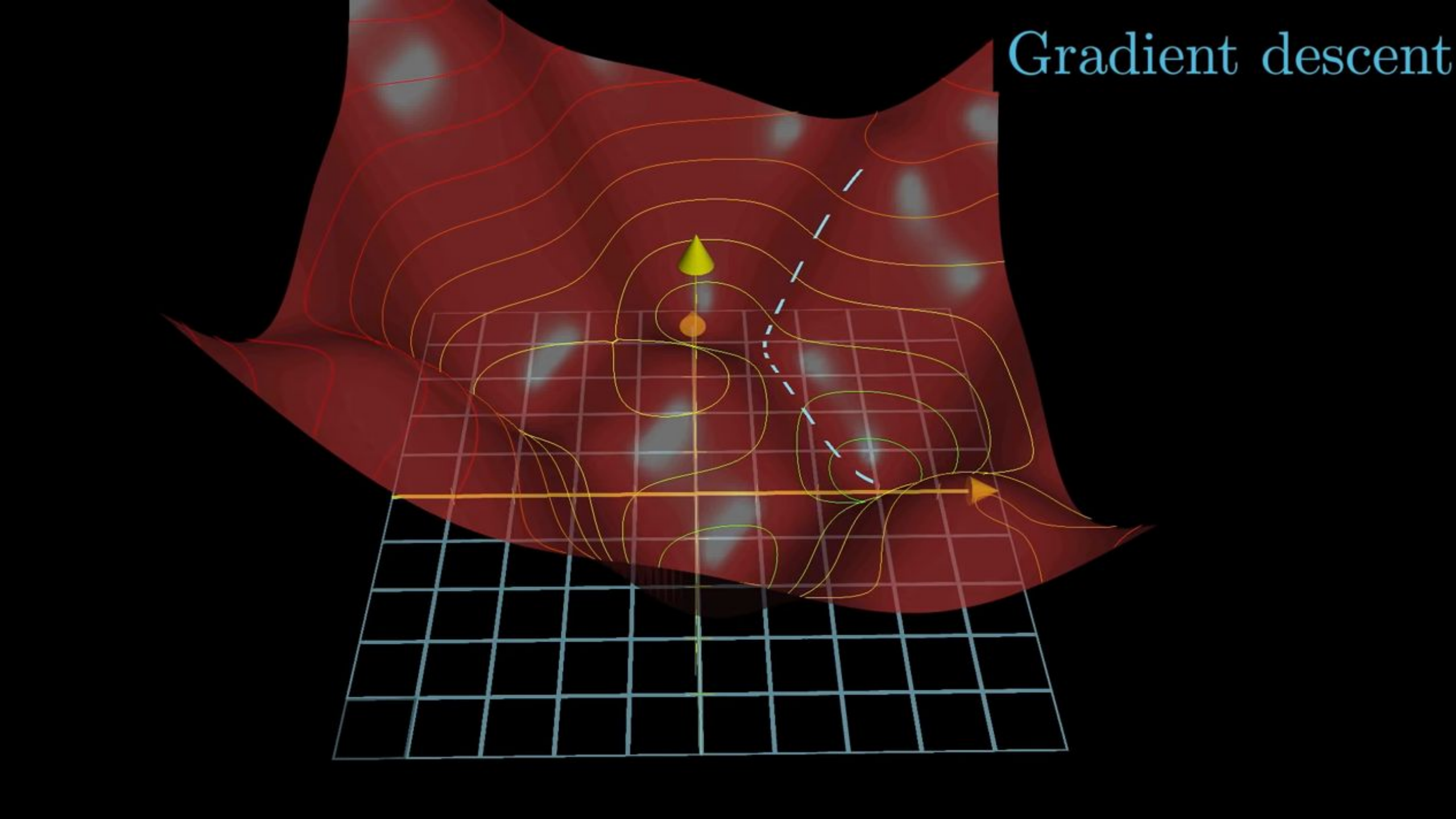
Minimize cost . . .

Cost of 

$$
\begin{aligned}
&(0.05 - 0.00)^2 + \\
&(0.02 - 0.00)^2 + \\
&(0.07 - 0.00)^2 + \\
&(0.88 - 1.00)^2 + \\
&(0.11 - 0.00)^2 + \\
&(0.63 - 0.00)^2 + \\
&(0.14 - 0.00)^2 + \\
&(0.73 - 0.00)^2 + \\
&(0.57 - 0.00)^2 + \\
&(0.13 - 0.00)^2
\end{aligned}
$$

What's the "cost" of this difference?

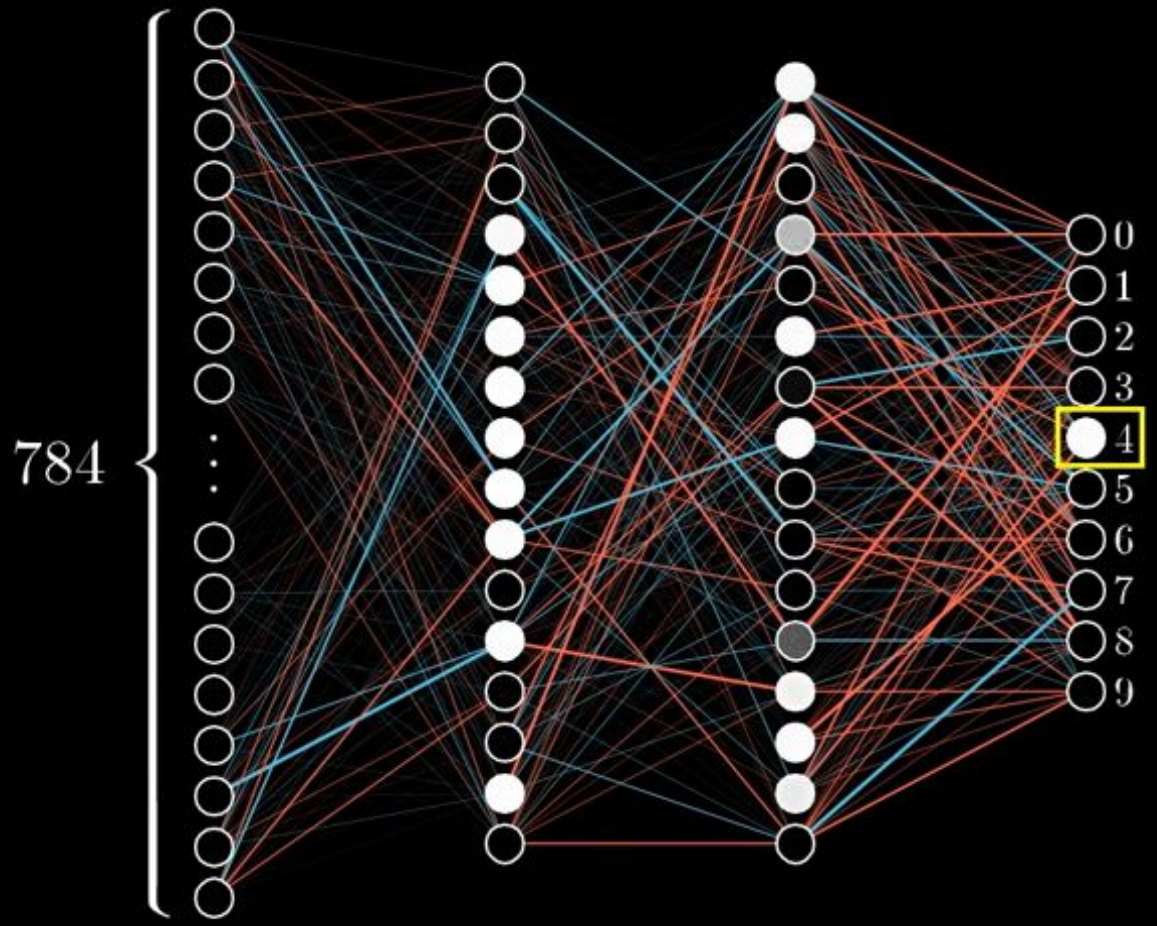



Gradient descent

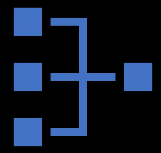
Testing data

Guess → 4

$$\frac{\text{Number correct}}{\text{total}} = \frac{238}{249} = 0.956$$



Algoritmus zpětného šíření (Back propagation)



Vstup – Skryté vrstvy – Výstup

Výpočet chyby

Zpětné šíření chyby

Úprava vah

Řetězové pravidlo (Chain rule)

- Pravidlo o derivaci složené funkce – zjednodušení výpočtu derivace
- Úprav vah: $w_{i+1} = wi - \eta \nabla G(wi)$

