Sztuczne sieci neuronowe

Marek Grochowski

Wydział Fizyki, Astronomii i Informatyki Stosowanej UMK

http://www.fizyka.umk.pl/~grochu/nn
grochu@is.umk.pl

3 marca 2026

Plan

1. Perceptrony, sieci wielowarstwowe jednokierunkowe (MLP)
2. Metody uczenia i algorytm wstecznej propagacji błędu
3. Radialne funkcje bazowe (RBF) i metody aproksymacji
4. Samoorganizacja, sieci SOM, uczenie konkurencyjne
5. Sieci dynamiczne: model Hopfielda, maszyny Boltzmana
6. Głębokie sieci neuronowe (DNN)
7. Sieci splotowe (CNN)
8. Sieci rekurencyjne (RNN) i uczenie sekwencji
9. Autoenkodery, wykrywanie cech, DBN
10. Modele generatywne GAN, VAE
11. Transformery, mechanizm uwagi, LLM, ViT

Literatura I

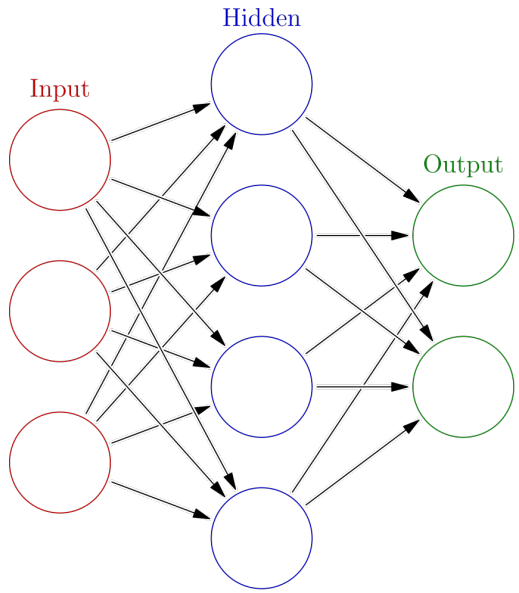
- S. Osowski, *Sieci neuronowe w ujęciu algorytmicznym*, Wydawnictwo Naukowo-Techniczne, Warszawa 1996.
- R. Tadeusiewicz, *Sieci neuronowe*, Akademicka Oficyna Wydawnicza RM, Warszawa 1993.
- R. Tadeusiewicz, T. Gąciarz, B. Borowik, B. Lepe, *Odkrywanie właściwości sieci neuronowych przy użyciu programów w języku C#*, Polska Akademia Umiejętności, 2008.
- J. Zurada, M. Barski, W. Jędruch, *Sztuczne sieci neuronowe*, Wydawnictwo Naukowe PWN, 1996.

Literatura II

- Ian Goodfellow, Yoshua Bengio i Aaron Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`
- Michael Nielsen, *Neural Networks and Deep Learning*, `http://neuralnetworksanddeeplearning.com/`
- Julien Vitay, *Neurocomputing*, `https://julien-vitay.net/course-neurocomputing/`
- Denny Britz, *Deep Learning Glossary*, `http://www.wildml.com/deep-learning-glossary/`

Sztuczne Sieci Neuronowe

Systemy obliczeniowe inspirowane biologicznymi sieciami neuronowymi, składające się ze zbioru prostych jednostek przetwarzających sygnał (neuronów), komunikujących się między sobą za pomocą połączeń (wag synaptycznych).



- model matematyczny realizujący odwzorowanie $f(\mathbf{x}; \mathbf{W}) = \mathbf{y}$
- systemy uczące się, adaptowanie wag $\mathbf{W}$, Uczenie Maszynowe
- systemy inteligentne: Inteligencja Obliczeniowa, Sztuczna Inteligencja

Inteligencja obliczeniowa

Computational Intelligence (CI) zajmuje się rozwiązywaniem problemów, które nie są efektywnie algorytmizowalne.
Cechą wielu systemów CI jest rozwiązywanie zadań na podstawie przykładów — uczenie się z empirycznych danych zamiast programowania reguł.
Problemy niealgorytmizowalne:

- zagadnienia o złożoności wykładniczej lub wyższej (np. problem komiwojażera)
- procesy z niejasnymi regułami lub silnym komponentem stochastycznym
- zjawiska trudne do opisania przez proste reguły
- środowiska zmienne, wymagające adaptacji algorytmu

Problemy efektywnie nieal gorytmizowalne

Problemy NP-trudne - złożoność obliczeniowa rośnie szybciej niż jakikolwiek wielomian liczby elementów

Przykład: problem komiwojażera

Rysunek 1.1. Najkrótsza trasa premiera przebiegająca przez wszystkie miasta wojewódzkie w podziale administracyjnym z 1975 roku (A. Adrabiński)

Tabela 1.1. Czasy znalezienia przez komputer, wykonujący 100 miliardów operacji na sekundę, najkrótszej trasy podróży premiera (zob. ćwiczenie 1.3)

Liczba województw	Czas znajdowania najkrótszej trasy
17	3.5 minuty
25	$2 \cdot 10^5$ lat
49	$4 \cdot 10^{42}$ lat

Dla 100 miejsc mamy 100! (liczba 161-cyfrowa)

Rysunek: M. Sysło, „Algorytmy”

Problemy niealgorytmizowalne - przykłady

- rozumienie sensu zdań,
- działania twórcze, decyzje intuicyjne,
- rozpoznawanie twarzy i obrazów,
- rozpoznawanie pisma ręcznego,
- rozpoznawanie mowy i sygnałów, percepcja,
- sterowanie robotem, nieliniowymi układami,
- diagnostyka medyczna, planowanie terapii.

CI i sztuczna inteligencja (AI)

Klasyczne rozumienie AI i CI:
CI - percepcja i sterowanie: zachowania sensomotoryczne – sieci neuronowe i uczenie maszynowe
AI - wyższe czynności poznawcze: logika, język, rozumowanie, planowanie

Good old-fashioned **Artificial Intelligence (GOFAI)** to część CI posługująca się symboliczną reprezentacją wiedzy, zajmuje się rozumowaniem, tworzeniem systemów ekspertowych.

CI i sztuczna inteligencja (AI)

Obecnie „sztuczna inteligencja” bywa używana do opisu szerokiego spektrum metod i algorytmów naśladujących inteligentne zachowania.

- **Strong AI, Artificial General Intelligence (AGI)** - agent świadomie działający i uczący sie działać w środowisku (podobnie do człowieka)
- **Weak AI** - systemy rozwiązujące (uczące sie rozwiązywać) konkretne zadania

Efekt AI (Larry Tesler)
„AI is whatever hasn’t been done yet”

