Sztuczne sieci neuronowe

Marek Grochowski

Plan

1. Perceptrony, sieci wielowarstwowe jednokierunkowe (MLP)
2. Metody uczenia sieci i algorytm wstecznej propagacji błędu
3. Radialne funkcje bazowe (RBF) i metody aproksymacji
4. Samoorganizacja, sieci SOM
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

Literatura I

[1] Osowski S., *Sieci neuronowe w ujęciu algorytmicznym*, Wydawnictwo Naukowo-Techniczne, Warszawa 1996

[2] Tadeusiewicz R., *Sieci neuronowe*, Akademicka Oficyna Wydawnicza RM, Warszawa 1993

[3] Ryszard Tadeusiewicz, Tomasz Gąciarz, Barbara Borowik, Bartosz Lepe, *Odkrywanie właściwości sieci neuronowych przy użyciu programów w języku C#*, Polska Akademia Umiejętności, 2008

[4] J. Żurada, M. Barski, W., *Jędruch Sztuczne sieci neuronowe*, Wydawnictwo Naukowe PWN 1996

Literatura II

[5] Ian Goodfellow, Yoshua Bengio and Aaron Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org

[6] Michael Nielsen, *Neural Networks and Deep Learning*, http://neuralnetworksanddeeplearning.com/

[7] Denny Britz, *Deep Learning Glossary*, http://www.wildml.com/deep-learning-glossary/

Inteligencja obliczeniowa

Computational Intelligence (CI) zajmuje się rozwiązywaniem problemów, które nie są efektywnie algorytmizowalne.
Problemy niealgorytmizowalne:

- zagadnienie jest zbyt złożone, np. problemy NP-trudne
- modelowany proces może zawierać trudne do zdefiniowania niejasności lub moze być stochastyczny z natury
- procesu nie da się opisać przez zrozumiałe zasady
- warunki mogą się zmieniać, algortym musi się dostosowac do nowyej sytuacji

Cechą wielu systemów CI jest rozwiązywanie zadań na podstawie znanych przykładów, uczenie się z empirycznych danych zamiast programowania rozwiązania. Systemy uczące się (machine learning, ML) oraz sztuczne sieci neuronowe (artificial neural networks, ANN) są ważnymi elementami CI.

Problemy efektywnie nialgorytmizowalne

Problemy NP-trudne – Liczba kroków algorytmu dla złożonych sytuacji rośnie w sposób szybszy niż jakikolwiek wielomian liczby elementów (złożoności specyfikacji problemu).

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)

Definicje AI i CI (W. Duch):
CI: percepcja i sterowanie: zachowania sensomotoryczne – sieci neuronowe i uczenie maszynowe;
AI: wyższe czynności poznawcze: logika, język, rozumowanie, rozwiązywanie problemów.

Artificial Intelligence (AI) to część CI posługująca się symboliczną reprezentacją wiedzy, zajmuje się rozumowaniem, tworzeniem systemów ekspertowych.