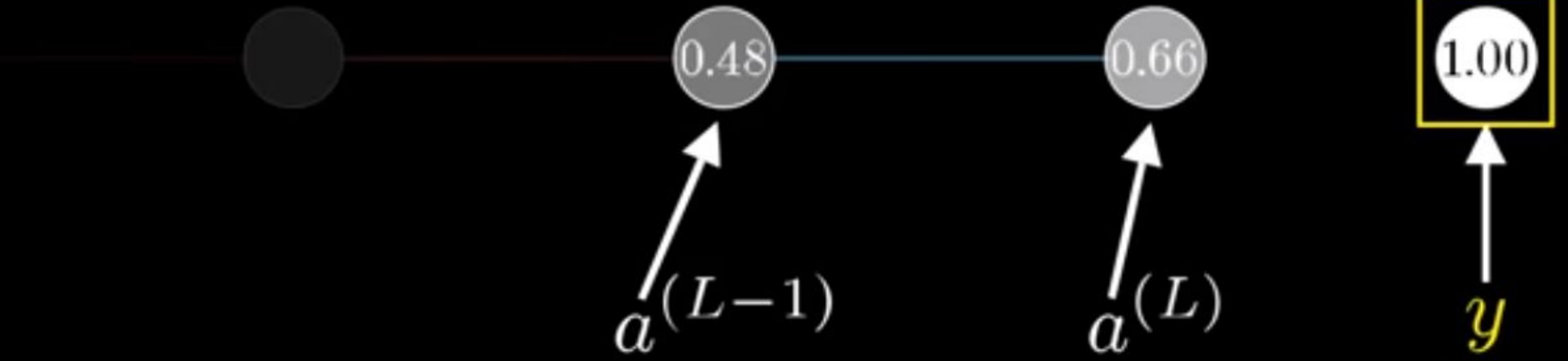
Back Propagation

$$C(w_1, b_1, w_2, b_2, w_3, b_3)$$

$$\text{Cost} \longrightarrow C_0(\dots) = (a^{(L)} - y)^2$$

$$z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$$

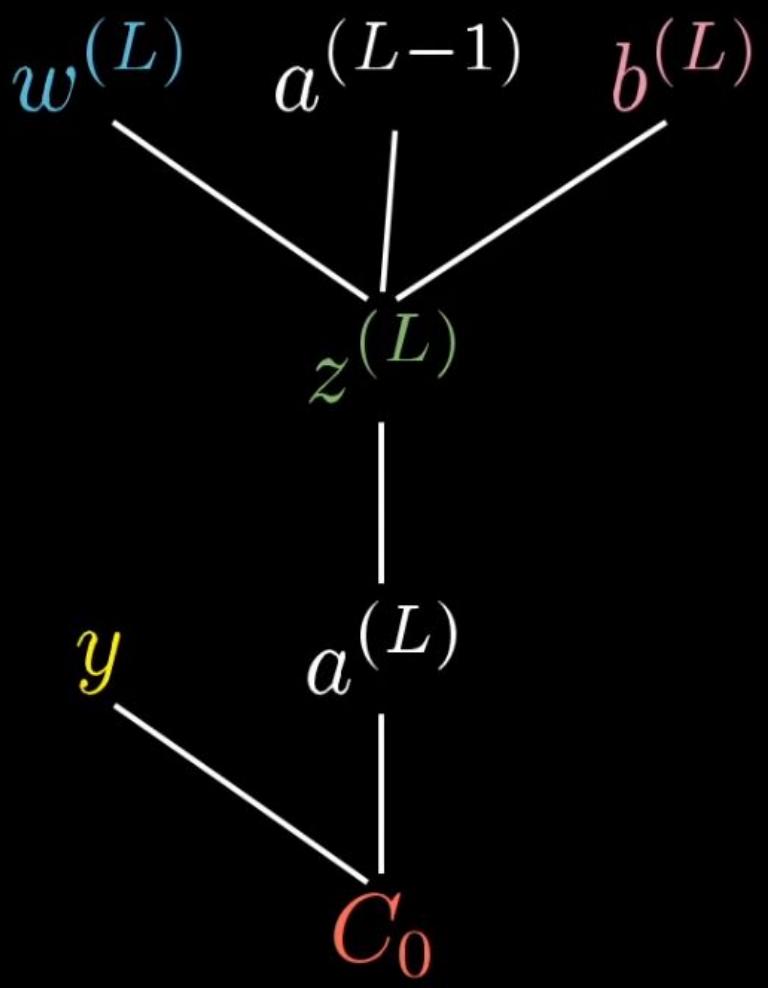
$$a^{(L)} = \sigma(z^{(L)})$$

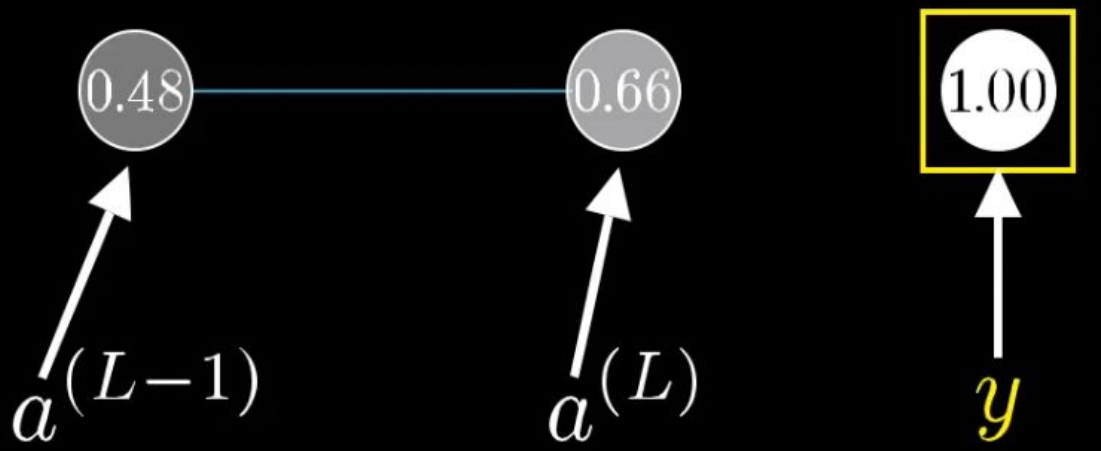


Cost $\longrightarrow C_0(\dots) = (a^{(L)} - y)^2$

$$z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$$

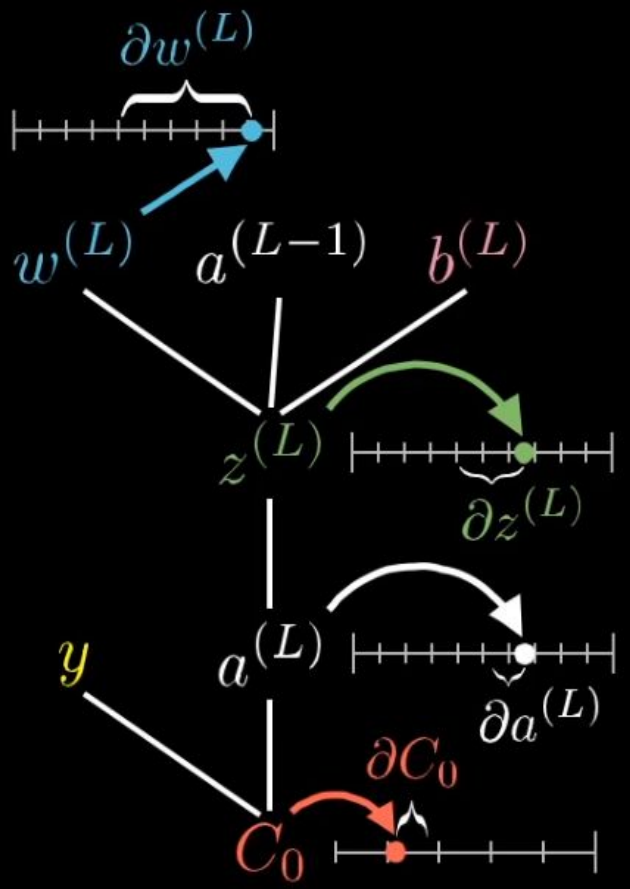
$$a^{(L)} = \sigma(z^{(L)})$$





$$\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial z^{(L)}}{\partial w^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial C0}{\partial a^{(L)}}$$

Chain rule

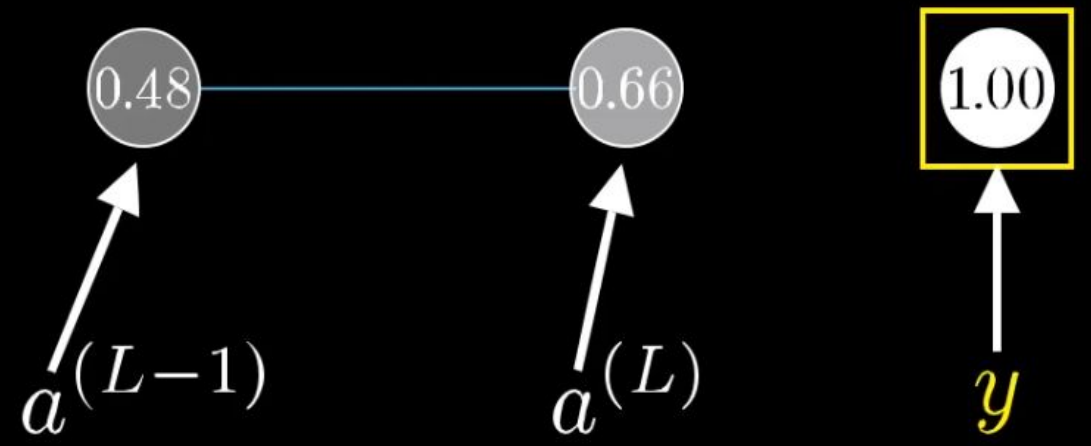


$$C_0(\dots) = (a^{(L)} - y)^2$$

$$z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$$

$$a^{(L)} = \sigma(z^{(L)})$$

Desired output



$$\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial z^{(L)}}{\partial w^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial C0}{\partial a^{(L)}}$$

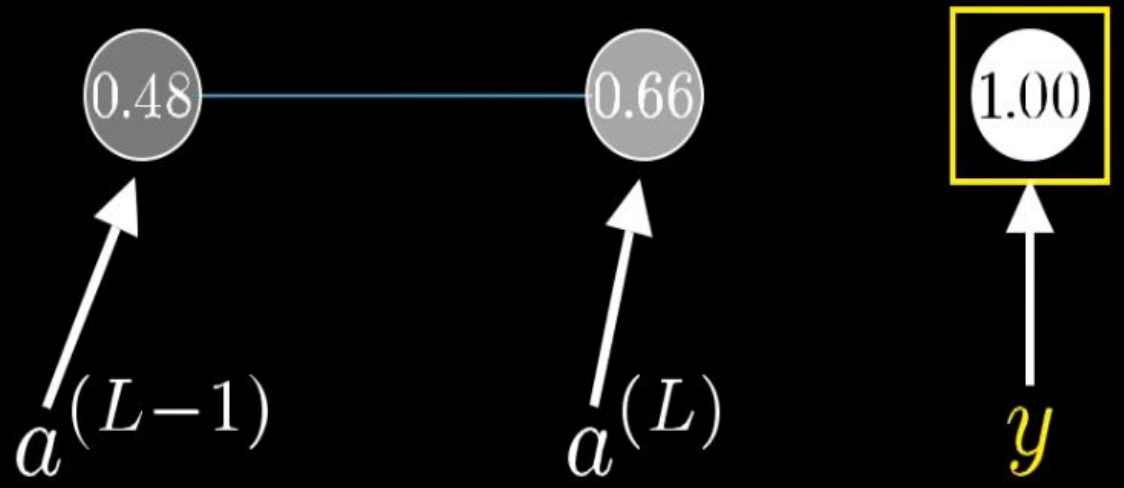
$$C_0 = (a^{(L)} - y)^2$$

$$z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$$

$$\frac{\partial C0}{\partial a^{(L)}} = 2(a^{(L)} - y)$$

$$a^{(L)} = \sigma(z^{(L)})$$

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'(z^{(L)})$$



$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = a^{(L-1)}$$

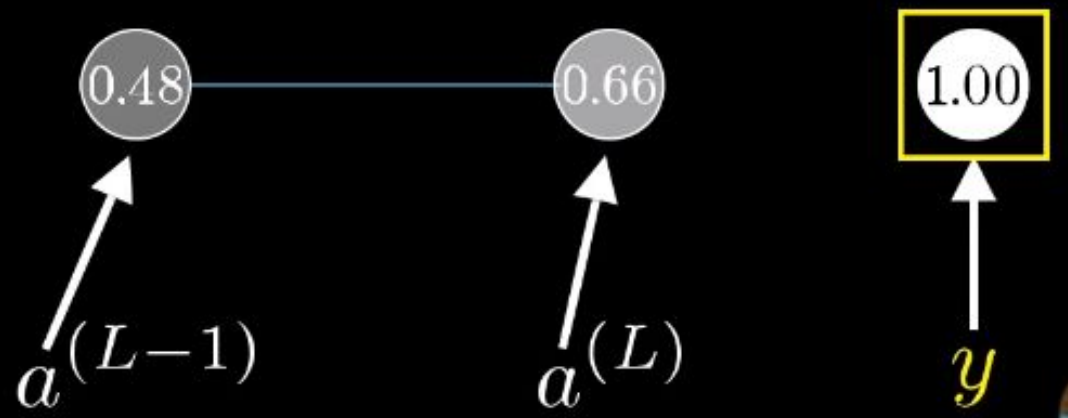
$$\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial z^{(L)}}{\partial w^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial C0}{\partial a^{(L)}} = a^{(L-1)} \sigma'(z^{(L)}) 2(a^{(L)} - y)$$