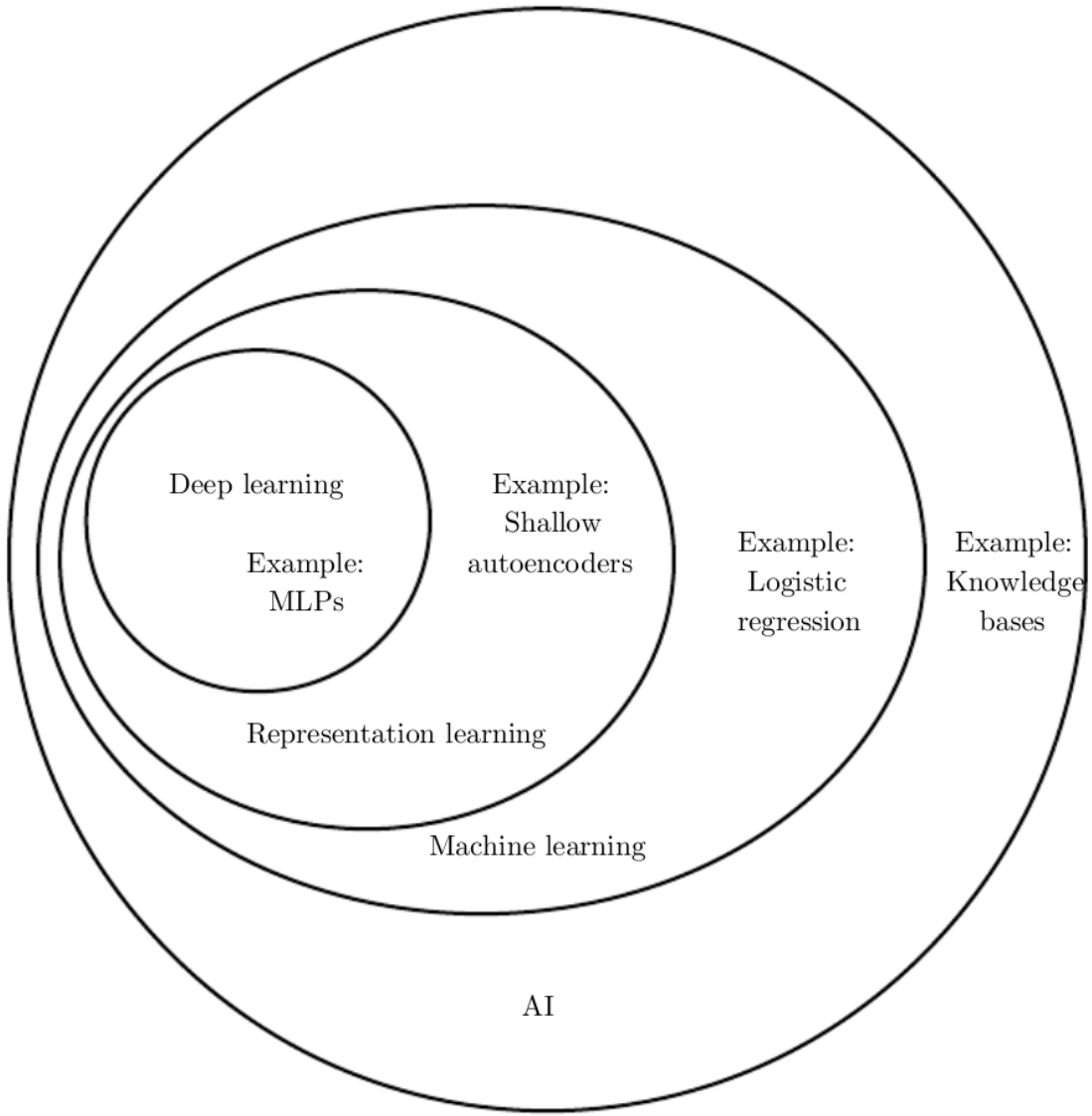
Definicja AI jest elastyczna - granica między tym, co uważamy za AI, a tym, co jest już standardową technologią, ciągle się przesuwa.

Uczenie maszynowe

- **Uczenie maszynowe** (Machine Learning, ML) - sytemy uczące się rozwiązywać problemy na podstawie przypadków uczących (danych)
- **Generalizacja** - dobrze wyuczony model działa poprawnie także dla przypadków spoza danych treningowych
- Gdy dane się zmienią - system można dostosować, trenując go na nowych danych

Rysunek: https://xkcd.com/

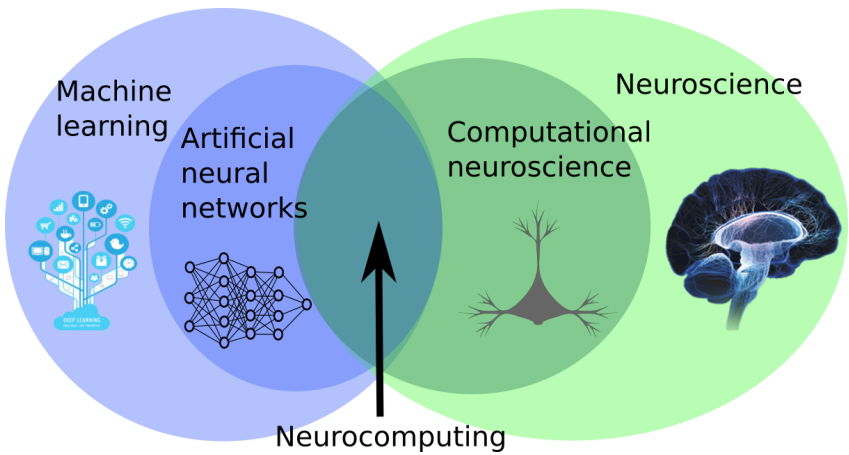
ANN i sztuczna inteligencja (AI)



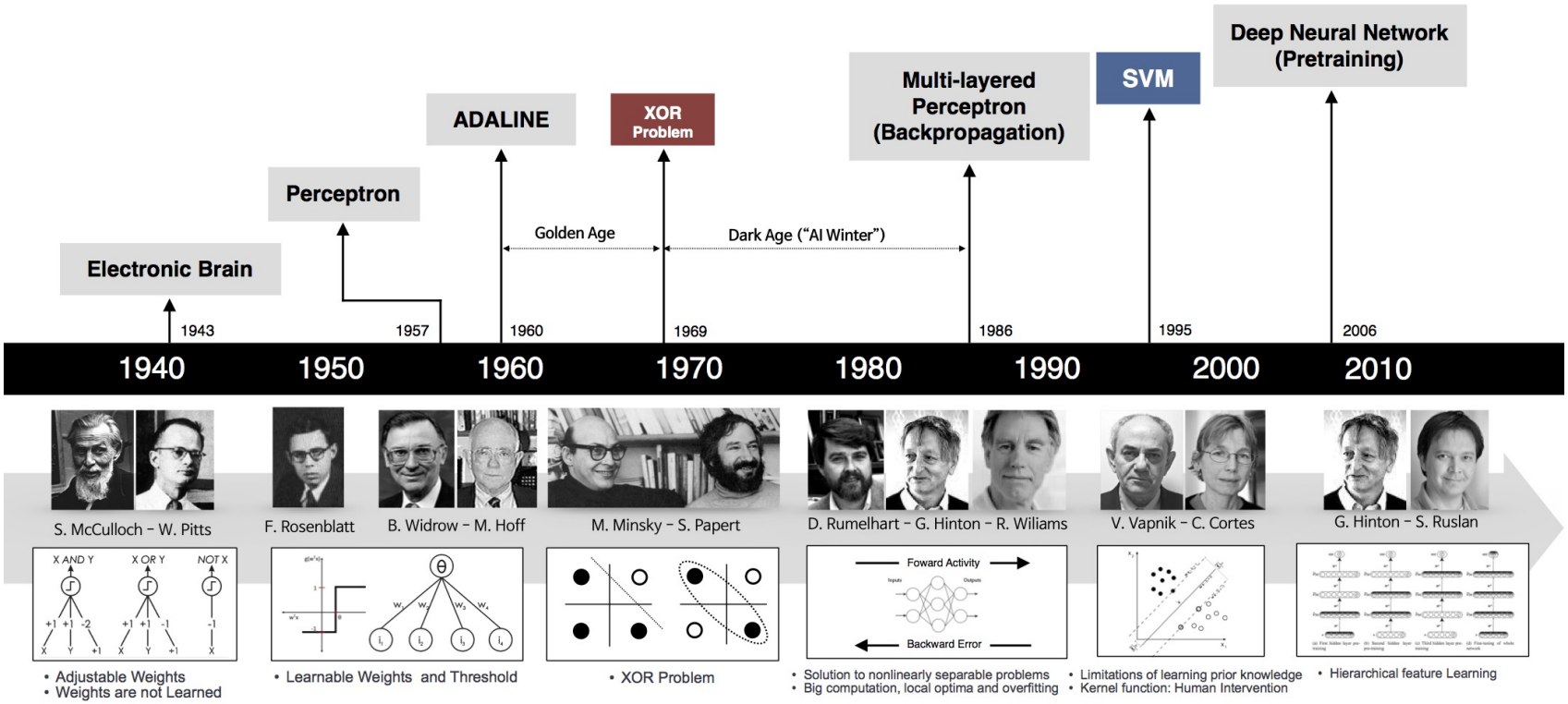
Goodfellow, 2016 [5]

Obliczenia neuronowe

- **modelowanie działania mózgu** - symulacje komputerowe, modele procesów kognitywnych
- przetwarzanie równoległe bardzo wydajne przy rozwiązywaniu niektórych problemów
- odporność na uszkodzenia, oddziaływania lokalne, uszkodzenie części sieci nie musi powodować drastycznych skutków
- rozwiązywanie praktycznych problemów metodami inspirowanymi biologicznie
  -> **Sztuczne Sieci Neuronowe (Artificial Neural Networks)**



Źródło: J. Vitay [7]



https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

Wczesny etap rozwoju (1943-1960)

1938 N. Rashevsky - neurodynamika, sieci neuronowe jako układ dynamiczny (pierwszy model sieci neuronowej)

1943 W. McCulloch, W. Pitts - sieci neuronowe jako układy logiczne

1949 D. Hebb - uczenie się na poziomie synaptycznym (pierwsza reguła uczenia)

1952 A. Hodgkin, A. Huxley - szczegółowy matematyczny model neuronu kałamarnicy olbrzymiej (nagroda Nobla 1963)

1958 F. Rosenblatt - Perceptron, sieć jako funkcja matematyczna
J. von Neuman, *The Computer and the Brain*

Pierwsze algorytmy uczące się (1960-1986)