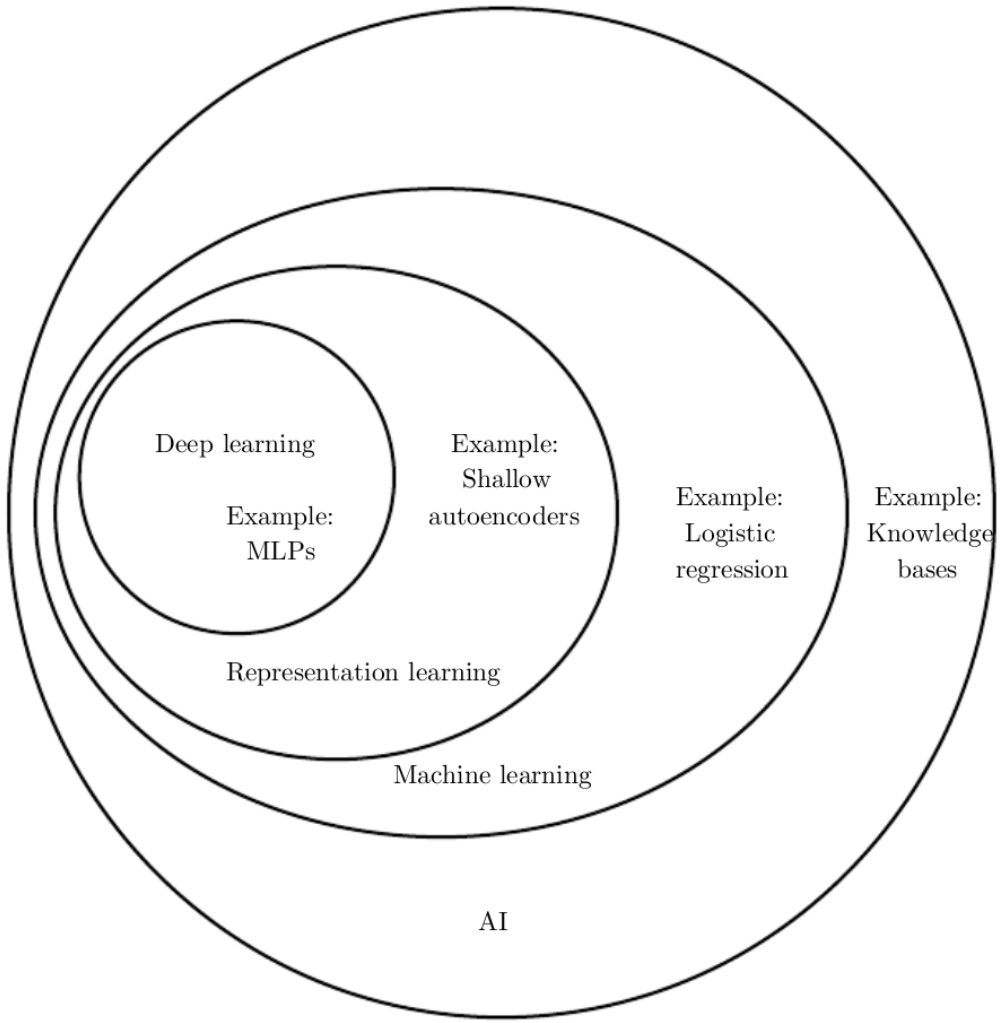
Obecnie „sztuczna inteligencja” używana jest na określenie wszystkiego, co się kojarzy z inteligencją maszyn.

Uczenie maszynowe

- **Uczenie maszynowe** (Machine learning, ML) - sytemy, które uczą się rozwiązywać problemy na podstawie dostarczonych przypadków uczących (zebranych danych)
- **Generalizacja** - dobrze wyuczony model działa poprawnie także dla przypadków, których nie było w danych treningowych
- Gdy dane się zmienią system można dostosować trenując go na nowych danych

Rysunek: https://xkcd.com/

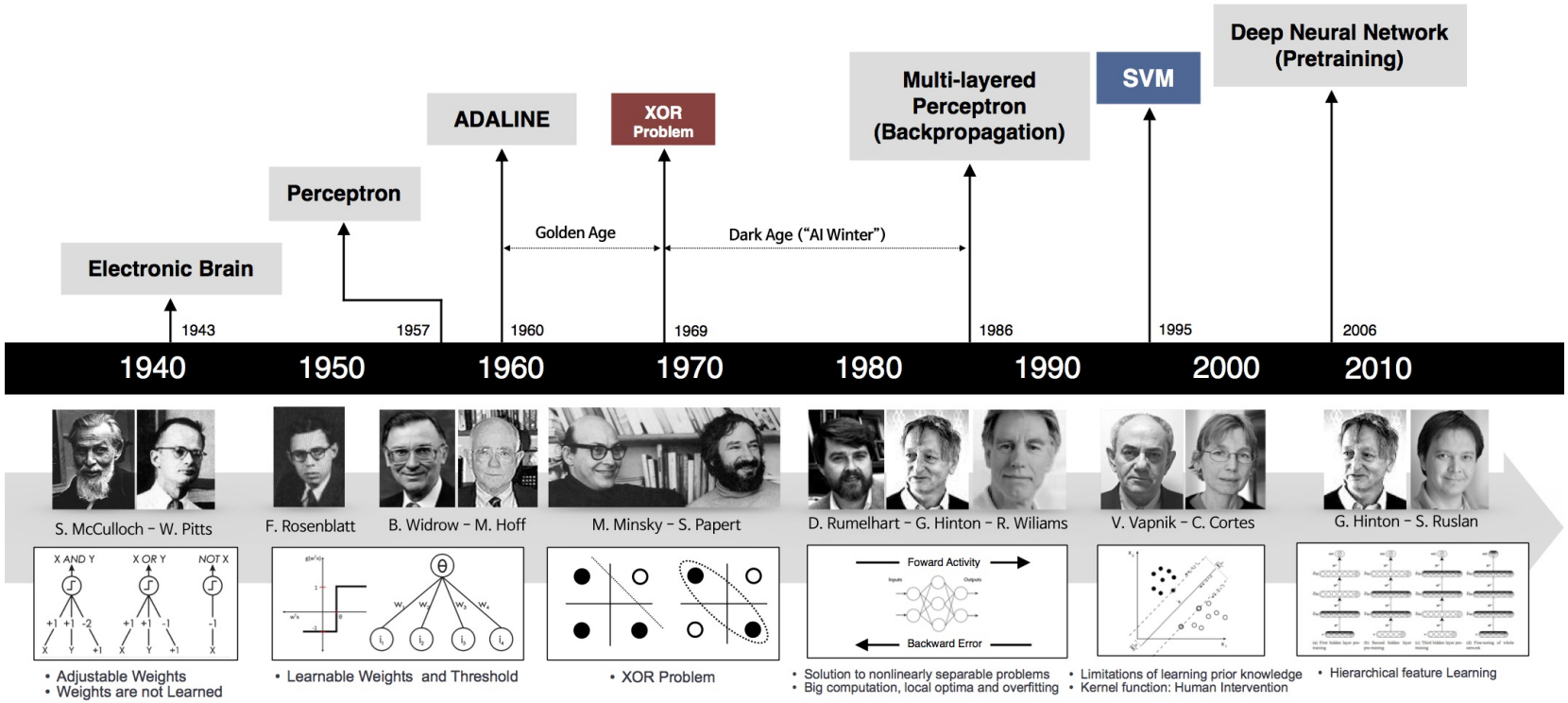
ANN i sztuczna inteligencja (AI)



