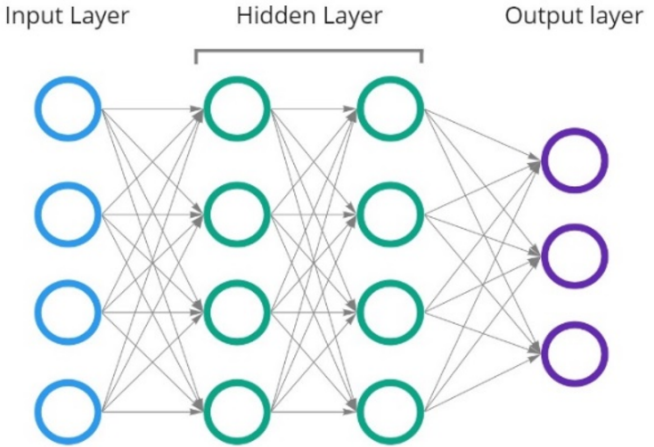
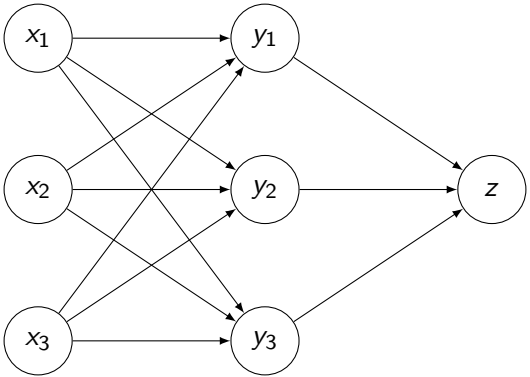
Typowa reprezentacja sieci neuronowej



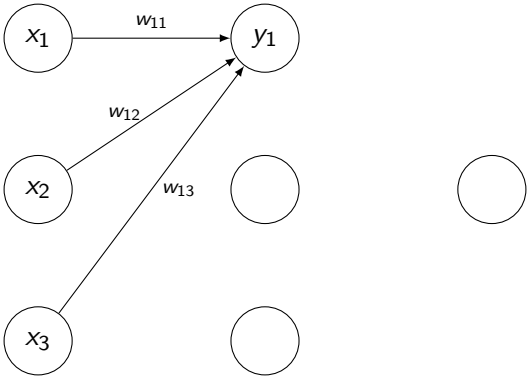
https://www.researchgate.net/figure/Neural-Network-Representation˙fig2˙370269724

Typowa reprezentacja sieci neuronowej



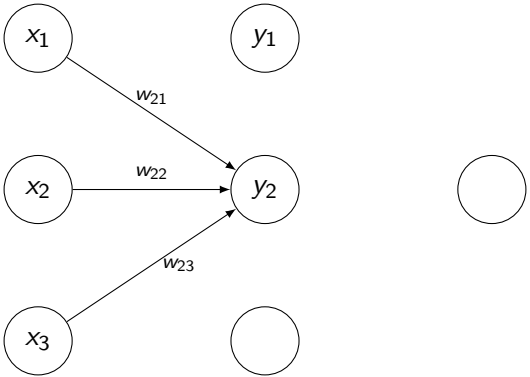
Niech $\sigma\colon \mathbb{R} \to \mathbb{R}$ – funkcja aktywacji (np. sigmoid, ReLU, tgh).

Typowa reprezentacja sieci neuronowej



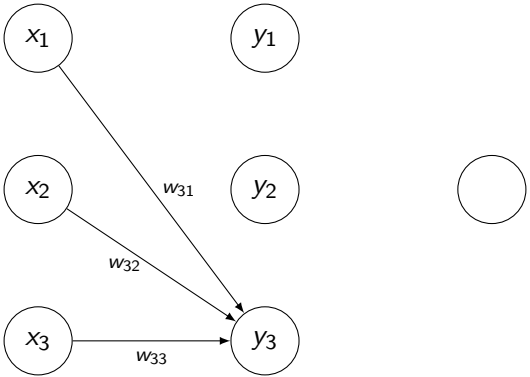
$$y_1 := \sigma(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1).$$

Typowa reprezentacja sieci neuronowej



$$y_2 := \sigma(w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2).$$

Typowa reprezentacja sieci neuronowej



$$y_3 := \sigma(w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + b_3).$$

Razem:

$$
\begin{aligned}
y_1 &= \sigma(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1),\\
y_2 &= \sigma(w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2),\\
y_3 &= \sigma(w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + b_3).
\end{aligned}
$$

Oznaczenie:

$$
\begin{aligned}
y'_1 &= w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1,\\
y'_2 &= w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2,\\
y'_3 &= w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + b_3.
\end{aligned}
$$