1960 B. Widrow, M. Hoff - Adaline

1967 K. Steinbuch, E. Schmitt - macierze uczące się

1969 M. Minsky, S. Papert - książka *Perceptrony* i ograniczenia sieci liniowych

1973 Chr. von der Malsburg - samoorganizacja w układzie nerwowym

1982 J. Hopfield - model Hopfielda
T. Kohonen - mapy samoorganizujące się

1985 D. Ackley, G. Hinton, T. Sejnowski - maszyny Boltzmanna

1986 D. Rumelhart, G. Hinton, R. Williams - wsteczna propagacja błędów

1990 T. Poggio, F. Girosi - sieci RBF, teoria regularyzacji

1995 V. Vapnik - SVM i „koniec ery Data Mining"

Początki głębokiego uczenia (1980-2010)

1983 K. Fukushima, S. Miyake, T. Ito - Neokognitron, głęboka sieć neuronowa, inspiracja późniejszych sieci splotowych

1989 Yann LeCun - pierwsze praktyczne sieci splotowe (LeNet)

1997 S. Hochreiter, J. Schmidhuber - sieć rekurencyjna LSTM

1998 Y. LeCun, LeNet5, praktyczne zastosowanie CNN w analizie obrazów

2005 GPU do trenowania głęokich sieci, przyspieszenie rozwoju

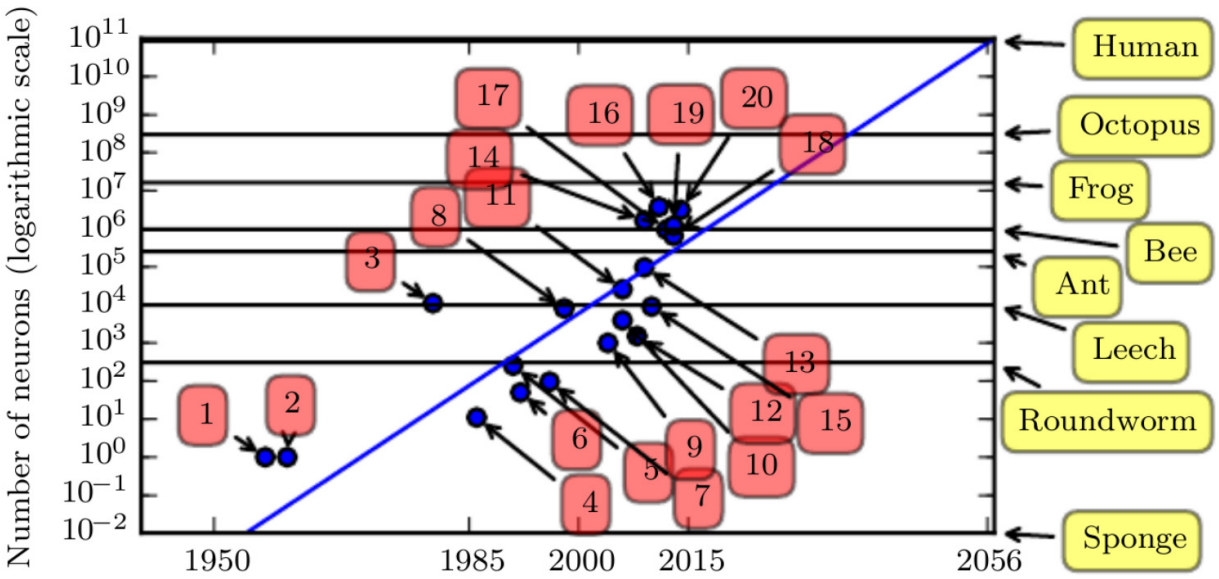
2006 Geoffrey Hinton - Deep Belief Networks (DBN) i rozwój pre-treningu

Współczesny rozwój głębokiego uczenia (2010-)

- 2012 Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton - AlexNet: znaczący spadek błędu w ImageNet (ok. 15%, wcześniej 30%), kluczowe zastosowanie GPU, bez pre-treningu
- 2013 Kingma, Welling - variational autoencoder (VAE)
- 2014 Ian Goodfellow - generative adversarial networks (GAN)
- 2015 DeepMind - Deep Q-Network (DQN), rozwój deep RL
- 2016 AlphaGo (DeepMind) - przełom w grach strategicznych, model pokonuje profesjonalnego gracza (Lee Sedol) w Go
- 2017 Ashish Vaswani (Google Brain) - transformery i rewolucja w przetwarzaniu języka naturalnego
- 2018 AlphaFold (DeepMind) - przewidywanie struktury białek
- 2020 GPT-3 (OpenAI)
- 2021 OpenAI, generator obrazów DALLE,
- 2022 Midjourney, Stable Diffusion, BARD/Gemini (Google AI)
- 2023 Llama (Meta), Grok (xAI)
- 2024 Sora (OpenAI) generowanie wideo, ...

☞ *The Evolution of Deep Learning: A Comprehensive Timeline*

Ilość neuronów w modelach neuronowych



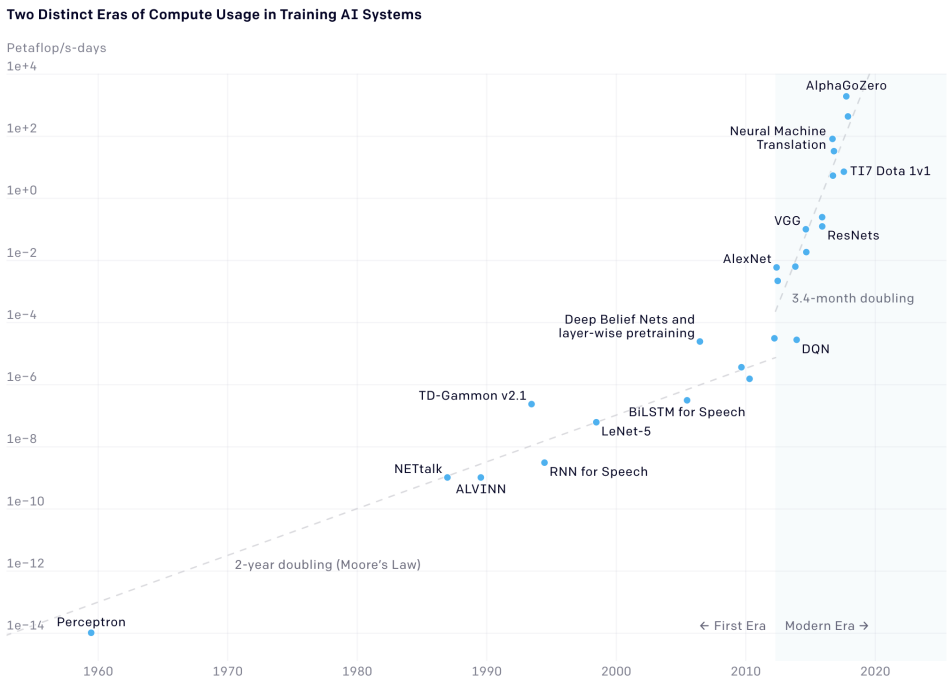
1. Perceptron (Rosenblatt, 1958, 1962)
2. Adaptive linear element (Widrow and Hoff, 1960)
3. Neocognitron (Fukushima, 1980)
4. Early back-propagation network (Rumelhart *et al.*, 1986b)
5. Recurrent neural network for speech recognition (Robinson and Fallsid
6. Multilayer perceptron for speech recognition (Bengio *et al.*, 1991)
7. Mean field sigmoid belief network (Saul *et al.*, 1996)
8. LeNet-5 (LeCun *et al.*, 1998b)
9. Echo state network (Jaeger and Haas, 2004)
10. Deep belief network (Hinton *et al.*, 2006)
11. GPU-accelerated convolutional network (Chellapilla *et al.*, 2006)
12. Deep Boltzmann machine (Salakhutdinov and Hinton, 2009a)
13. GPU-accelerated deep belief network (Raina *et al.*, 2009)
14. Unsupervised convolutional network (Jarrett *et al.*, 2009)
15. GPU-accelerated multilayer perceptron (Ciresan *et al.*, 2010)
16. OMP-1 network (Coates and Ng, 2011)
17. Distributed autoencoder (Le *et al.*, 2012)
18. Multi-GPU convolutional network (Krizhevsky *et al.*, 2012)
19. COTS HPC unsupervised convolutional network (Coates *et al.*, 2013)
20. GoogLeNet (Szegedy *et al.*, 2014a)