Gooffellow, 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 za pomocą metod inspirowanych systemami biologicznymi -> Sztuczne Sieci Neuronowe (Artificial Neural Networks, ANN)



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

Rys historyczny I

1938 N. Rashevsky, neurodynamika - sieci neuronowe jako układ dynamiczny

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

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

1958 F. Rosenblatt, Perceptron - sieć jako funkcja matematyczna
J. von Neuman, The computer and the brain

1960 B. Widrow, M. Hoff, Adeline

1967 K. Steinbuch, E. Schmitt, Macierze uczące się

1969 M. Minsky, S. Papert, książka „Perceptrony” - wykazali ograniczenia sieci liniowych

Rys historyczny II

1973 Chr. von der Malsburg, samoorganizacja w układzie nerwowym

1982 J. Hopfield, model Hopfielda
T. Kohonen, samoorganizacja map topograficznych mózgu

1983 K. Fukushima, S. Miyake, T. Ito, neokognitron, głęboka sieć neurowa

1985 D. Ackley, G. Hinton, T. Sejnowski, maszyny Boltzmana

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

1987 G. Carpenter, S. Grossberg, model ART; W. Freeman, Chaos w mózgu

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

1995 V. Vapnik, SVM i „koniec ery Data Mining”

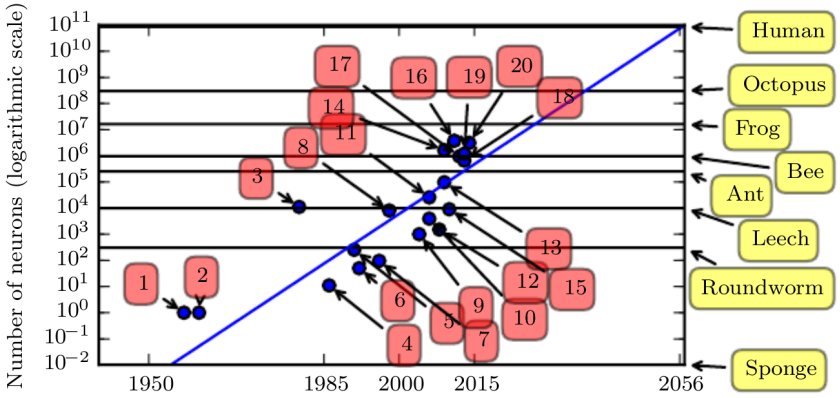
Rys historyczny III

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

1998 Y. LeCun i inn, sieci konwolucyjne w analizie obrazów.

2005 GPU do sieci konwolucyjnych, bardzo duże sieci, głębokie uczenie

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 Fallside, 1991)
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)