$$\nabla C = \begin{bmatrix} \frac{\partial C}{\partial w^{(1)}} \\ \frac{\partial C}{\partial b^{(1)}} \\ \vdots \\ \frac{\partial C}{\partial w^{(L)}} \\ \frac{\partial C}{\partial b^{(L)}} \end{bmatrix}$$

$$C_0 = (a^{(L)} - y)^2$$

$$z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$$

$$a^{(L)} = \sigma(z^{(L)})$$



Chybové funkce

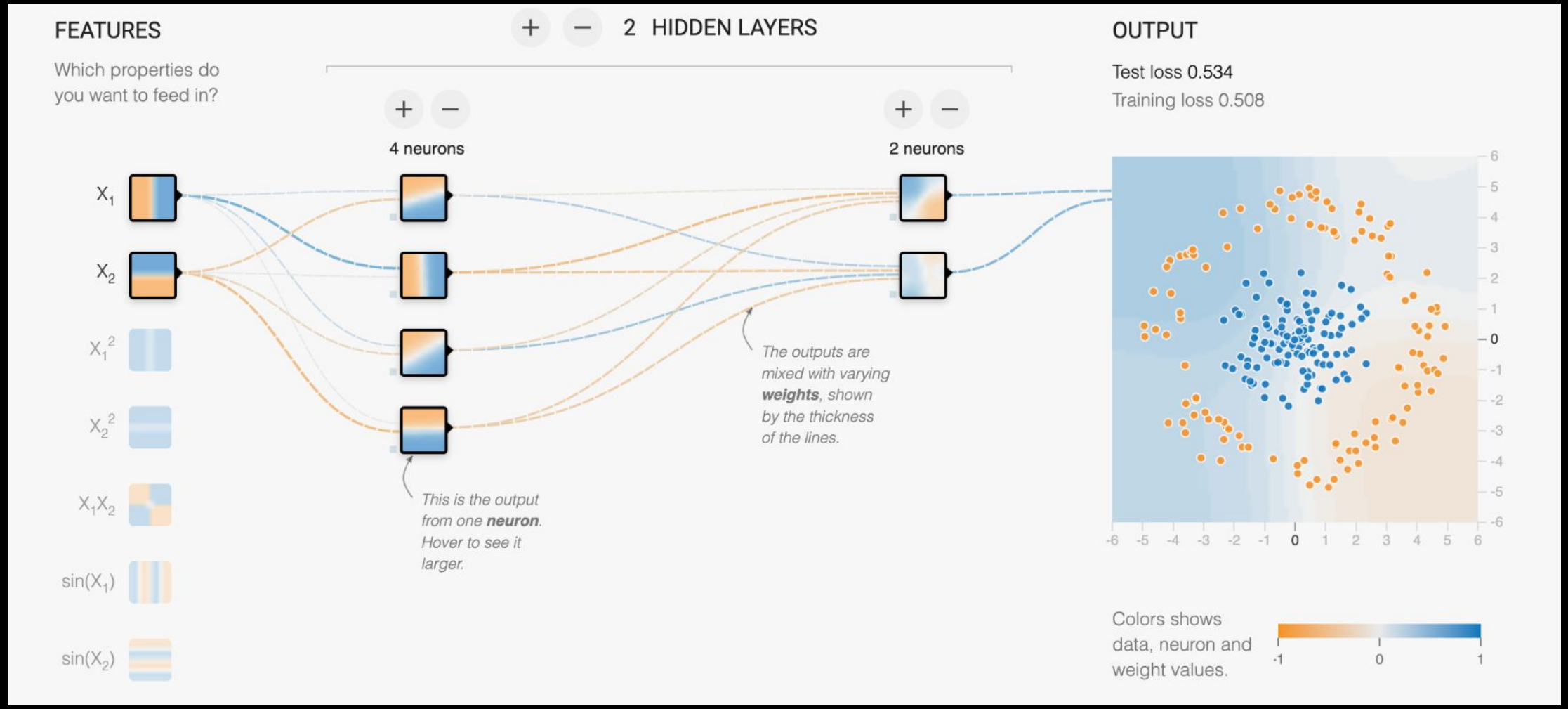
- Čtvercová chyba (MSE - Mean Squared Error)

- Křížová entropie (CE - Cross-Entropy)

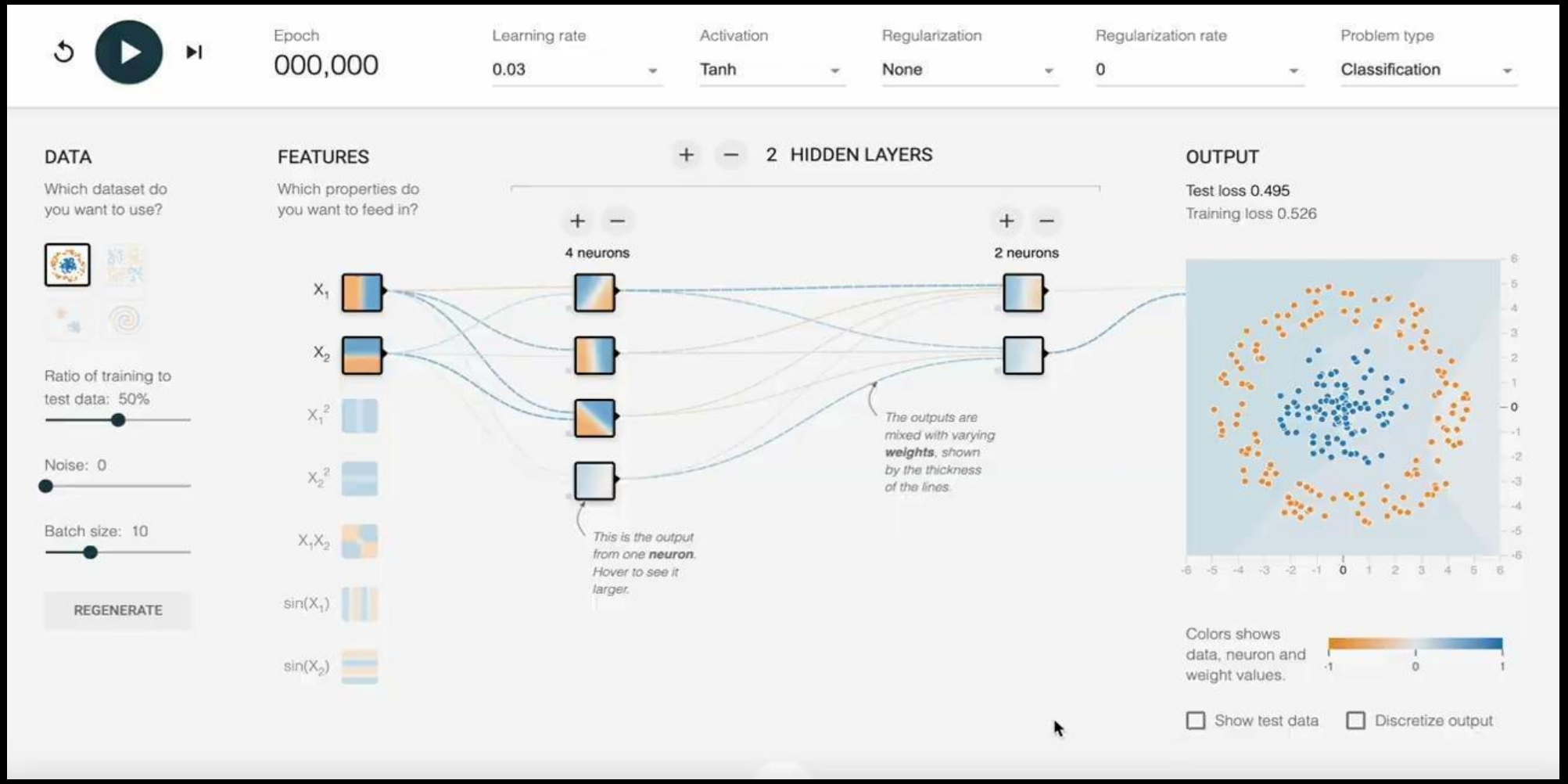
$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$L = -\frac{1}{m}\sum_{i=1}^{m} y_i \cdot \log(\hat{y}_i)$$