Prawo Moore'a dla sieci neuronowych:
liczba neuronów (paramtrów) podwaja się co 2.5 roku

Goodfellow, 2016 [5]

Era głębokiego uczenia

- Modern DL era: od 2012 roku, Alexnet, dostępność wydajnego sprzętu (GPU, TPU) umożliwia trenowanie modeli z milionami parametrów
- zmiana prawa Moore'a: podwojenie wydajności co 3.4 miesiąca

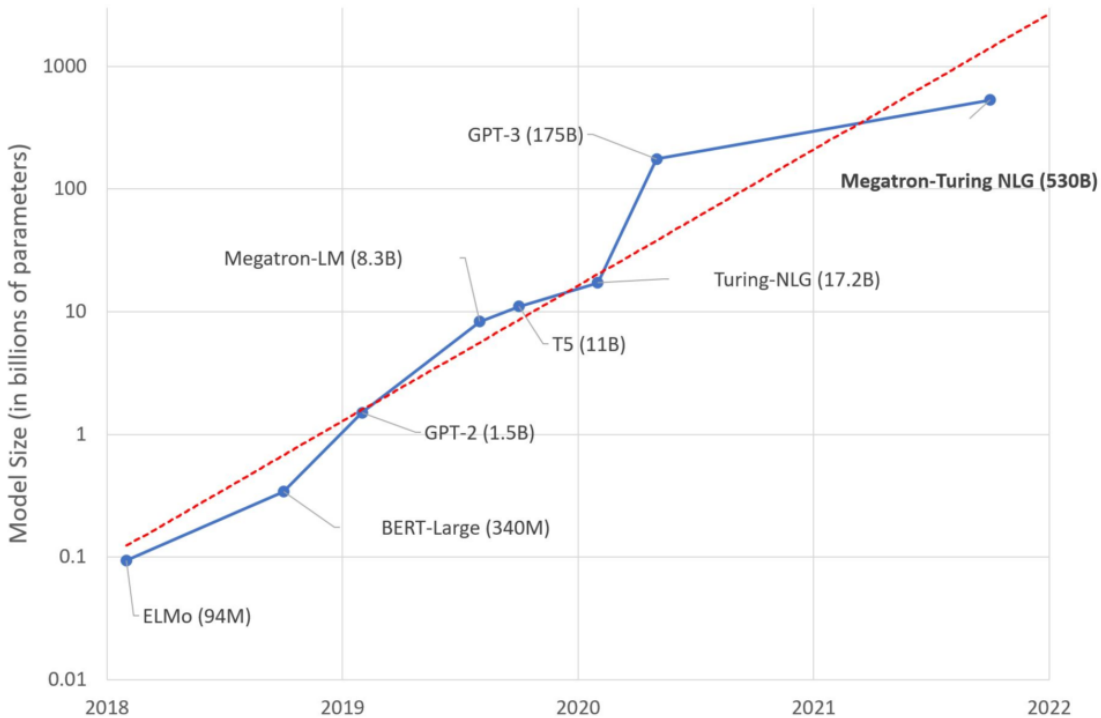


☞ *Computation used to train notable artificial intelligence systems*

☞ *AI and Scaling the Compute for the new Moore's Law*

Era dużej skali

- Large scale era: od 2016 roku, pojawienie się AlphaGo, masowe zrównoleglenie obliczeń, trenowanie na bardzo dużych zbiorach danych, algorytmy wyszukiwania architektur (NAS), expert iteration (ExIt) i inne techniki skalowania



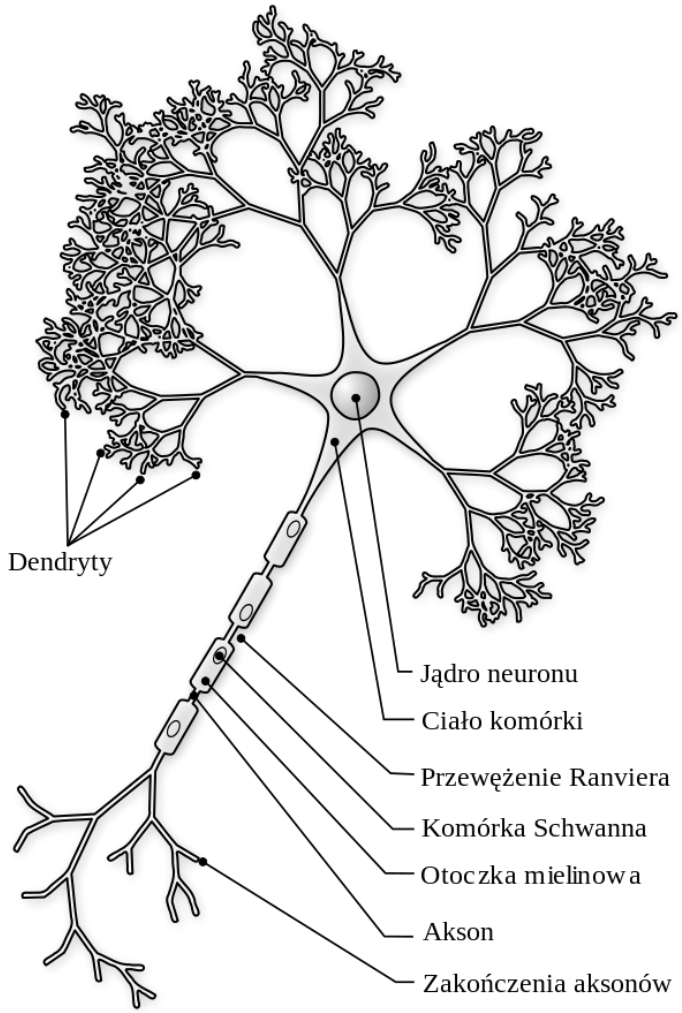
☞ *Computation used to train notable artificial intelligence systems*

☞ *AI and Scaling the Compute for the new Moore's Law*

Mózg i sieć neuronowa

- około 10^{11} neuronów (100 miliardów), do 100 Hz
- gęstość połączeń rzędu 10^4 synaps na neuron
- Receptory - dostarczają sygnały wejściowe (bodźce wewnętrzne i sygnał ze zmysłów)
- Sygnał wyjściowy z układu nerwowego steruje efektorami (reakcja na bodziec)
- obliczenia równoległe
- układy biologiczne są zbyt złożone do bezpośredniego modelowania - jedynie ogólne zasady organizacji i sposobu funkcjonowania układów biologicznych neuronów mogą służyć za inspirację
- sztuczne sieci neuronowe to uproszczone modele matematyczne, które można widzieć jako złożenie wielu (nieliniowych) funkcji i często nie mają nic wspólnego z sieciami biologicznymi

Neuron biologiczny



Rysunek: wikipedia.org

Neuron biologiczny

- **Dendryty** - wejście neuronu, odbierają sygnał od sąsiadujących neuronów, impulsy mogą być pobudzające lub hamujące
- **Ciało komórki (soma)** - akumuluje napływające w danym momencie sygnały
- **Akson** (wyjście) - emituje sygnał, gdy potencjał somy osiągnie próg
- **Synapsy** - styki pomiędzy dendrytami i aksonami; pobudzenie z aksonu powoduje uwolnienie chemicznych neurotransmiterów, które wpływają na zachowanie połączeń neuronów
- Działanie neuronów biologicznych jest bardzo złożonym procesem; neurony w sztucznych sieciach są dużym uproszczeniem

Logika progowa neuronu

Neuron zbiera dochodzące do niego sygnały x_i obliczając sumę ważoną tych sygnałów (wagi w_i określane są przez procesy synaptyczne), tworząc z nich sumaryczny sygnał wejściowy:

$$I(t) = \sum_i w_i x_i(t)$$

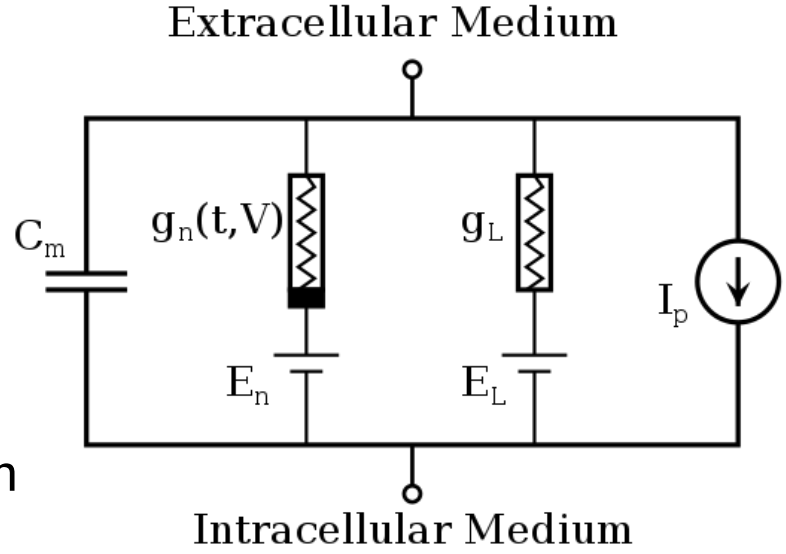
Sumowanie przestrzenne i czasowe potencjałów zgromadzonych na błonie komórkowej daje całkowitą aktywację elementu:

$$a(t) = F\big(I(t), a(t-1)\big)$$

Aktywacja, razem z progiem wzbudzenia T, określa sygnał wyjściowy

$$o(t) = f(a(t), T)$$

Model Hodgkin–Huxley (1952)