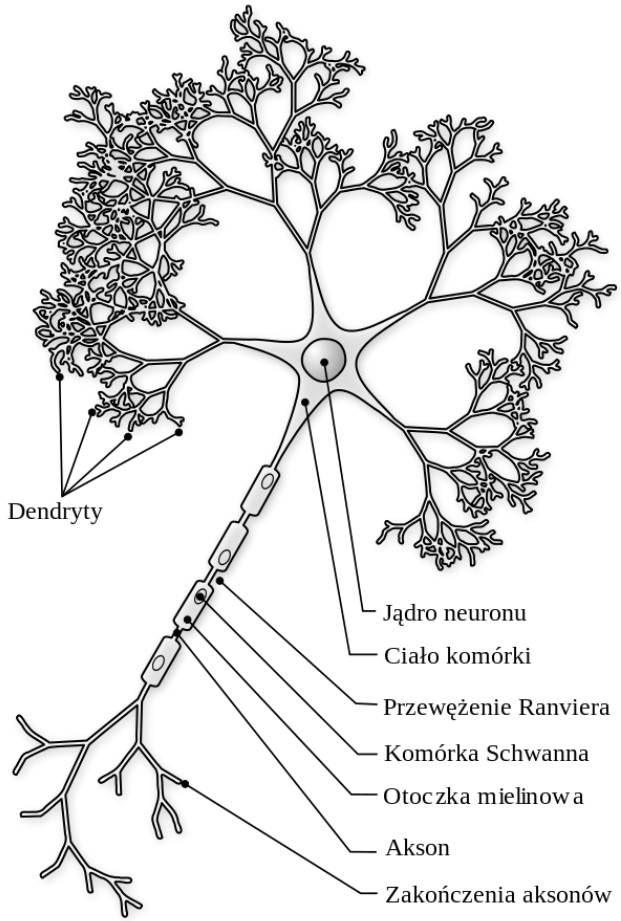
Liczba neuronów podwaja się co 2.5 roku

Gooffellow, 2016 [5]

Mózg i sieć neuronowa

- 10^{11} neuronów (100 miliardów), do 100Hz
- gęstość połączeń około 10^4 synaps na neuron
- Receproty - dostarczają sygnały wejsciowe (bodźce wewnętrzen i sygnał ze zmysłów)
- Sygnał wyjściowy z układu nerowego steruje efektorami (reakcja na bodziec)
- obliczenia równoległe
- układy biologiczne zbyt złożone - jedynie ogólne zasady organizacji i sposobu funkcjonowania układów biologicznych neuronów mogą służyć za inspirację
- Sztuczne sieci neuronowe to model matematyczny, który można widzieć jako złożenie wielu (nieliniowych) funkcji i często nie ma 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), sumuje napływające w danym momencie sygnały
- Akson (wyjście), emituje sygnał, gdy potencjał w somie osiągnie odpowiedni próg
- Synapsy - styki pomiędzy dendrytami i aksonami, pobudzenie napływające z axonu powoduje uwolnienie chemicznych neurotransmiterów, które wpływają na zachowanie połączeń przyłączonych neuronów
- Działanie neuronów biologicznych jest bardzo złożonym procesem, neurony w sztucznych sieciach są dużym uproszczeniem

☞ Schematic animation of spike generation in neural cell (J. Pineau)

Logika progowa

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

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

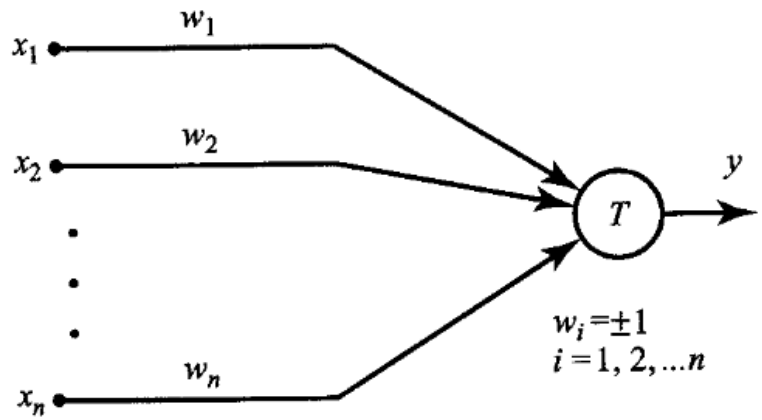
Sumowanie przestrzenne i czasowe potencjałów zgromadzonych na błonie komórkowej daje całkowitą aktywację elementu:

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

Aktywacja, razem z progiem T wzbudzenia, określa sygnał wyjściowy

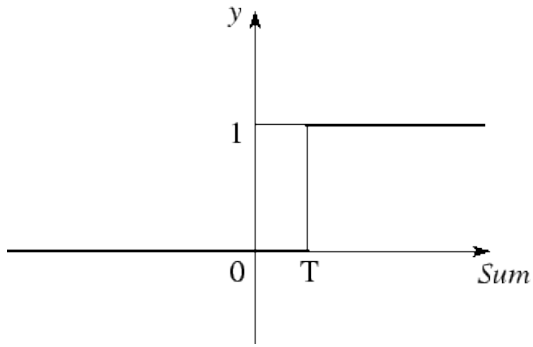
$$o(t) = f(a, T)$$

Neuron McCulloch. Pitts (1943-49)



wyjście neuronu w chwili $k+1$

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



grafika: www.wikipedia.org

Neuron McCulloch, Pitts (1943-49)