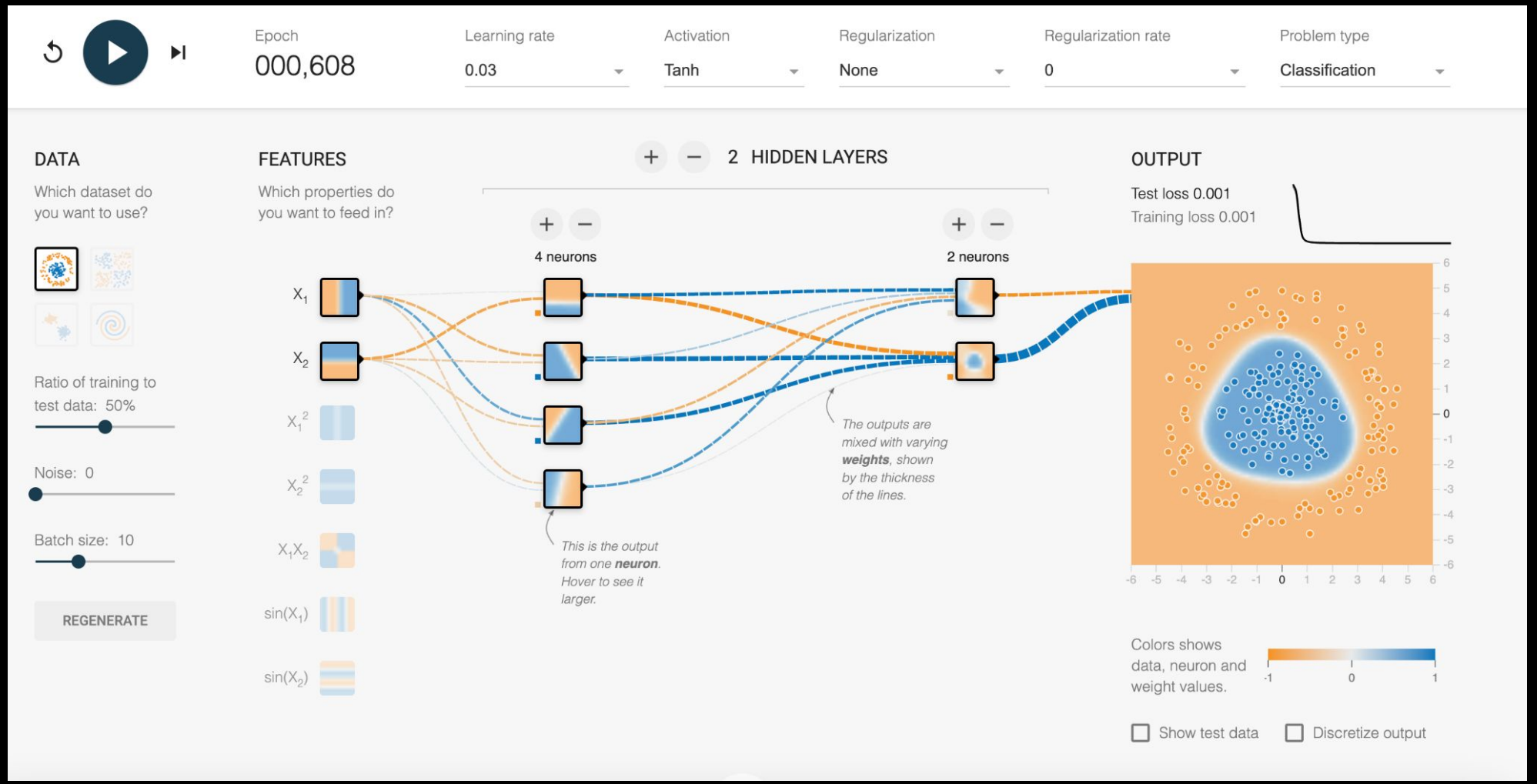
Jak se neuronová síť učí?



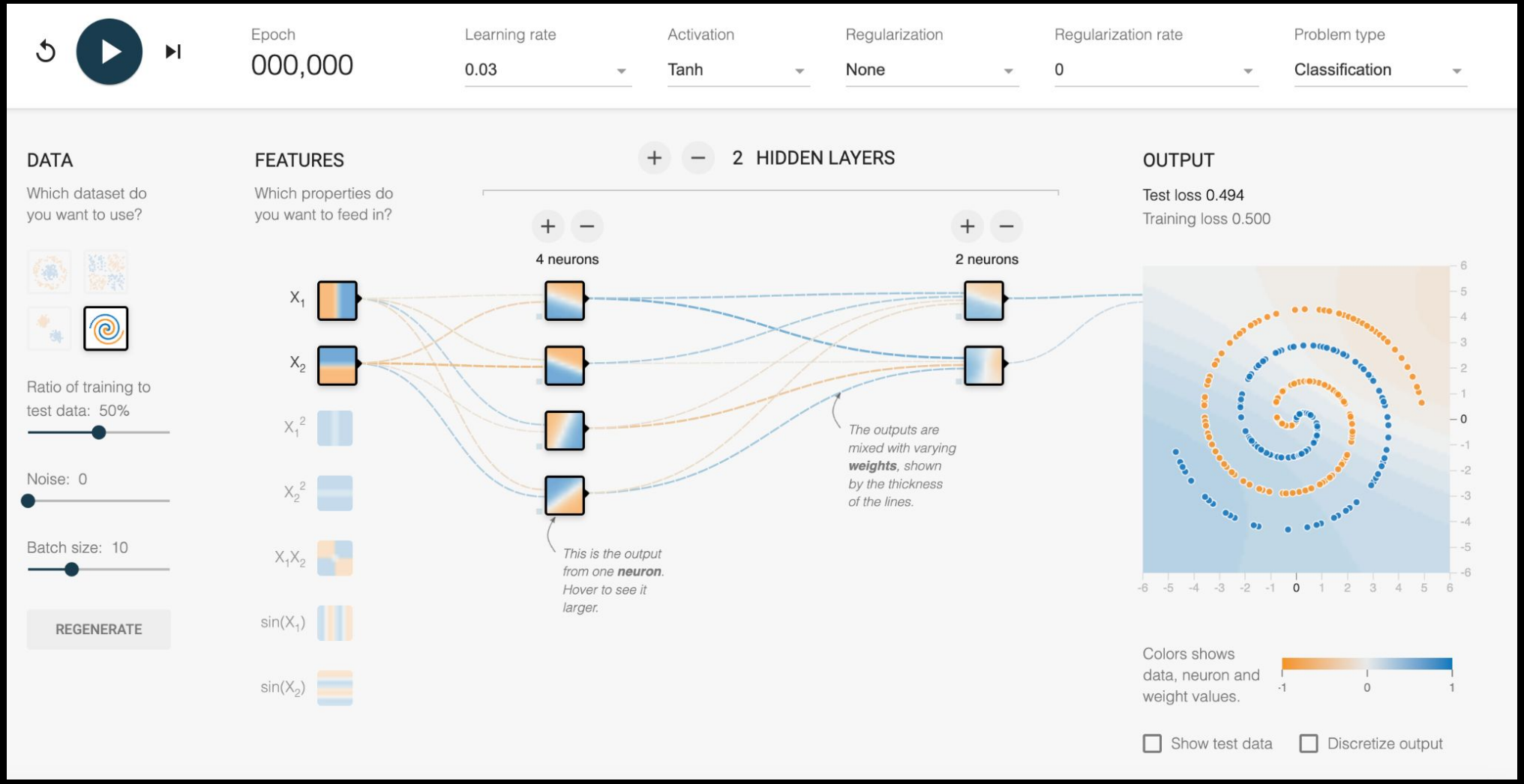
Jak se neuronová síť učí?



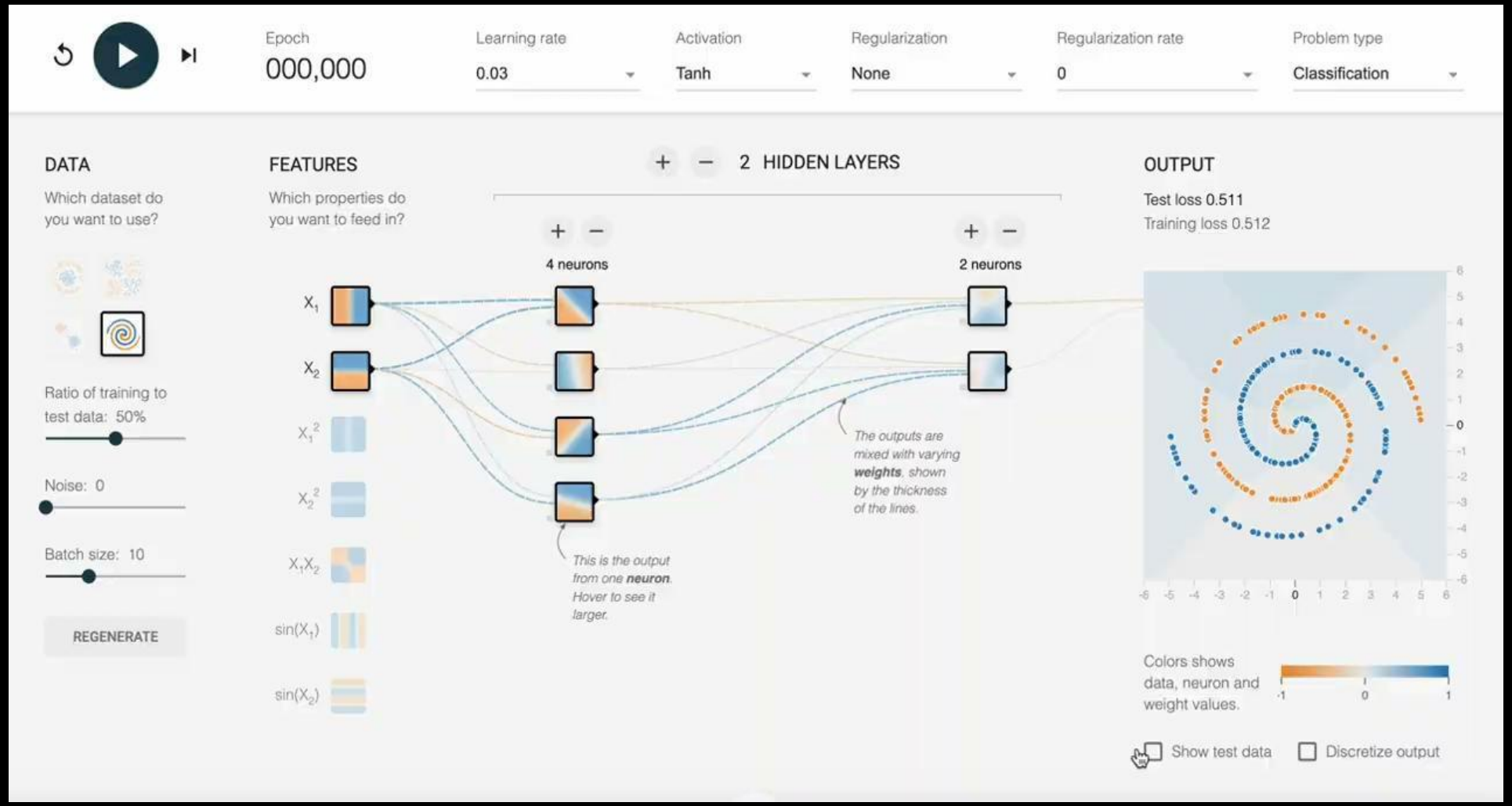
Jak se neuronová síť učí?