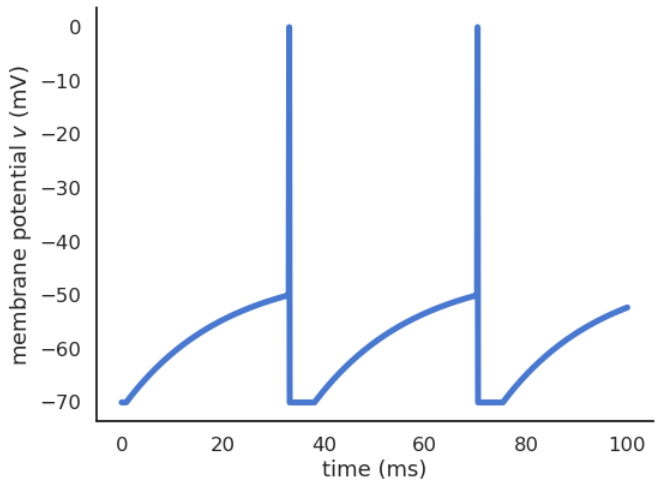
- pierwszy szczegółowy model matematyczny neuronu oparty na przewodnictwie jonowym
- nieliniowe równania różniczkowe opisujące zmianę potencjału błonowego $V(t)$, z uwzględnieniem prądów jonowych sodu (Na), potasu (K) i prądu upływu (L) oraz prądu zewnętrznego $I(t)$.

$$C_m \frac{\mathrm{d}V(t)}{\mathrm{d}t} = -g_K\big(V(t) - V_K\big) - g_{Na}\big(V(t) - V_{Na}\big) - g_L\big(V(t) - V_L\big) - I(t)$$

https://en.wikipedia.org/wiki/Hodgkin-Huxley_model

Neurony impulsujące (*spiking neurons*)

Sumuj i odpal: *Leaky integrate-and-fire* (LIF; Lapicque, 1907) - model generujący impuls, gdy sumaryczny potencjał membrany przekroczy próg



$$C_m \frac{\mathrm{d}V(t)}{\mathrm{d}t} = -g_L\big(V(t) - V_L\big) + I(t)$$

Gdy $V > V_T$ (próg wzbudzenia), neuron emituje impuls i następnie przechodzi w okres refrakcji (odpoczynku)

Źródło: Vitay, Neurocomputing [7]

Neurony kodowane częstością wzbudzeń

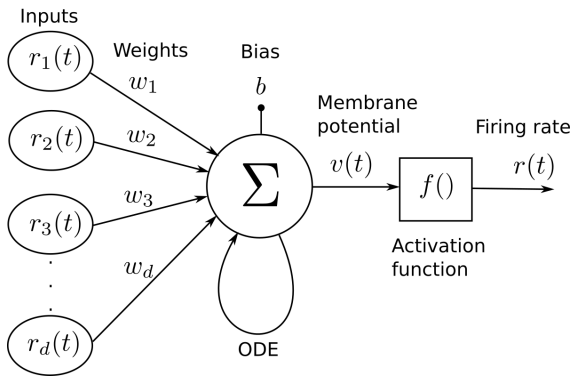
Rate-coded neurons

- potencjał membrany $V(t)$

$$\tau \frac{\mathrm{d}V(t)}{\mathrm{d}t} + V(t) = \sum_{i=1}^{d} w_{i,j} r_i(t) + b$$

- częstość wzbudzeń (*firing rate*) $r(t)$ odwzorowuje potencjał membrany w pojedynczą ciągłą wartość za pomocą **funkcji aktywacji**

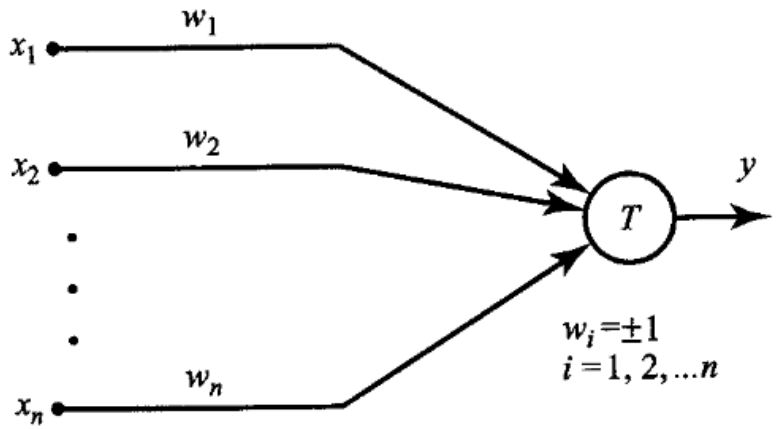
$$r(t) = f(V(t))$$



- nie symuluje pojedynczych impulsów (jak np. LIF), ale skupia się na średniej aktywności

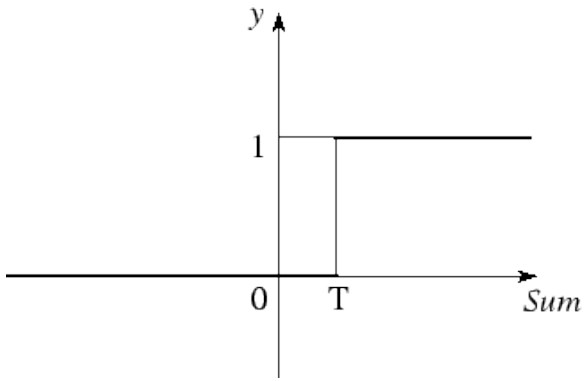
Źródło: Vitay, Neurocomputing [7]

Neuron McCulloch, Pitts (1943-49)



stan wyjściowy neuronu

$$o(\mathbf{x}) = \begin{cases} 1 & \text{gdy} \quad \sum_{i=1}^{n} w_i x_i \geq T \\ 0 & \text{gdy} \quad \sum_{i=1}^{n} w_i x_i < T \end{cases}$$



grafika: www.wikipedia.org

Neuron McCulloch, Pitts (1943-49)

- neurony mogą być tylko w dwóch stanach, aktywne albo nieaktywne, sygnał wejściowy i wyjściowy binarny

$$x_i \in \{0, 1\} \qquad o(\mathbf{x}) \in \{0, 1\}$$

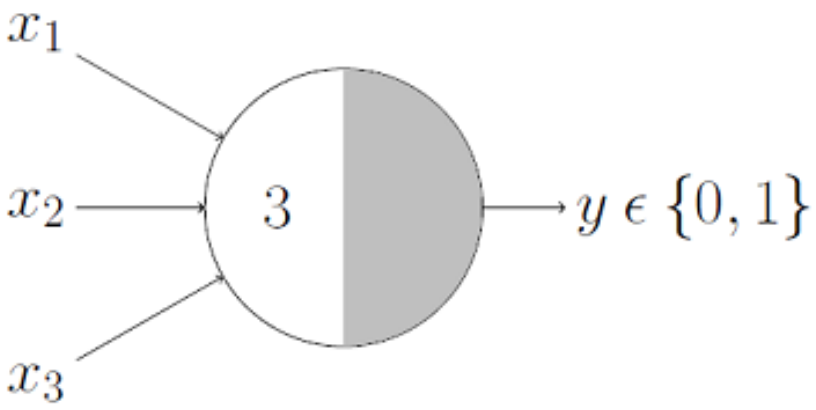
- sygnały dochodzą przez synapsy pobudzające i hamujące