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

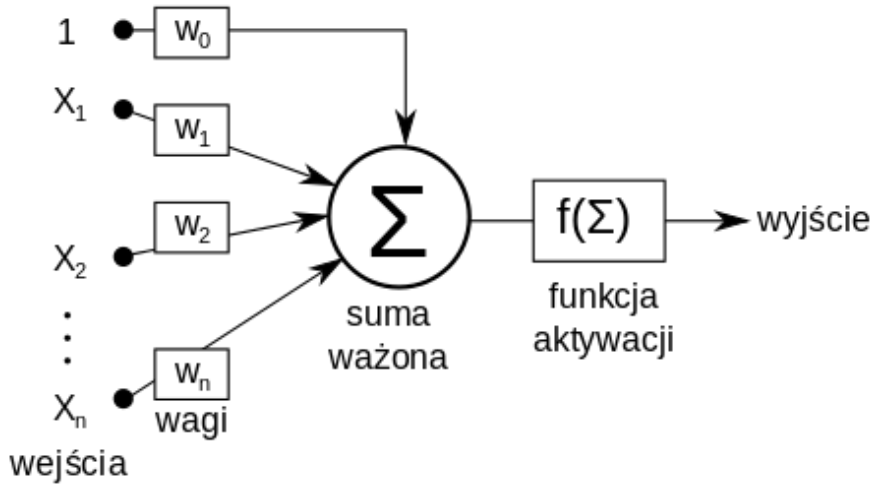
$$x_i = \{0, 1\}$$

- sygnały dochodzą przez synapsy pobudzające i hamujące

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

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

Ogólny schemat neuronu



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

gdzie

$\mathbf{x} = [1, x_1, x_2, \ldots, x_n]$

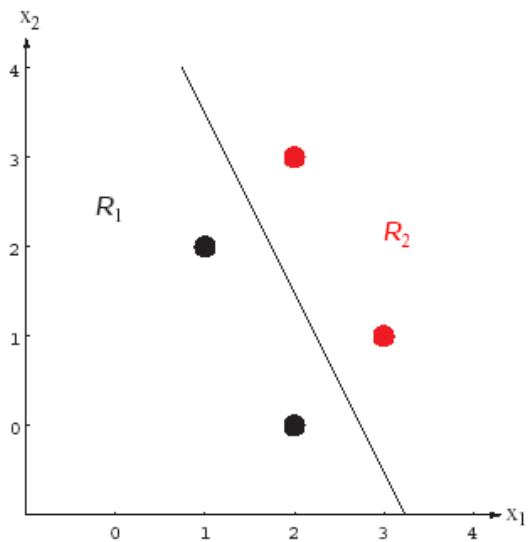
$\mathbf{w} = [w_o, w_1, w_2, \ldots, w_n]$

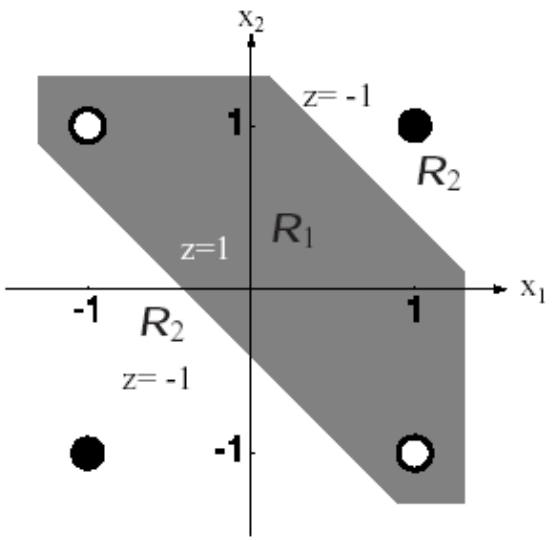
w_0 - wyraz wolny (bias)

