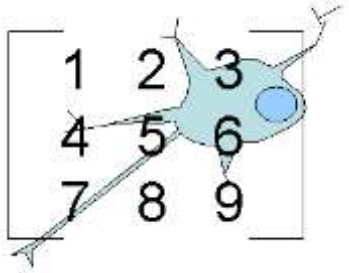
Wydział Zarządzania AGH
Katedra Informatyki Biznesowej i Inżynierii Zarządzania

Sztuczne sieci neuronowe

Inteligencja obliczeniowa

Sztuczne sieci neuronowe

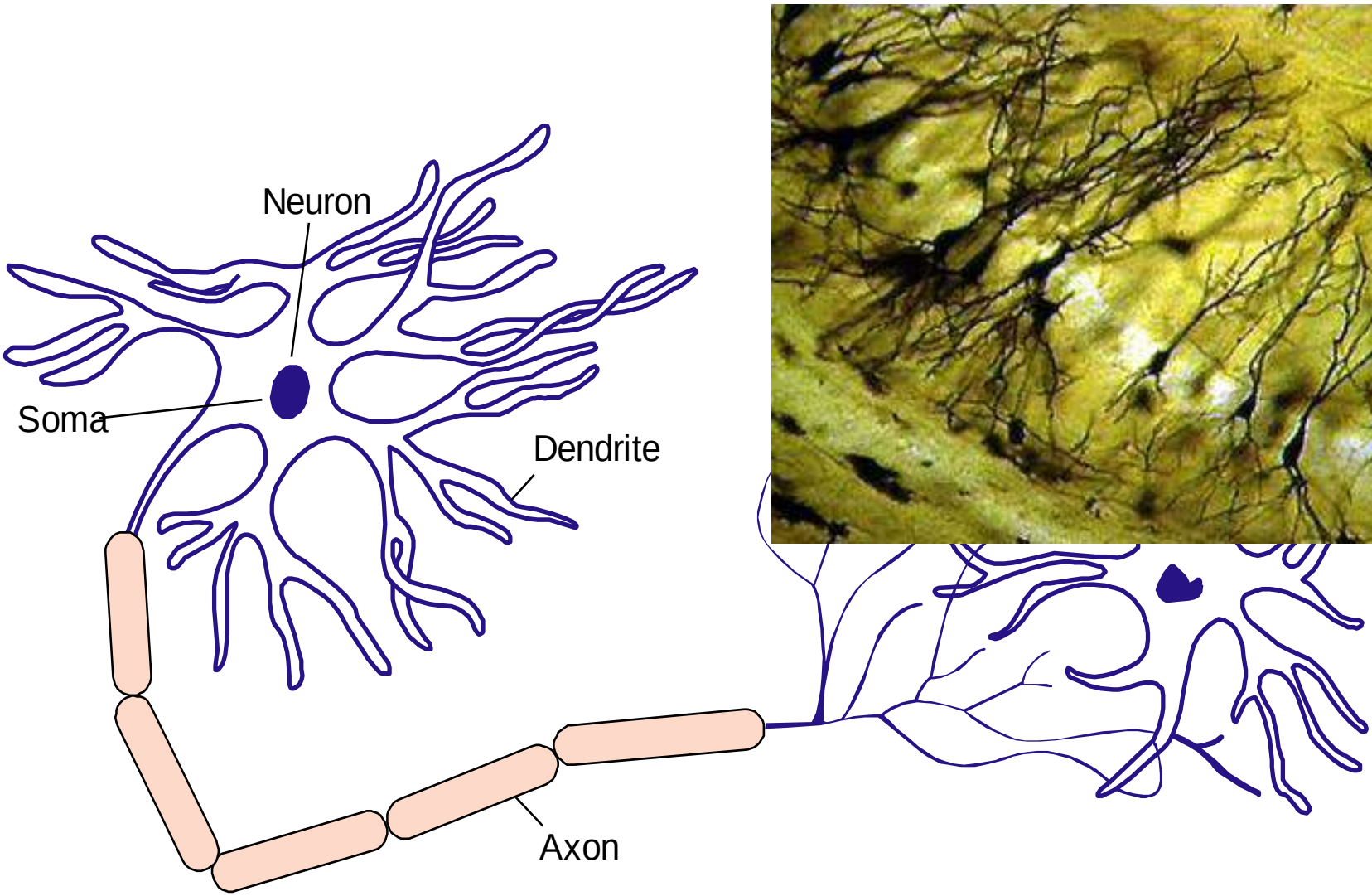
- Wprowadzenie
- Trochę historii
- Podstawy działania
- Funkcja aktywacji
- Uczenie sieci
- Typy sieci
- Zastosowania