$$w_{ij} \in \{-1, +1\}$$

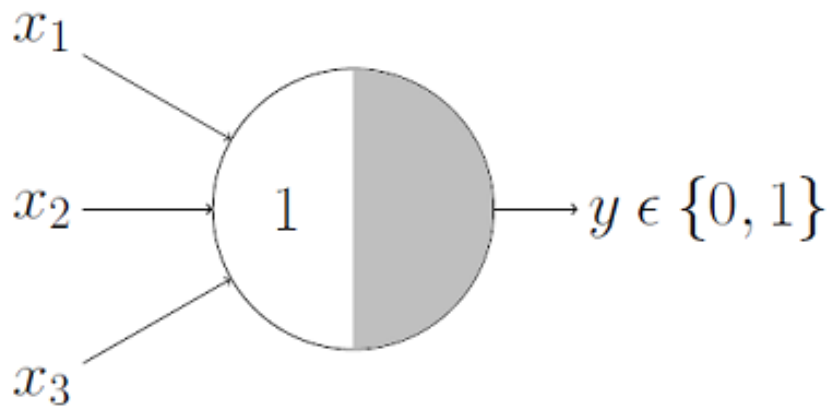
- każdy neuron ma ustalony próg pobudzenia $T > 0$
- sygnały sumują się w pewnym kwancie czasu i gdy przekroczą próg T to neuron się aktywuje
- brak uczenia się - wagi i próg mają stałą wartość
- odpowiedni dobór wag pozwala zrealizować bramki logiczne: NOT, OR, AND bądź NOR i NAND $\rightarrow$ sieć neuronów może implementować dowolną funkcję logiczną

Realizacja funkcji logicznych

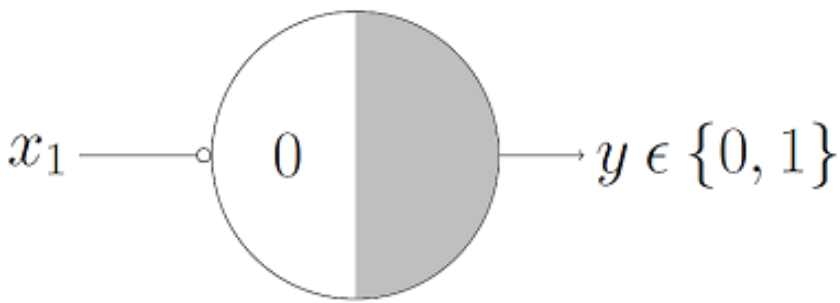
AND



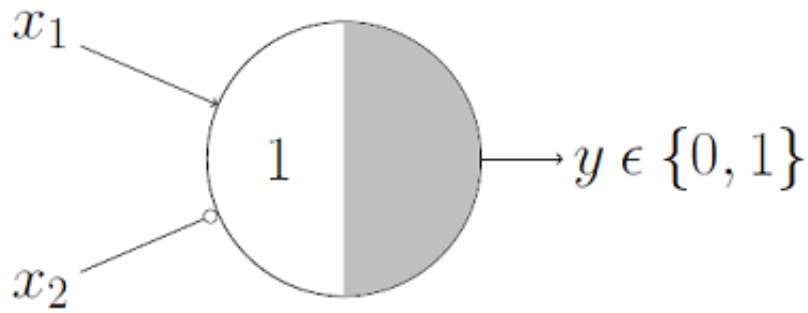
OR



NOT



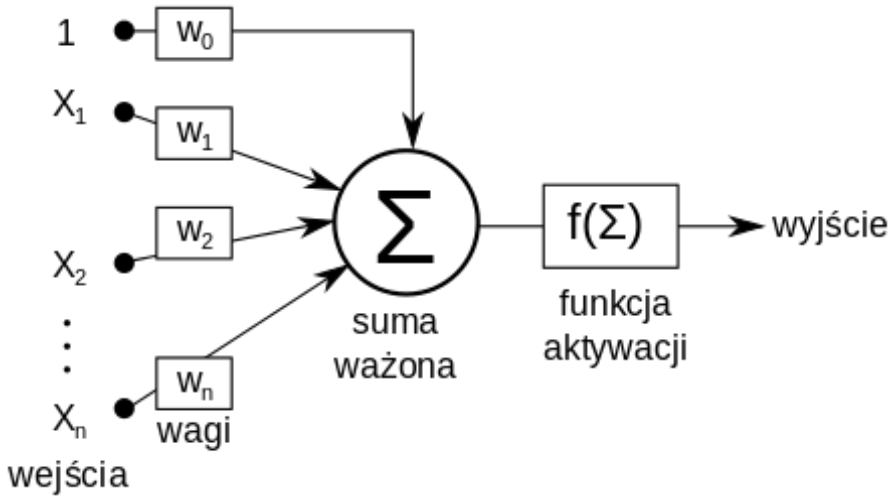
hamowanie sygnału wejściem x_2



$x_1\ AND\ !x_2$*

Źródło: Akshay L. Chandra, ☞ McCulloch-Pitts Neuron

Ogólny schemat neuronu



$$o(\mathbf{x}) = f\left(\sum_{i=1}^{n} w_i x_i + w_0\right) = f(\mathbf{w}^\top \mathbf{x})$$

gdzie

$\mathbf{x} = [1, x_1, x_2, \ldots, x_n]^\top$ wielowymiarowy sygnał wejściowy

$\mathbf{x} \in \mathbb{R}^{n+1}$

$\mathbf{w} = [w_0, w_1, w_2, \ldots, w_n]^\top$ wagi połączeń $\mathbf{w} \in \mathbb{R}^{n+1}$

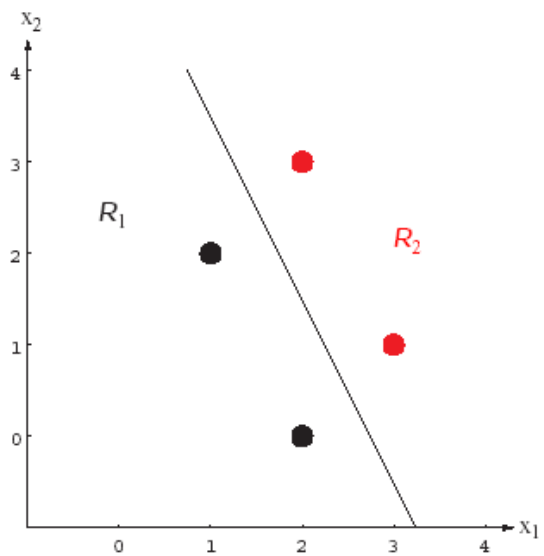
w_0 - wyraz wolny (bias), stały sygnał wejściowy $1 \cdot w_0$

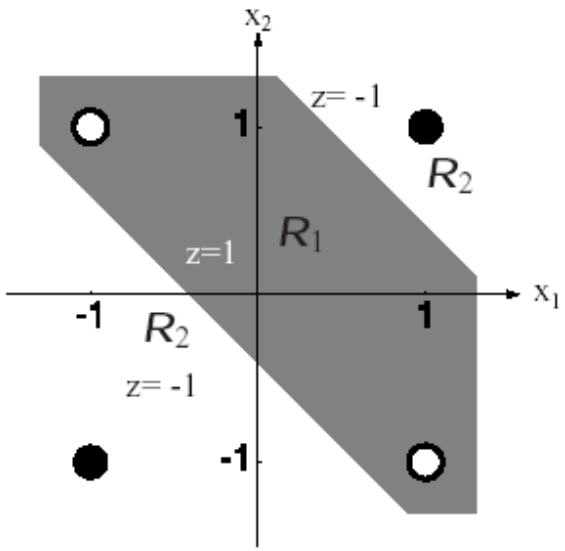
Interpretacja geometryczna - separowalność

Neuron progowy realizuje separowalność liniową: hiperpłaszczyzna dzieli przestrzeń wejściową $\mathbb{R}^n$ na dwie części