Interpretacja geometryczna - separowalność

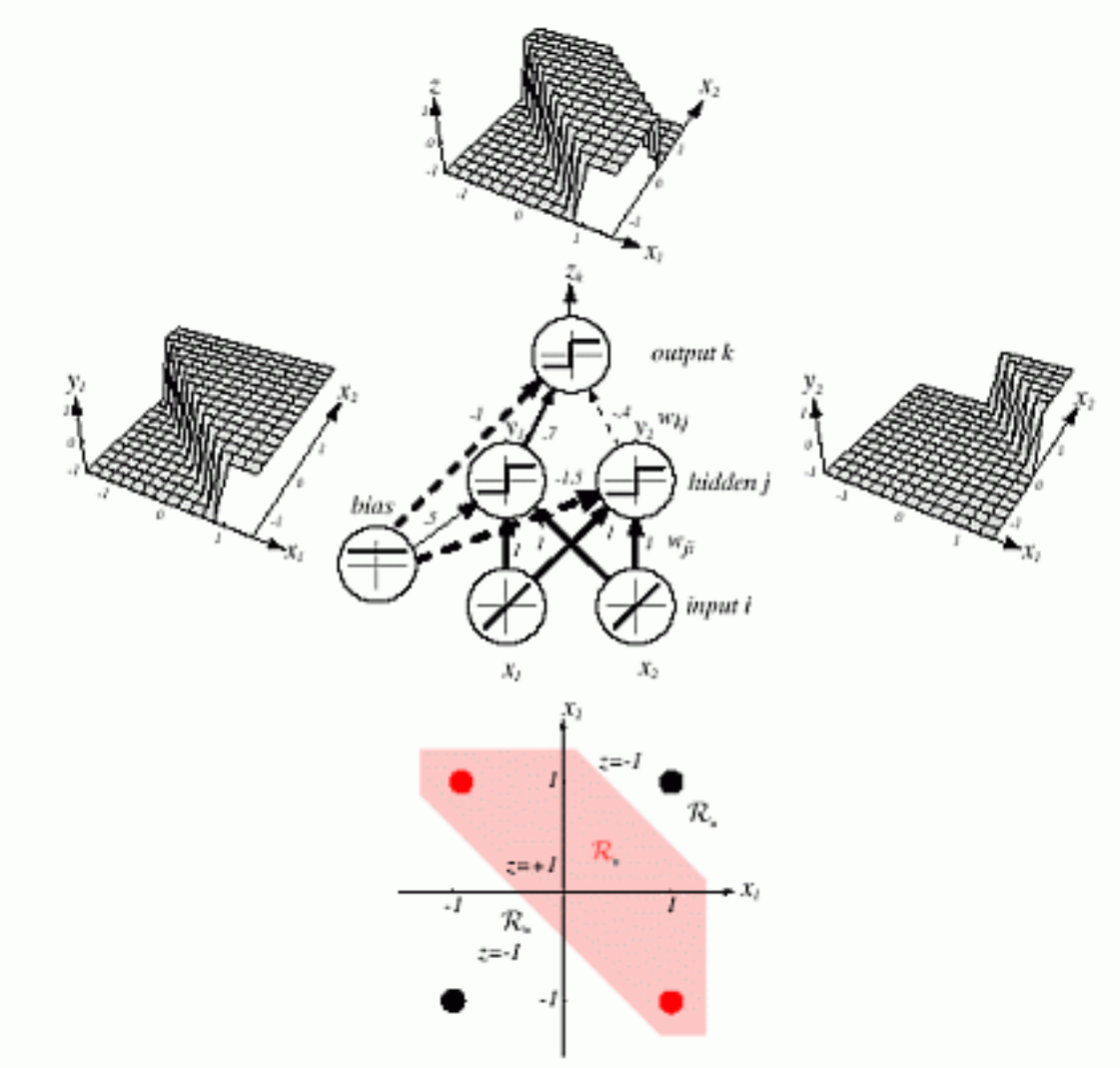
Neuron z progiem realizuje sepaowalność liniową, hiperpłaszczyzna dzieli przestrzeń na dwie części

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





XOR



Reguła Hebba (1949)

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

„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.”

Uczenie

Plastyczność układu (uczenie się): wagi synaptyczne w_i zmieniają się powoli w czasie.