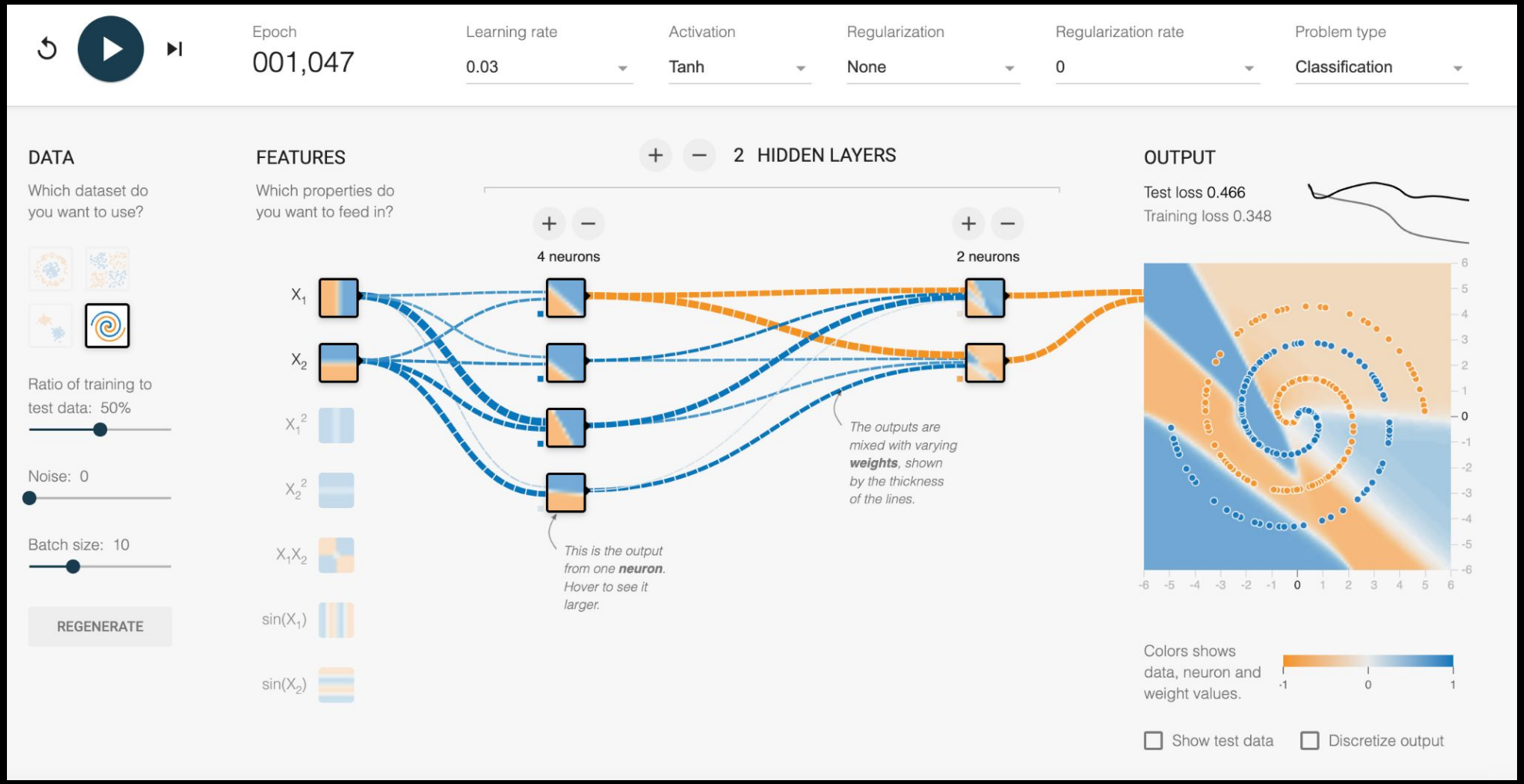
Jak se neuronová síť učí?



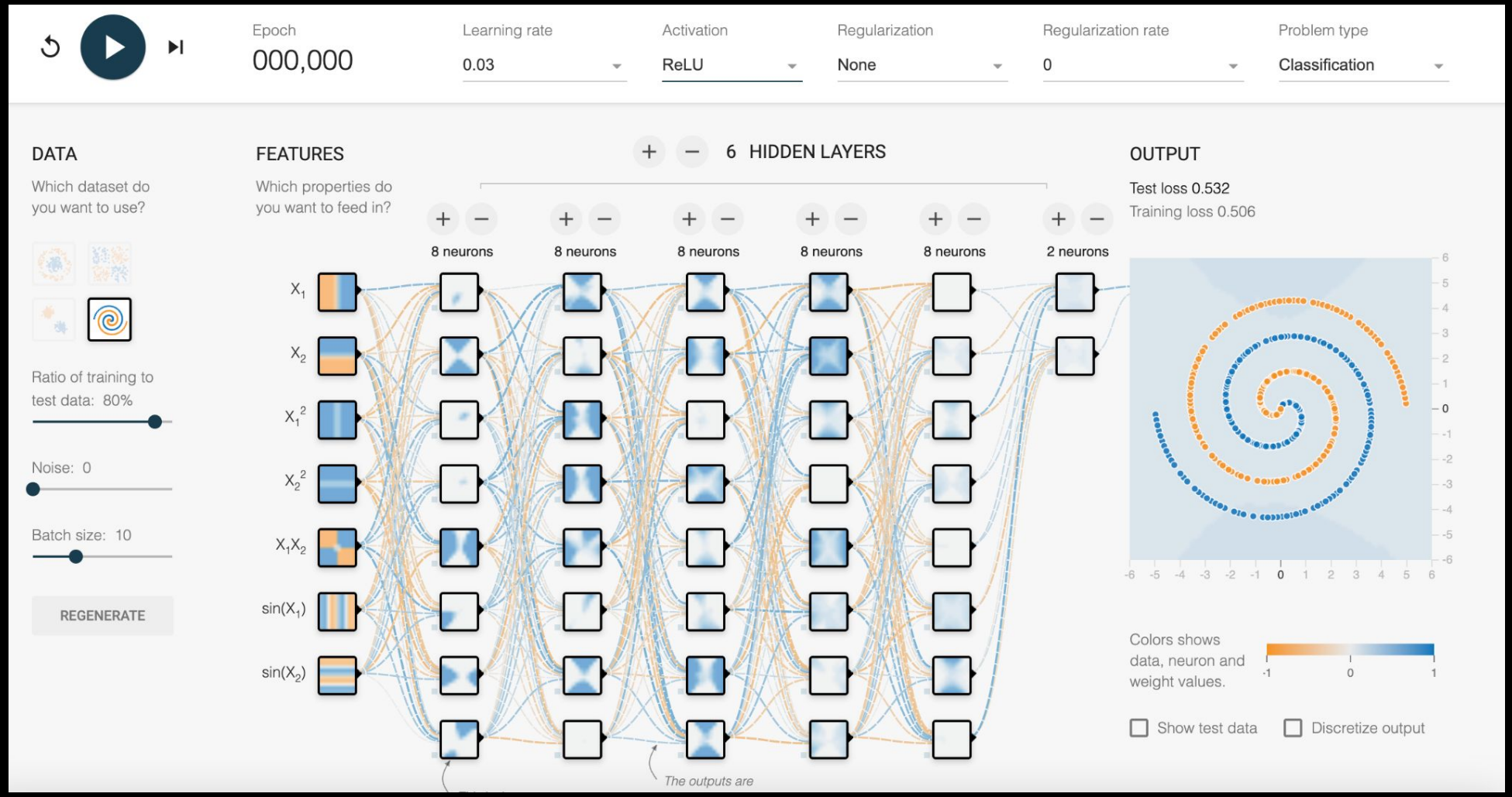
Jak se neuronová síť učí?



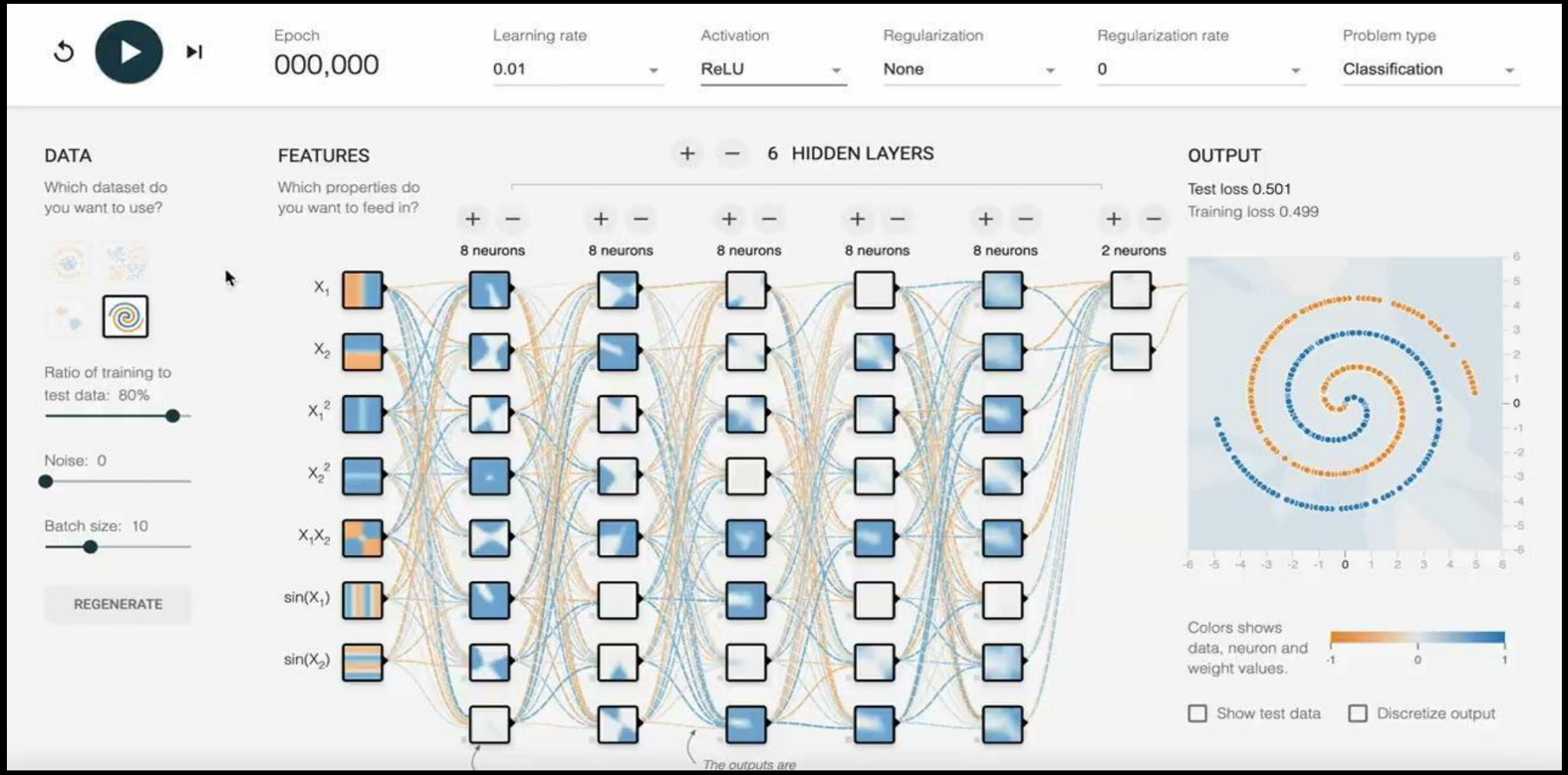
Jak se neuronová síť učí?