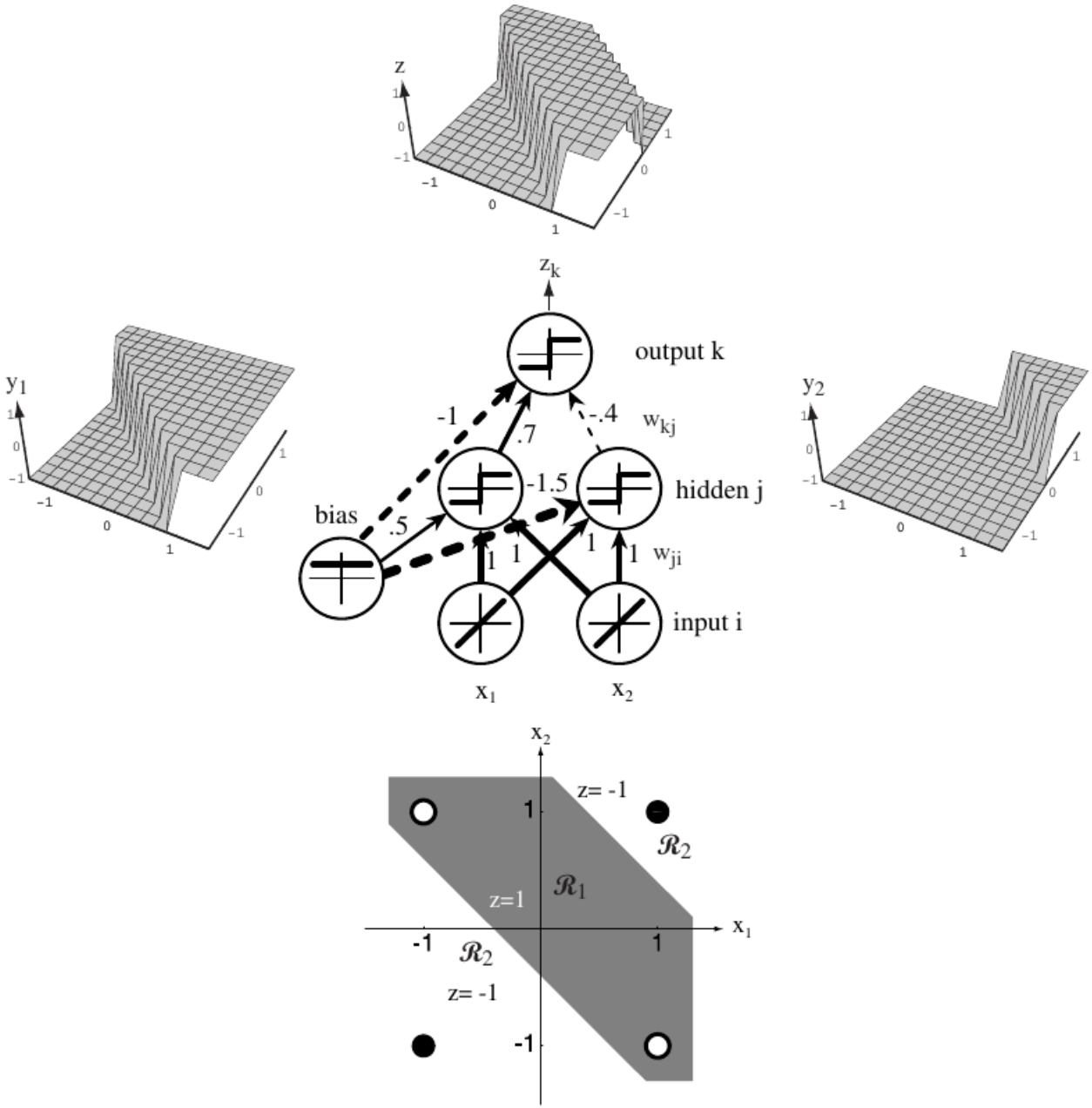
Przykład neuronu z 2 wejściami

$$o(\mathbf{x}) = \begin{cases} 1 & \text{gdy} \quad w_1 x_1 + w_2 x_2 \geq \theta \\ 0 & \text{gdy} \quad w_1 x_1 + w_2 x_2 < \theta \end{cases}$$





XOR


z
y1
y2
zk
output k
-1
.7
-.4
wkj
-1.5
hidden j
bias
.5
1
1
1
1
wji
input i
x1
x2
x2
z= -1
1
R2
R1
z=1
x1
-1
1
R2
z= -1
-1

Reguła Hebba (1949)

$$\Delta w_i = \eta \cdot o(\mathbf{x}) \cdot x_i = \eta f(\mathbf{w}^\top \mathbf{x}) x_i$$

„Kiedy akson komórki A jest dostatecznie blisko by pobudzić komórkę B i wielokrotnie w sposób trwały bierze udział w jej pobudzaniu, procesy wzrostu lub zmian metabolicznych zachodzą w obu komórkach tak, że sprawność neuronu A jako jednej z komórek pobudzających B, wzrasta."

,,neurons that fire together wire together"

Plastyczność układu (uczenie)

Uczenie się: zmiana wag synaptycznych w_i w czasie

Reguła Hebba

$$\Delta w_i = \eta o(\mathbf{x}) x_i = \eta f(\mathbf{w}^T \mathbf{x}) x_i$$

Reguła delta (Widrowa-Hoffa)

$$\Delta w_i = \eta (y - o(\mathbf{x})) x_i$$

gdzie y to spodziewana odpowiedź dla sygnału wejściowego $\mathbf{x}$

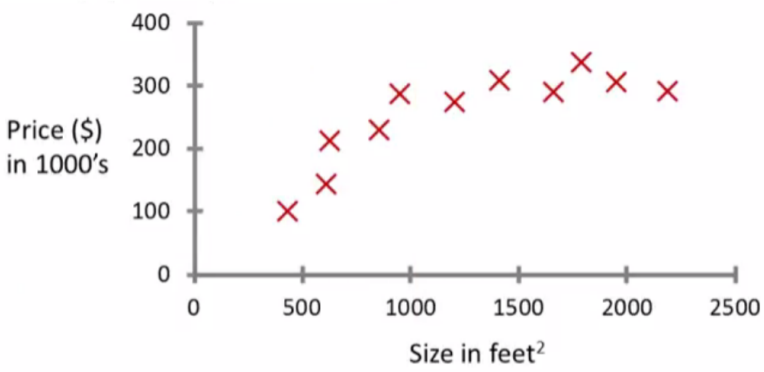
Rodzaje ucznia

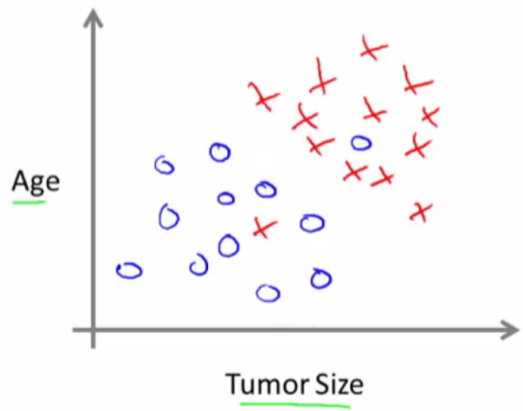
- **Uczenie bez nadzoru** (*unsupervised learning*)
  uczenie wyłącznie na podstawie sygnału wejściowego **X** (bez etykiet **y**), odkrywanie struktur i zależności w danych.
  Uczenie spontaniczne, odpowiada uczeniu w okresie niemowlęcym.
- **Uczenie nadzorowane** (*supervised learning*) - dla każdego sygnału wejściowego **x** dana jest pożądana odpowiedź **y**.
  Uczenie „szkolne".
- **Uczenie z krytykiem**, ze „wzmocnieniem" (*reinforcement learning*) - uczenie się zachowań, które przynoszą zysk po dłuższym czasie (np. autonomiczne roboty, gra z przeciwnikiem).
  Uczenie dojrzałe (nabieranie „mądrości").

Uczenie nadzorowane

Każdy przypadek **x** ze zbioru treningowego posiada przypisaną wartość **y**

- **Regresja (aproksymacja)** - obiekt wyjściowy **y** jest liczbą rzeczywistą lub wektorem liczb (np. temperatura)
- **Klasyfikacja** - obiekt wyjściowy **y** jest etykietą klasy (np. nazwa obiektu na zdjęciu)





Rysunek: Mohamad Ghassany, http://www.mghassany.com/MLcourse/introduction.html

Uczenie nadzorowane

- Model

$$f(\mathbf{x}; \mathbf{W}) = \mathbf{y}$$

  realizuje mapowanie zdefiniowane przez parametry sieci $\mathbf{W}$ (wagi połączeń) wektora wejściowego $\mathbf{x}$ do wektora wyjściowego $\mathbf{y}$ (pożądanego wyjścia)
- Uczenie najczęściej polega na takim doborze parametrów $\mathbf{W}$ aby zminimalizować różnice pomiędzy wyjściem sieci $f(\mathbf{x})$ a wartością pożądaną $\mathbf{y}$, np. błąd kwadratowy:

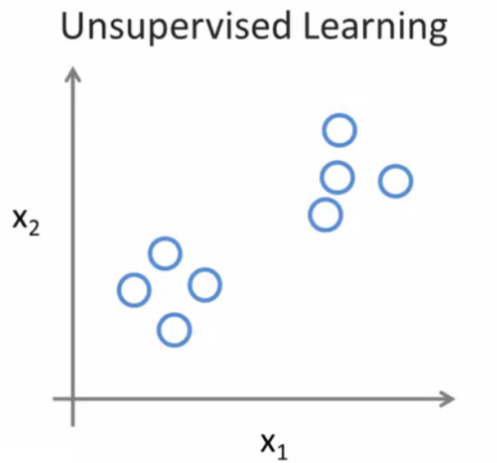
$$\|f(\mathbf{x}; \mathbf{W}) - \mathbf{y}\|^2$$

- Celem uczenia nadzorowanego nie jest uczenie „na pamięć”, lecz **generalizacja**.