Wprowadzenie

- Zainteresowanie SSN wynika z potrzeby budowania efektywnych systemów przetwarzania informacji opartych na regułach neurofizjologii. Przetwarzanie w takich systemach odbywa się w sposób równoległy w odróżnieniu od klasycznych, sekwencyjnych systemów wywodzących się z modelu von Neumanna. Spostrzeżenia:
  - tradycyjne systemy są mało wydajne,
  - mózg ludzki ma przewagę nad maszyną.
- Często podkreślane podobieństwo do systemów biologicznych jest raczej nieuzasadnione, gdyż wciąż zbyt mało wiemy o rzeczywistym funkcjonowaniu mózgu, a sztuczne sieci są z pewnością znacznie uproszczonym modelem systemu neuronów.

Wprowadzenie



Wprowadzenie

- Liczba neuronów: ~ $8,6*10^{10}$
- Liczba połączeń poj. neuronu: ~ 10^4 - 10^5
- Czas reakcji: ~ 0,001 sekundy
- Liczba operacji neuronu: 10-100/s
- Szybkość pracy mózgu ~ 10^{15} - 10^{17} op/s

- W rezultacie:
  czas rozpoznania obrazu: ~ 0,1 sekundy

Wprowadzenie - trochę historii

- w 1943 McCulloch i Pitts opracowali matematyczny model sztucznego neuronu.
- w 1957 Rosenblatt zbudował pierwszy (hardwarowy) model neuronu przeznaczony do rozpoznawania znaków alfanumerycznych.
- w 1969 Minsky i Papert udowodnili, że perceptron ma bardzo ograniczone zastosowanie.
- Lata 80. to model sieci wielowarstwowej i efektywny algorytm uczenia sieci.
- Lata 90. - eksplozja. Ponad 80% firm z listy Fortune 500 ma programy R&D z dziedziny SSN:
  - tysiące publikacji,
  - komercyjne pakiety aplikacyjne.

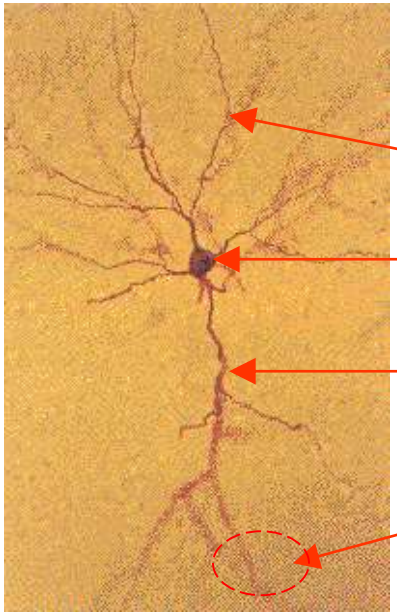
Wprowadzenie

- Sztuczna sieć neuronowa jest przykładem idei równoległego przetwarzania rozproszonego i jako taka składa się z prostych elementów przetwarzających (sztucznych neuronów), które przesyłają między sobą sygnały na zasadzie ważonych połączeń.
- Podstawową cechą różniącą SSN od programów realizujących algorytmiczne przetwarzanie informacji jest zdolność generalizacji czyli uogólniania wiedzy dla nowych danych nieznanych wcześniej, czyli nie prezentowanych w trakcie nauki. Określa się to także jako zdolność SSN do aproksymacji wartości funkcji wielu zmiennych w przeciwieństwie do interpolacji możliwej do otrzymania przy przetwarzaniu algorytmicznym.
- Można to ująć jeszcze inaczej. Np. systemy ekspertowe z reguły wymagają zgromadzenia i bieżącego dostępu do całej wiedzy na temat zagadnień, o których będą rozstrzygały. SSN wymagają natomiast jednorazowego nauczenia, przy czym wykazują one tolerancję na nieciągłości, przypadkowe zaburzenia lub wręcz braki w zbiorze uczącym. Pozwala to na zastosowanie ich tam, gdzie nie da się rozwiązać danego problemu w żaden inny, efektywny sposób.