Wtedy:

$$
\begin{aligned}
y_1 &= \sigma(y'_1),\\
y_2 &= \sigma(y'_2),\\
y_3 &= \sigma(y'_3).
\end{aligned}
$$

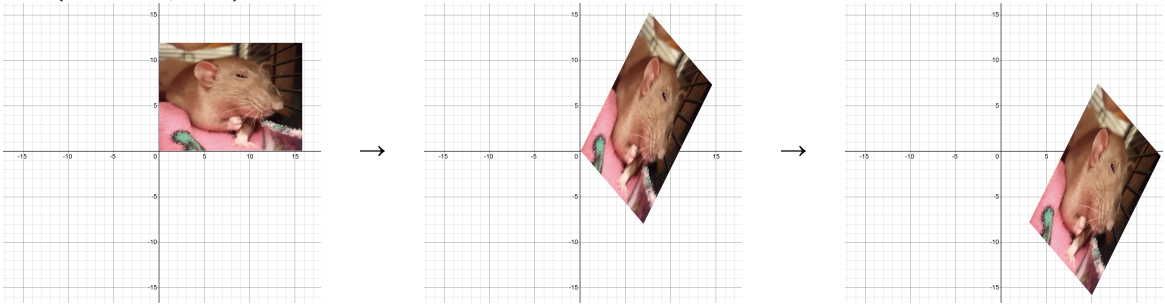
Macierzowo:

$$\begin{pmatrix} y_1' \\ y_2' \\ y_3' \end{pmatrix} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}.$$

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} \sigma(y_1') \\ \sigma(y_2') \\ \sigma(y_3') \end{pmatrix} \qquad \left[= \sigma \left(\begin{pmatrix} y_1' \\ y_2' \\ y_3' \end{pmatrix} \right) \right].$$

Trochę algebry liniowej

Przekształcenie afiniczne – złożenie przekształcenia liniowego z translacją (przesunięciem).



$$F(X) = AX + B.$$

$$X \in \mathbb{R}^n, A \in M_{m\times n}(\mathbb{R}), B \in \mathbb{R}^m.$$

Trochę algebry liniowej

Sieć neuronowa F (właściwie: wielowarstwowy perceptron) to zatem złożenie przekształceń afinicznych „przeplatanych" z funkcjami aktywacji, to znaczy:

$$F = \sigma_n \circ F_n \circ \ldots \circ \sigma_1 \circ F_1 \circ \sigma_0 \circ F_0,$$

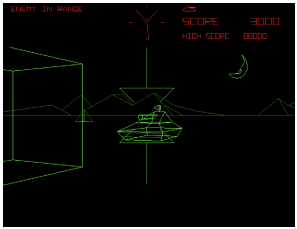
gdzie $\sigma_i \colon \mathbb{R}^* \to \mathbb{R}^*$ to zwektoryzowane funkcje aktywacji, a F_i to kompatybilne przekształcenia afiniczne.

Dla konkretnego wejścia (wektora X), obliczenia związane z wyznaczeniem wyjścia ($= F(X)$) to głównie „czysta" algebra liniowa. Te obliczenia łatwo się paralelizują.

Dygresja: grafika 3D

Grafika 3D: w znacznej części algebra liniowa.

- Elementy sceny 3D to wielościany.
- Główny problem: narysować dużo wielościanów uwzględniając perspektywę i ruch kamery.
- Typowe rozwiązanie: kamera stoi w miejscu, przekształcamy resztę świata (obrót świata wokół kamery zamiast obrotu kamery w świecie).
- Te transformacje (wielościanów) to przekształcenia afiniczne.



1980: mało ścian

2015: więcej ścian

Dygresja: grafika 3D

Pierwotnie obliczenia związane z grafiką 3D były wykonywane przez CPU.

Lata '90: upowszechnienie GPU – wyspecjalizowanych układów do (głównie) obliczeń macierzowych.

- CPU: mało rdzeni (∼kilkanaście) ogólnego zastosowania.
- GPU: dużo rdzeni (tysiące); rdzenie są wyspecjalizowane, indywidualnie mało wydajne, wykorzystujące wysoką paralelizację obliczeń macierzowych.
- Surowa moc obliczeniowa GPU i CPU (z podobnej „półki"): ∼10:1 na korzyść GPU.

Obliczenia w grafice 3D to „te same" obliczenia, które pojawiają się w sieciach neuronowych – stąd szerokie zastosowanie GPU w uczeniu maszynowym.

Druga połowa lat '10: TPU (*tensor processing unit*).