Reguła Hebba

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

Reguła delta (Widrowa-Hoffa)

$$\Delta w_i = \eta(y - o) x_i$$

gdzie y to sygnał nagrody

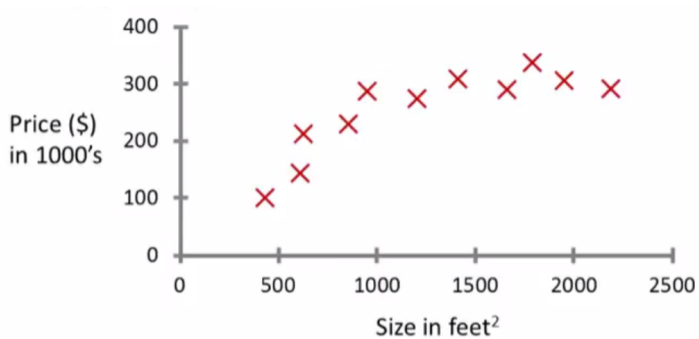
Rodzaje ucznia

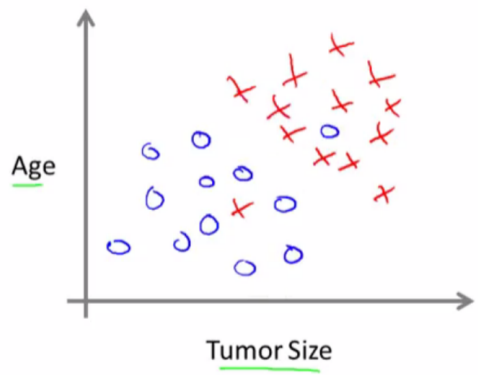
- **Uczenie bez nadzoru** (unsupervised learning)
  uczenie wyłącznie na podstawie sygnały wejściowego **X**, odkrywanie ciekawych struktur w przestrzeni danych wejściowych.
  Uczenie spontaniczne, odpowiada uczeniu w okresie niemowlęcym.
- **Uczenie nadzorowane** (supervised learning) - dla każdego sygnału wejściowego **x** dana jest pożądana odpowiedź **y**.
  Uczenie „szkolne".
- **Uczenie z krytykiem**, z „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 jest **x** jest etykietą klasy (np. nazwa obiektu na zdjęciu)





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

Uczenie nadzorowane

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

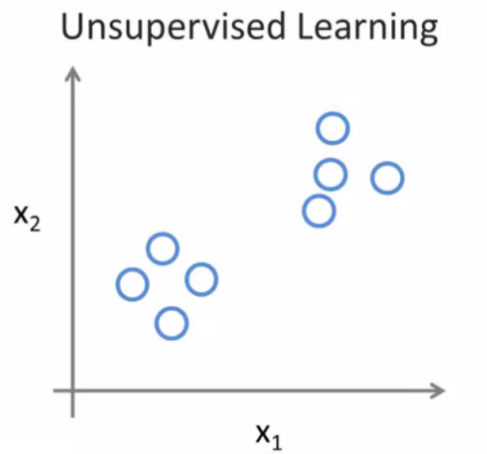
$$(f(\mathbf{x}; \mathbf{W}) - \mathbf{y})^2$$

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

Uczenie nienadzorowane

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

- **klasteryzacja** (analiza skupień) oparta na jakiejś mierze podobieństwa, szukanie struktur.

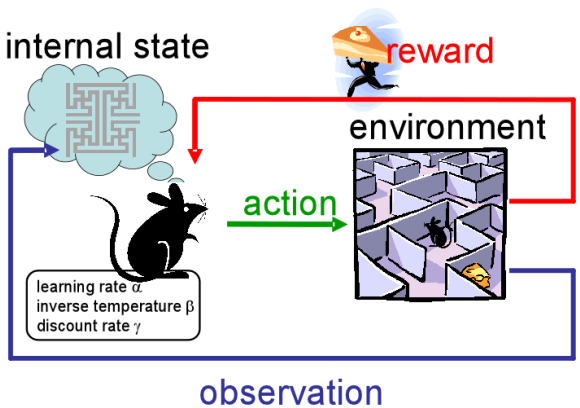


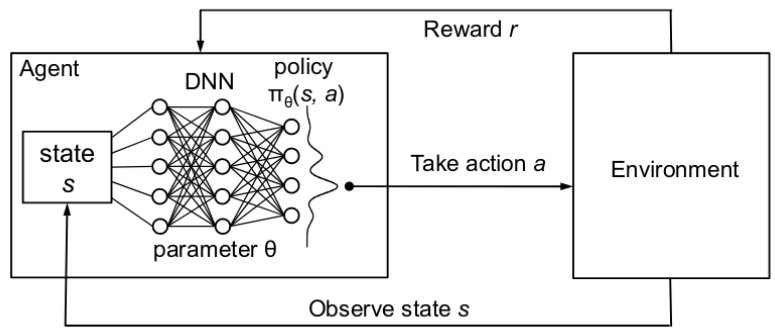
- Wykrywanie cech i regularności w danych, tworzenie przydatnych reprezentacji danych, kompresja (kodowanie) sygnału, redukcja wymiarowości, modelowanie sygnału, redukcja szumu