Podstawy działania

- Cechy charakterystyczne:
  - zbiór elementów przetwarzających,
  - stan aktywacji a_i (wejście) dla każdego elementu,
  - połączenia między elementami; ogólnie rzecz biorąc każde połączenie jest zdefiniowane przez wagi w_{ij}, które określają efekt sygnału (wpływ) elementu j na element i,
  - reguła propagacji, która określa wejście i_i elementu na podstawie sygnałów wejściowych,
  - funkcja aktywacji F_i określająca nowy poziom wyjścia elementu na podstawie wejścia $i_i(t)$,
  - zewnętrzne wejście z_i dla każdego elementu i.
- Poniższe rysunki ilustrują te zasady.

Podstawy działania



Wygląd

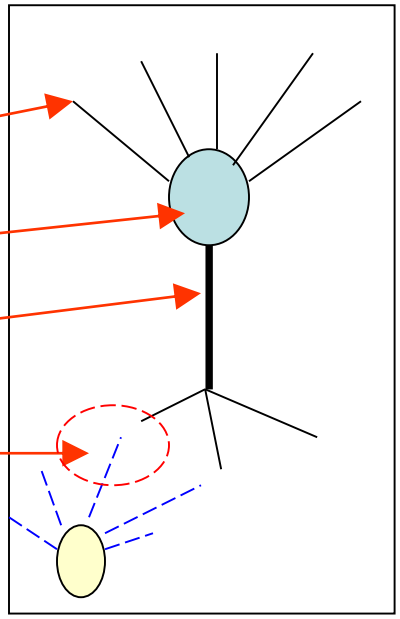
NEURON

Dendryt

Ciało

Akson

Synapsa

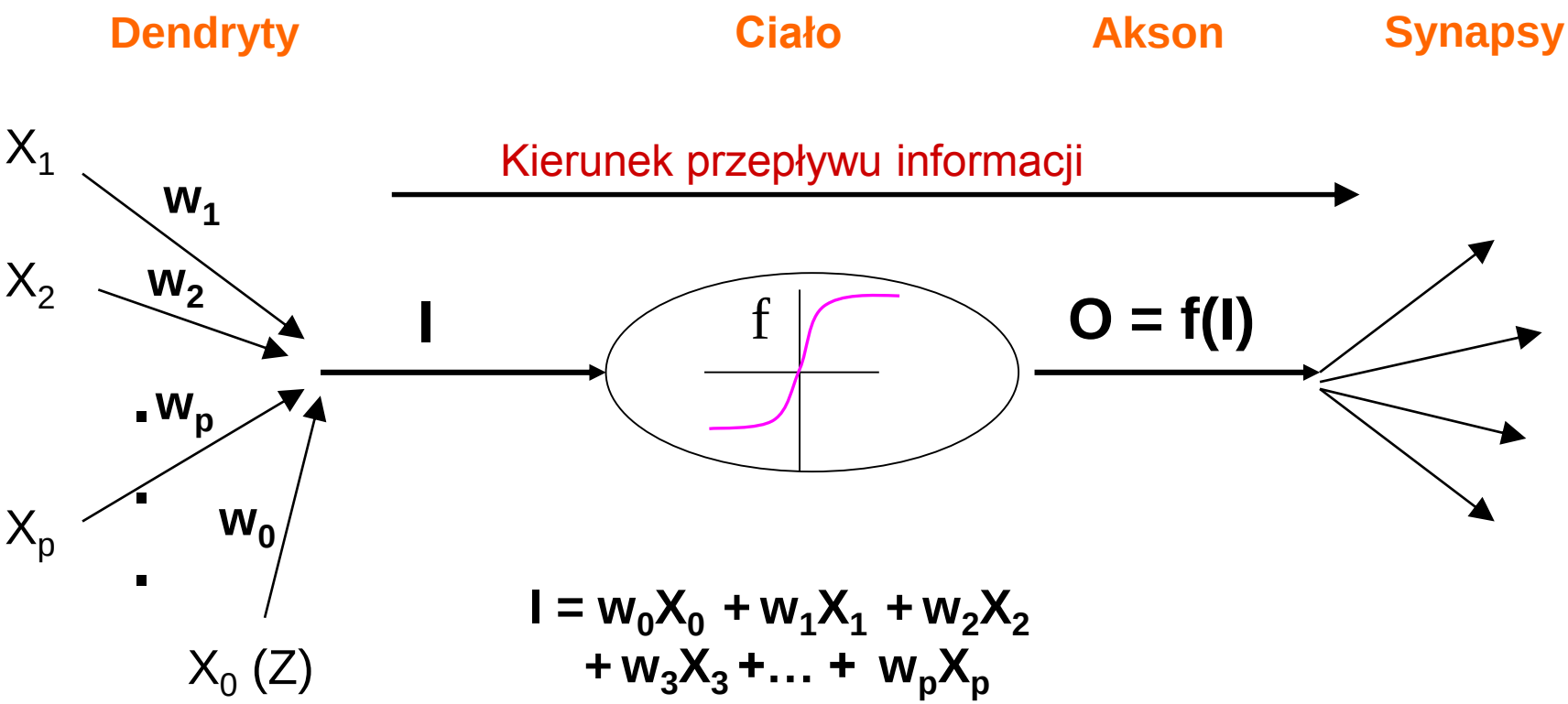


Schemat

- **Dendryty** – otrzymują informacje
- **Ciało** – przetwarza je
- **Akson** – dostarcza przetworzone informacje do innych neuronów
- **Synapsa** – połączenie między synapsami i dendrytami różnych neuronów

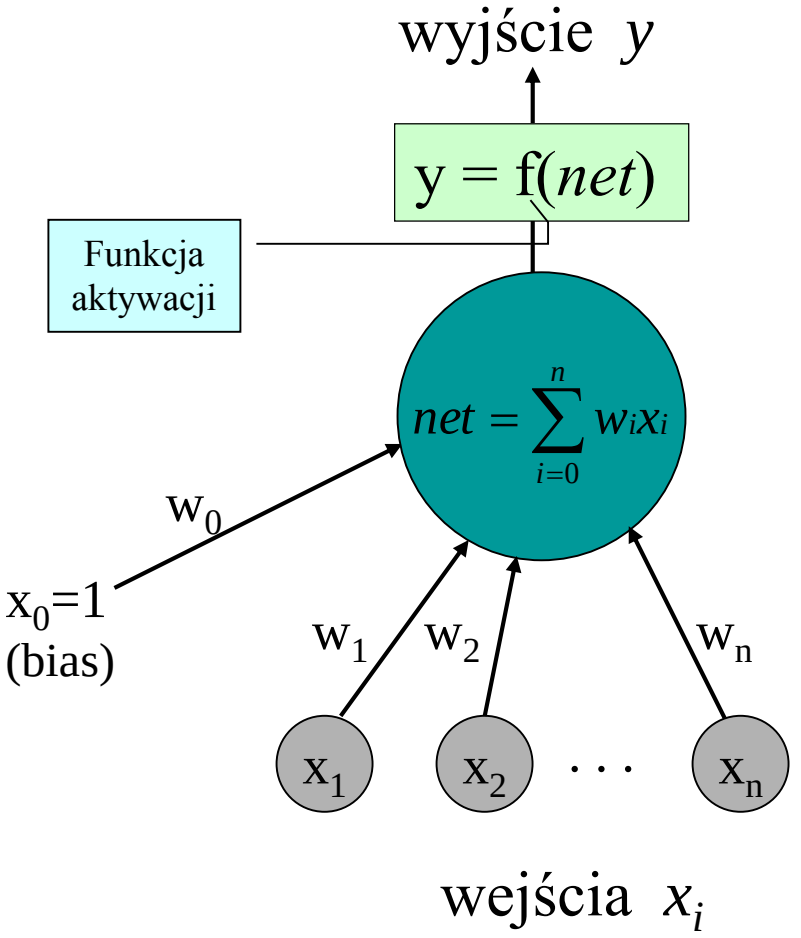
Źródło: Angshuman Saha http://www.geocities.com/adotsaha/NNinExcel.html