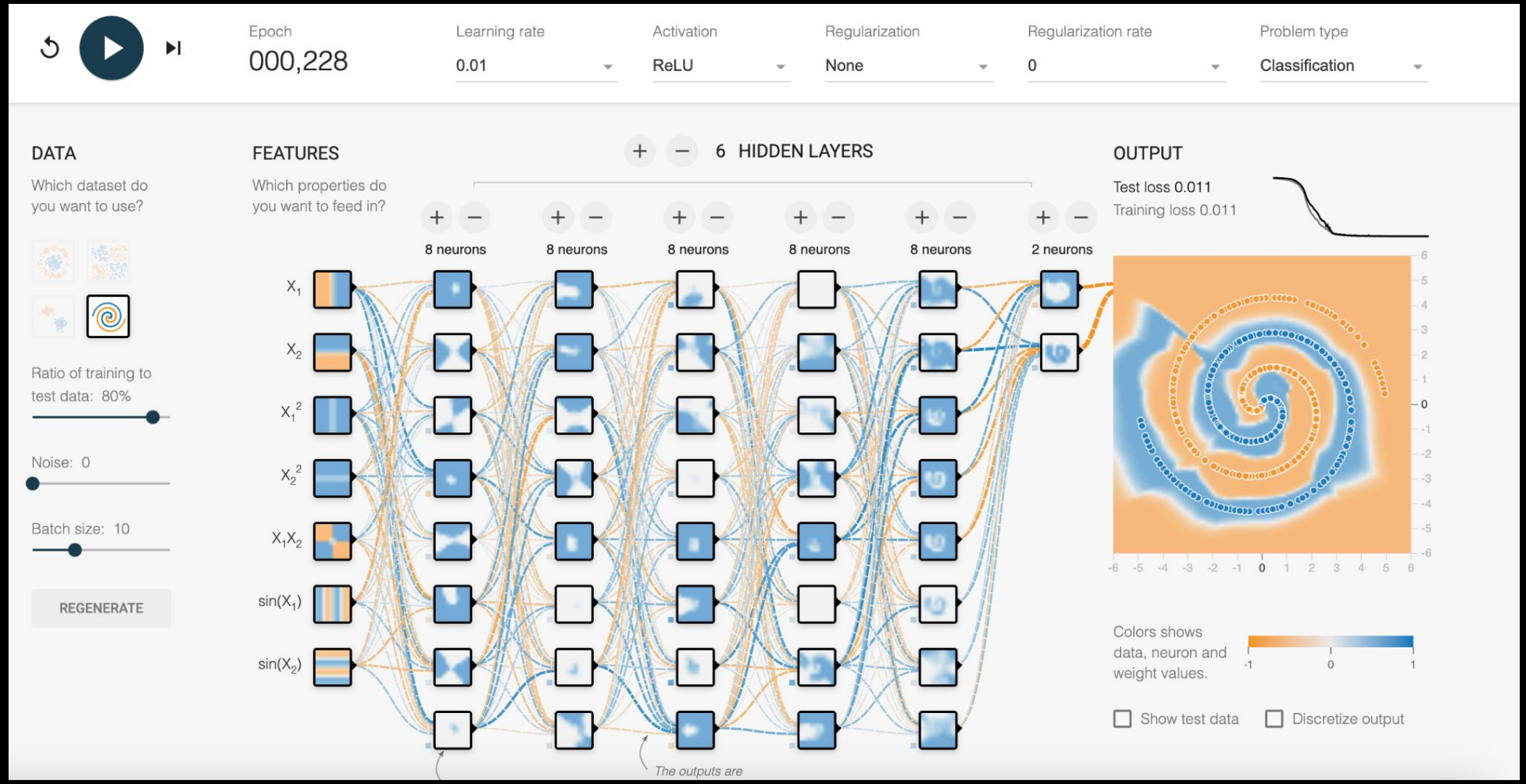
Jak se neuronová síť učí?



Jak se neuronová síť učí?



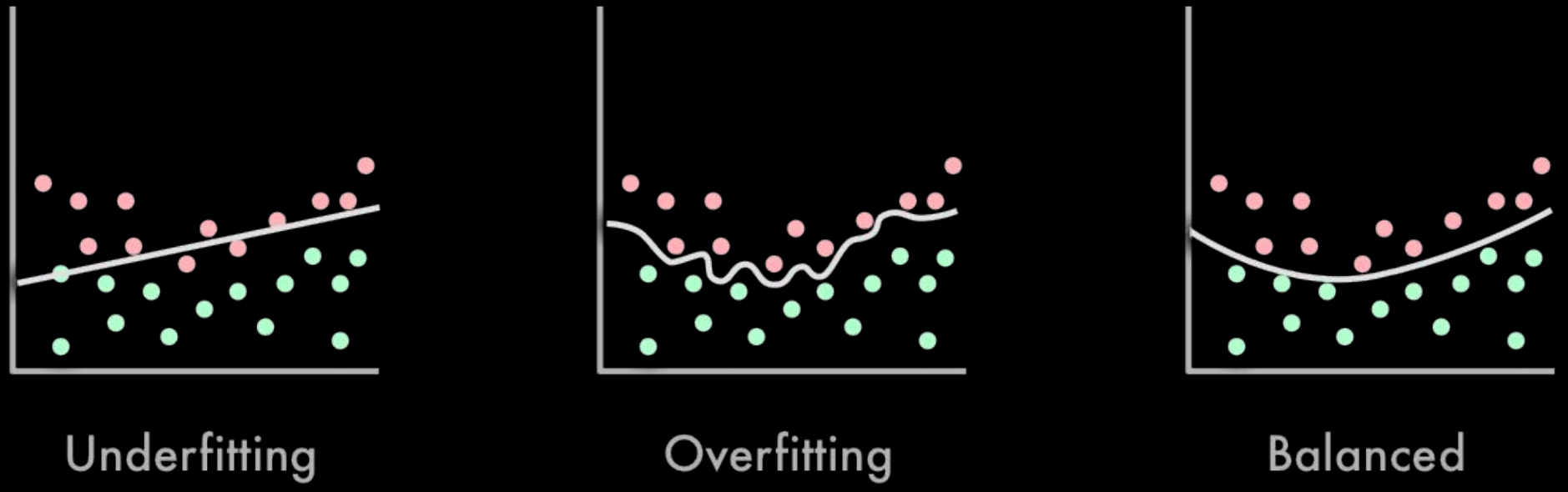
Jak se neuronová síť učí?



Problémy neuronových sítích

- Málo dat
- Špatně označená data
- Přeučení neuronové sítě (Overfitting)
- Nenaučení neuronové sítě (Underfitting)
- Dlouhé učení

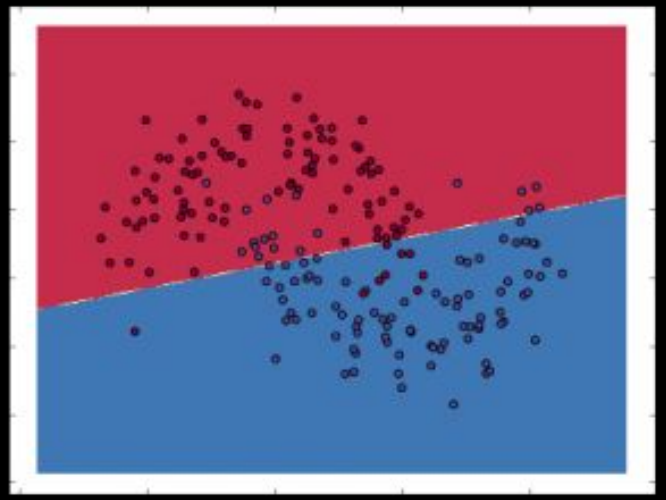
Overfitting, Underfitting

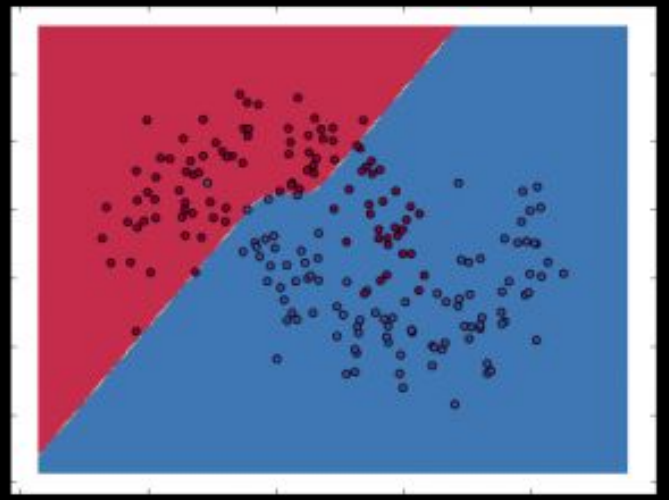


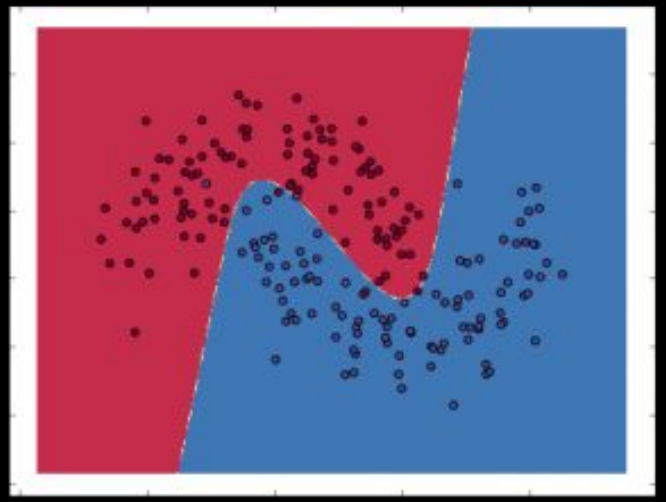
Regularizace Neuronové sítě

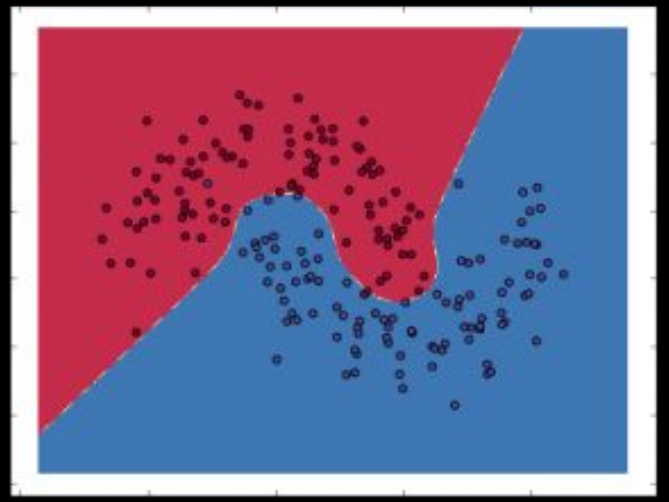
- Počet skrytých vrstev a neuronů
- Rozšiřování a normalizace dat
- L2 regularizace
- Míra učení (Learning rate)
- Včasné zastavení (Early stopping)
- Dropout regularizace