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

Uczenie z krytykiem

- Sygnał wyjściowy jest zazwyczaj akcją (sekwencją akcji) podejmowanych przez agenta
- Nagroda (lub kara) jest dostarczana z opóźnieniem (np. przegrana partia w szachy)

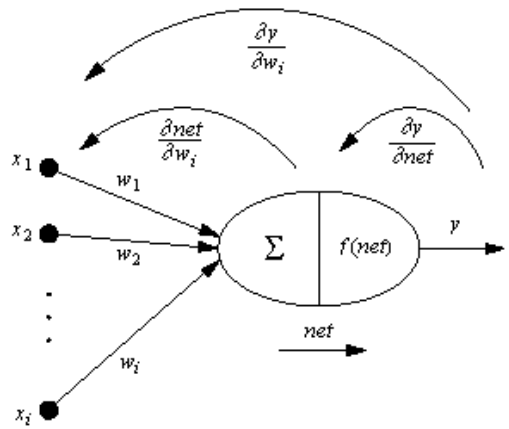




Rysunek: Hongzi Mao, „Resource Management with Deep Reinforcement Learning"
Aneek Das, The very basics of Reinforcement Learning

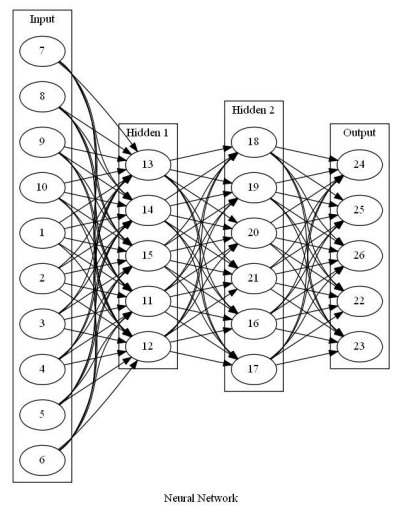
Główne aspekty modeli neuronowych I

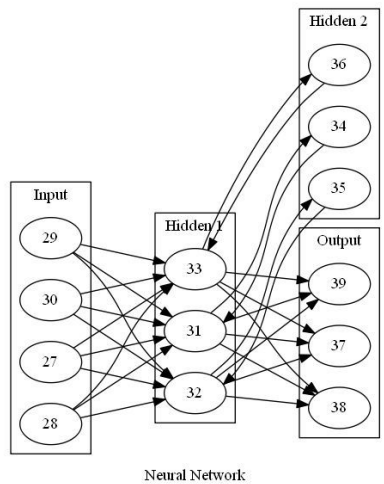
1. Sposób modelowania pojedynczego neuronu
   - Stan aktywacji (wzbudzenia) poszczególnych neuronów, sposób aktywacji tych neuronów, funkcja opisująca sygnał wyjściowy elementu,
   - neurony: progowe, liniowe, nieliniowe (np. sigmoidalne)

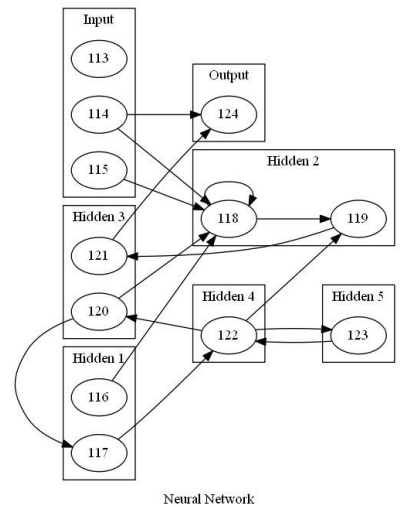


Główne aspekty modeli neuronowych II

2. Sposób propagacji sygnałów przez sieć neuronową
   - sieci jednokierunowe (feedforward)
   - sieci ze zprzężeniami zwrotnymi, seici rekurencyjne (reccurent)
3. Topologia połączeń elementów (architektura sieci):
   - budowa warstwowa (warstwa wejściowa, wyjściowa, warstwy ukryte), hierarchiczna
   - sieci jednowarstwowe, wielowartswowe, głebokie







Główne aspekty modeli neuronowych III

4. Reguły uczenia - reguły modyfikacji parametrów opisujących neurony i połączenia pomiędzy nimi, algorytm optymalizacji, funkcja kosztu
5. Otoczenie, w którym działa sieć neuronowa: sposób prezentacji danych, rodzaj danych wejściowych, reprezentacja sygnału wyjściowego (np. kodowanie etykiet klas)
6. Realizacja techniczna: język programowania, wykorzystanie GPU, zrównoleglenie obliczeń