Uczenie nienadzorowane

Uczenie wyłącznie na podstawie sygnału wejściowego **X**, brak pożądanego sygnału wyjściowego. Przykłady:

- **klasteryzacja** (analiza skupień), automatyczne wykrywanie grup o podobnych właściwościach

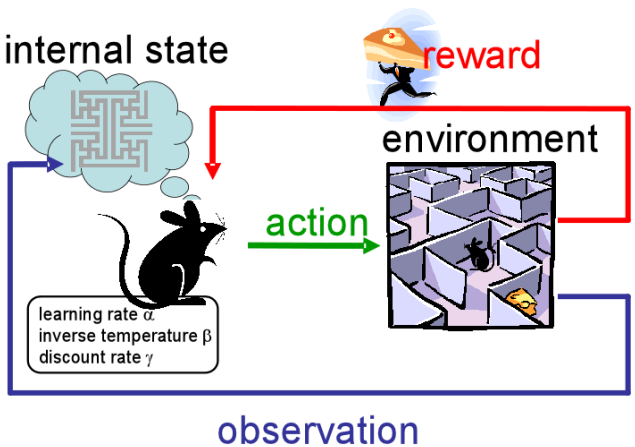


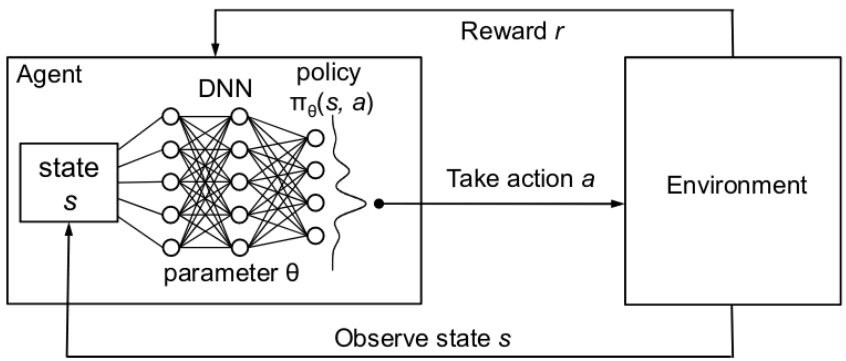
- wykrywanie cech i regularności, tworzenie przydatnych reprezentacji danych, kompresja (kodowanie) sygnału, redukcja wymiarowości, modelowanie sygnału, redukcja szumu

Rys.: Mohamad Ghassany, http://www.mghassany.com/MLcourse/introduction.html

Uczenie z krytykiem

- Sygnał wyjściowy to zazwyczaj akcja (sekwencja akcji) podejmowana przez agenta
- Nagroda (lub kara) jest dostarczana z opóźnieniem (np. przegrana/wygrana partia w szachy)

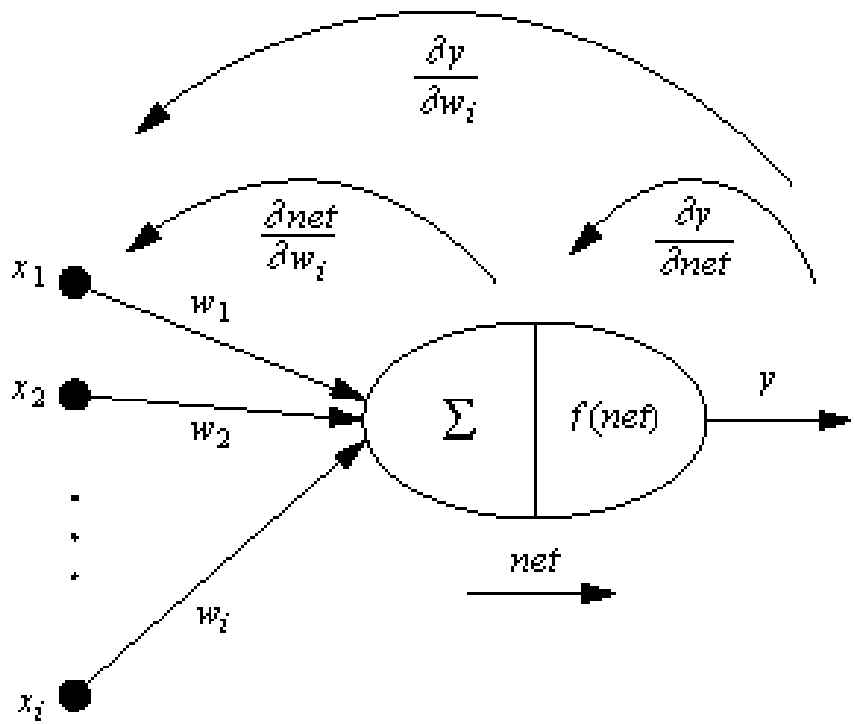




Rysunek: Hongzi Mao, „Resource Management with Deep Reinforcement Learning"
Aneek Das, The very basics of Reinforcement Learning

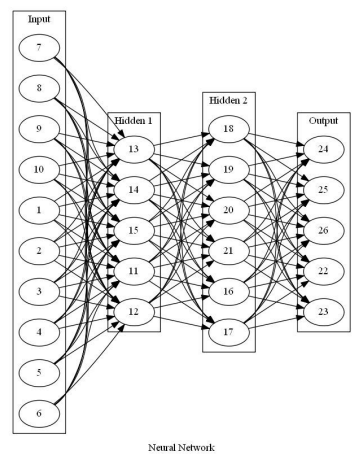
Główne aspekty modeli neuronowych I

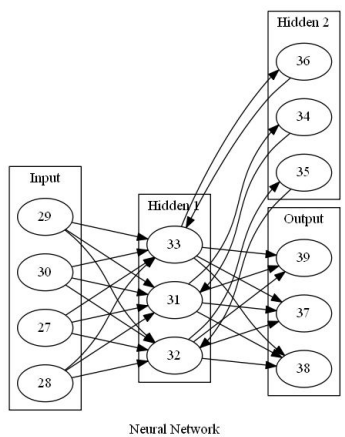
1. Sposób modelowania pojedynczego neuronu
   - stan aktywacji (wzbudzenia) poszczególnych neuronów, sposób aktywacji tych neuronów, funkcja opisująca sygnał wyjściowy elementu
   - neurony: progowe, liniowe, nieliniowe (np. sigmoidalne), ...

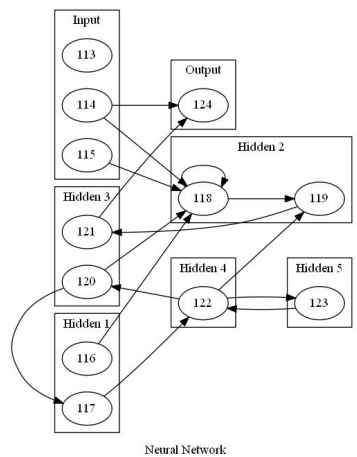


Główne aspekty modeli neuronowych II

2. Sposób propagacji sygnałów przez sieć neuronową
   - sieci jednokierunkowe (feedforward)
   - sieci ze sprzężeniami zwrotnymi, sieci rekurencyjne (recurrent), sieci dynamiczne
3. Topologia połączeń elementów (architektura sieci):
   - budowa warstwowa (warstwy wejściowe, ukryte, wyjściowe), struktura hierarchiczna
   - sieci jednowarstwowe, wielowarstwowe, płytkie, głębokie
   - warstwy w pełni połączone, splotowe, rzadkie połączenia, współdzielenie połączeń
   - sieci o stałej architekturze, sieci ontogeniczne (zmieniające strukturę), rozrastające się, kurczące się,







Główne aspekty modeli neuronowych III

4. Uczenie sieci
   - reguły uczenia, reguły modyfikacji parametrów opisujących neurony i połączeń pomiędzy nimi
   - algorytm optymalizacji, funkcja kosztu
   - regularyzacja
5. Otoczenie, w którym działa sieć neuronowa: reprezentacja danych, sposób prezentacji danych, jakość danych, dostępność danych, sposób oceny jakości
6. Realizacja techniczna: język programowania, wykorzystanie GPU, zrównoleglenie obliczeń