Počet skrytých vrstev a neuronů





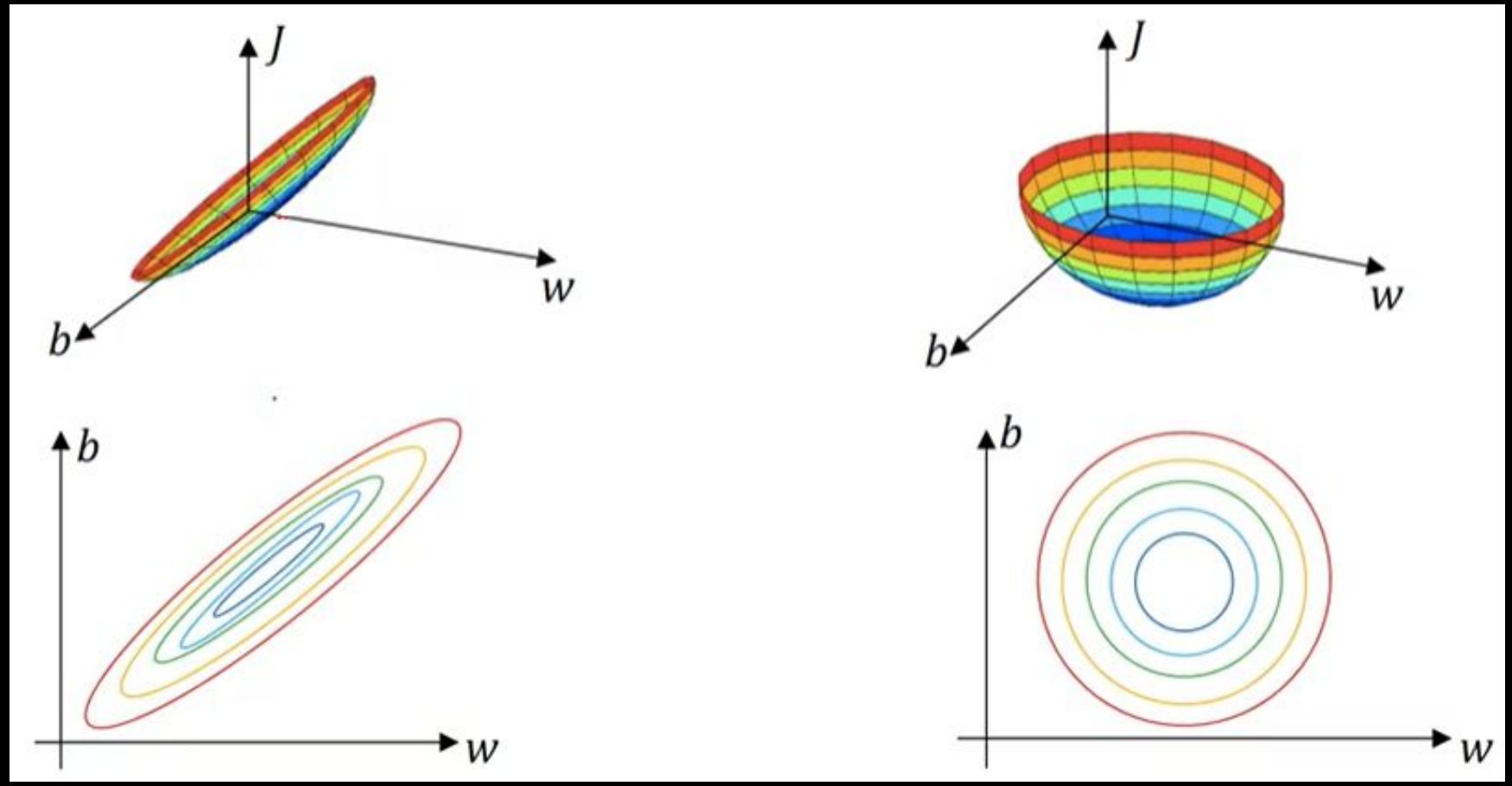




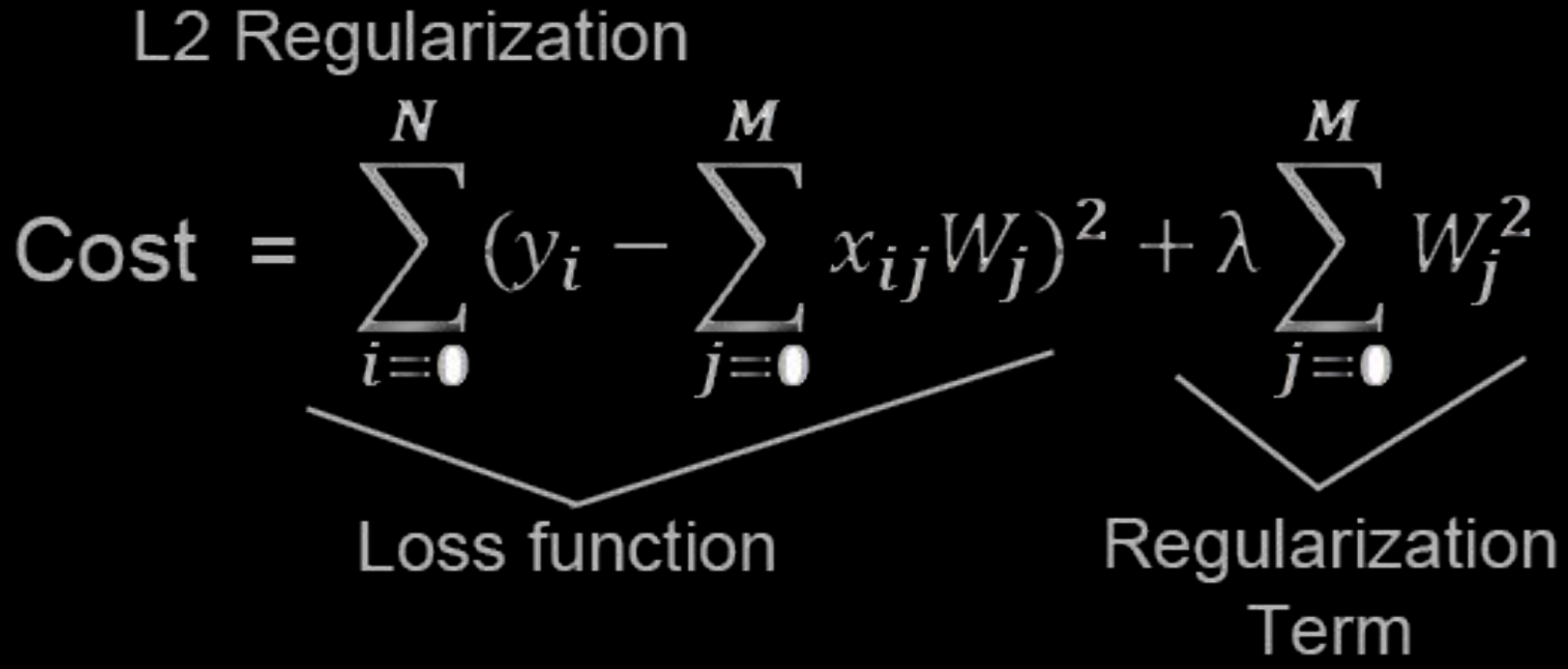
Rozšiřování dat

- Otáčení
- Ořezávání
- Distorze

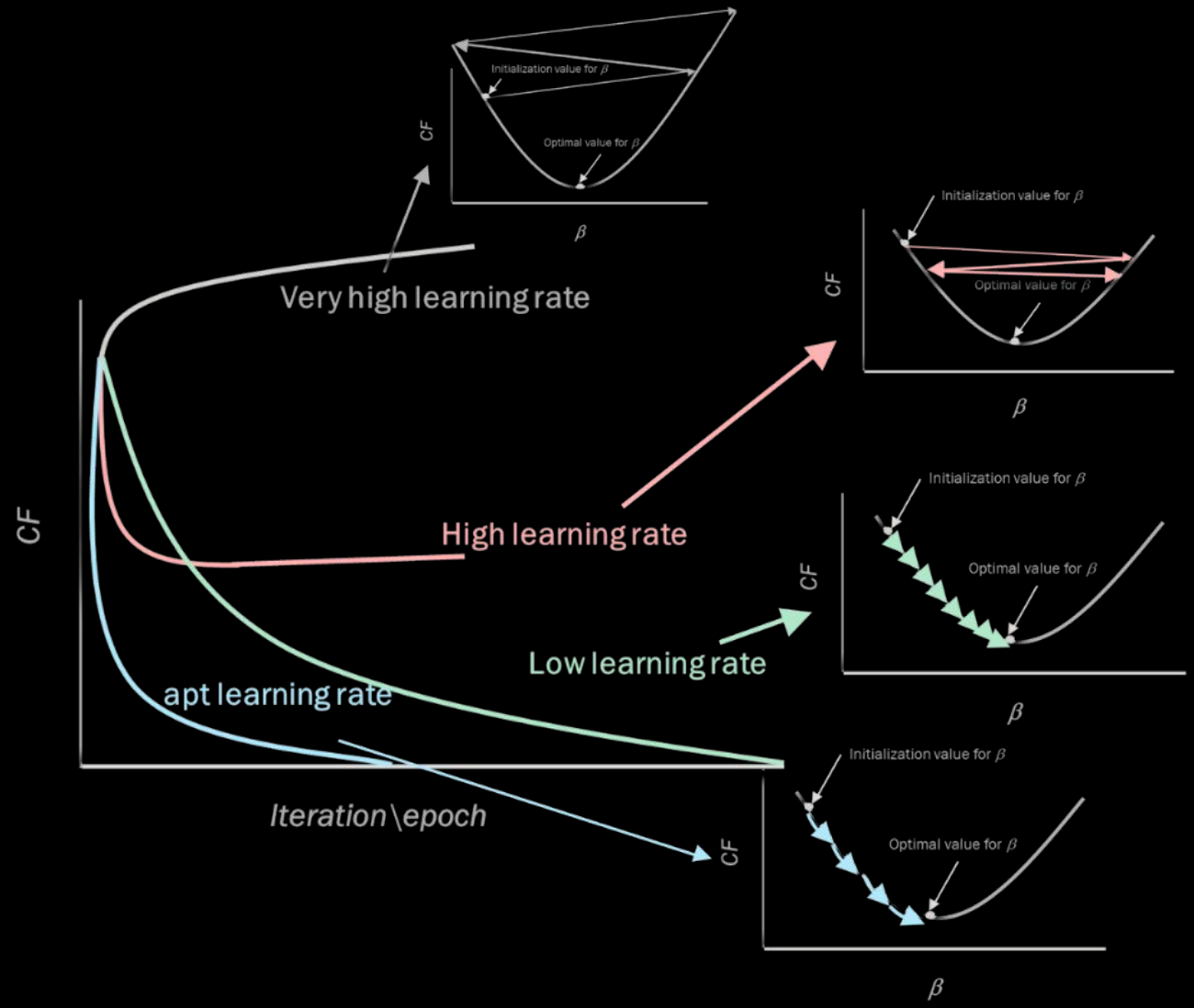
Normalizace dat



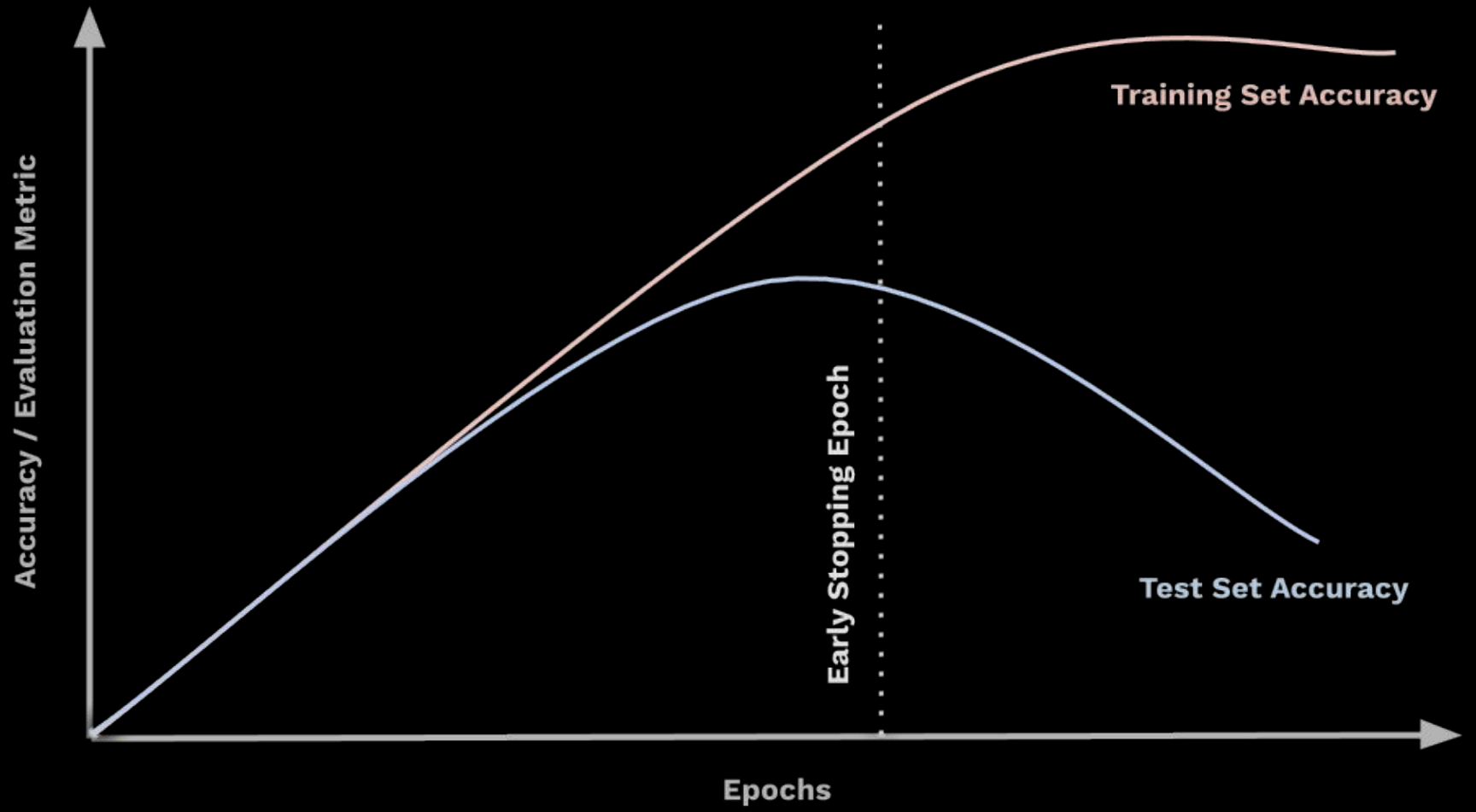
L2 regularizace



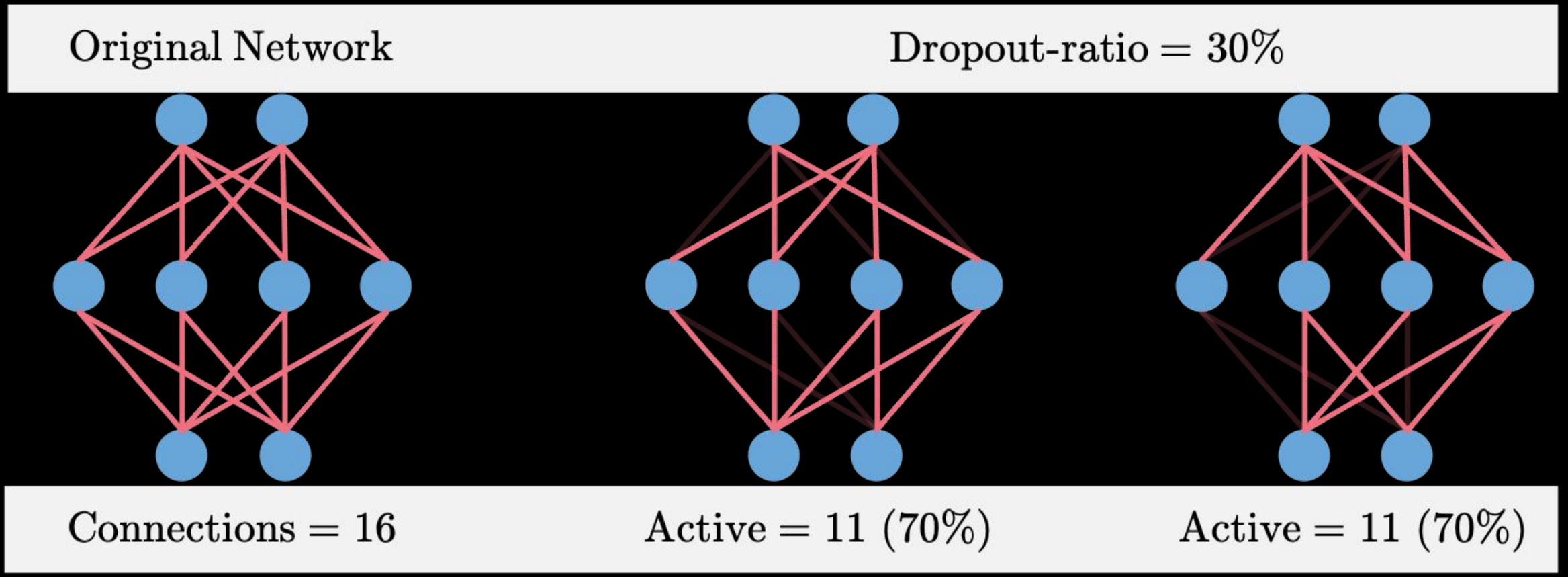
Míra učení (Learning rate)



Včasné zastavení (Early stopping)

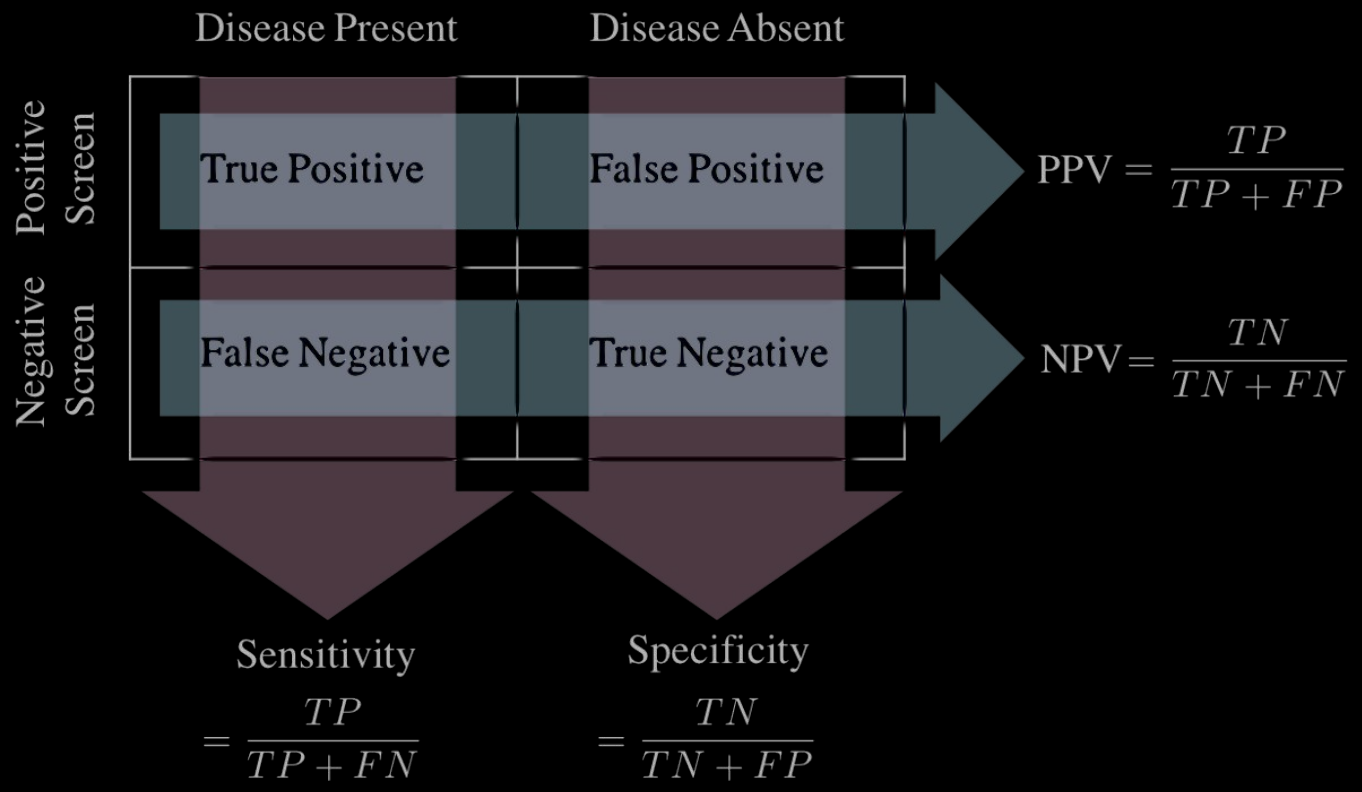


Dropout regularizace



Testování neuronové sítě

- Senzitivita (Recall)
- Specificita
- F1 scóre
  - Precision = PPV



$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$