Podstawy działania



Podstawy działania

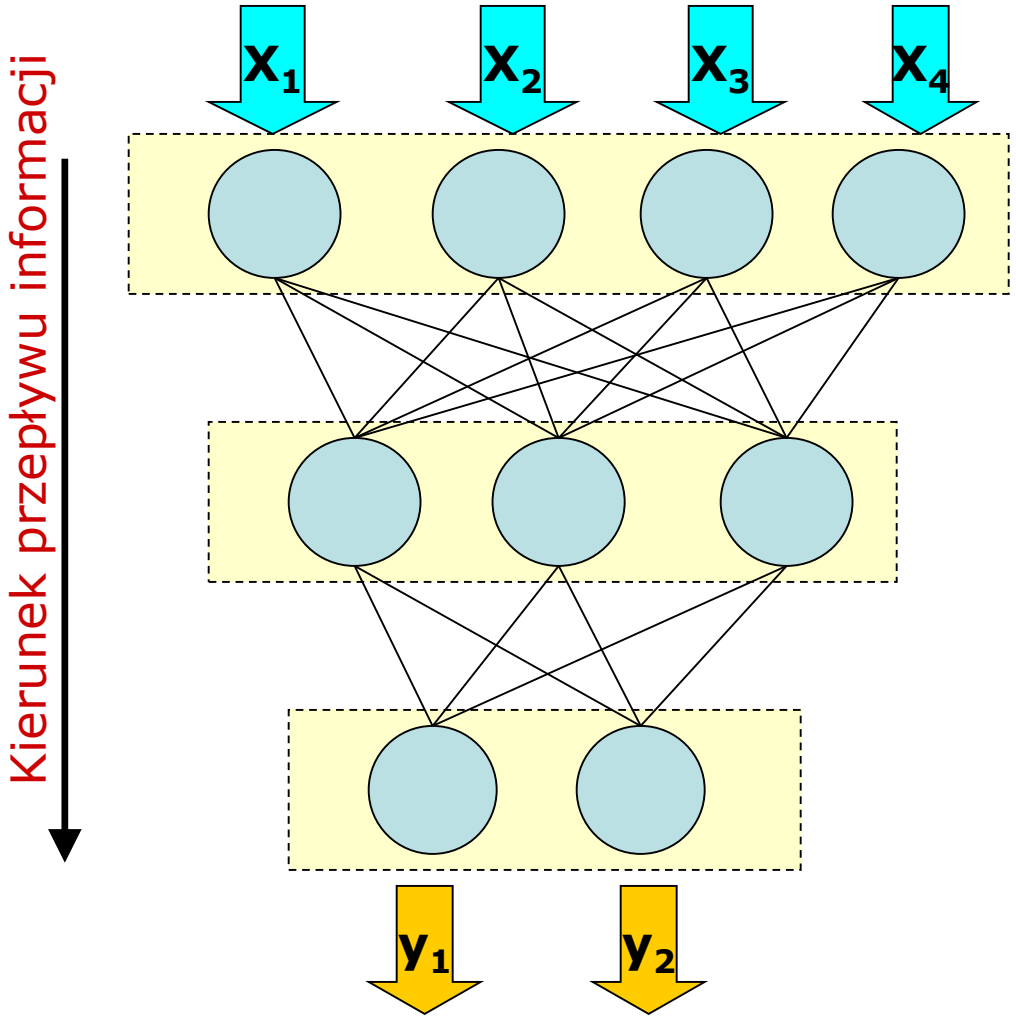
Element przetwarzający - perceptron



Zadaniem elementu jest przetworzyć otrzymany sygnał (od sąsiadów lub z zewnątrz) na sygnał wyjściowy i rozesłać go do innych elementów.

Podstawy działania

Zbiór neuronów tworzy tzw. warstwę



Warstwa wejściowa
każdy neuron otrzymuje tylko jedno wejście, bezpośrednio z zewnątrz

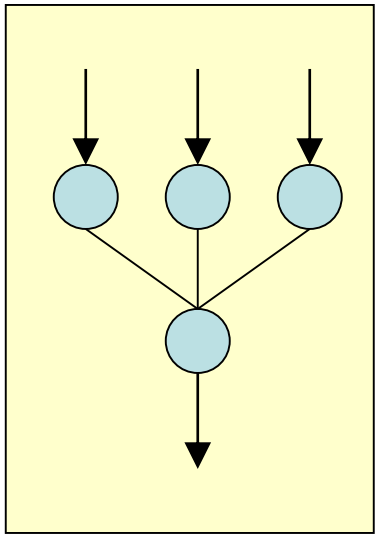
Warstwa ukryta
łączy warstwę wejściową z wyjściową

Warstwa wyjściowa
wyjście każdego neuronu skierowane bezpośrednio na zewnątrz

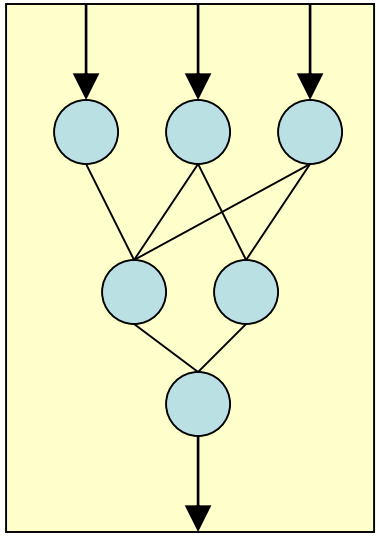
Podstawy działania

Liczba ukrytych warstw może być różna

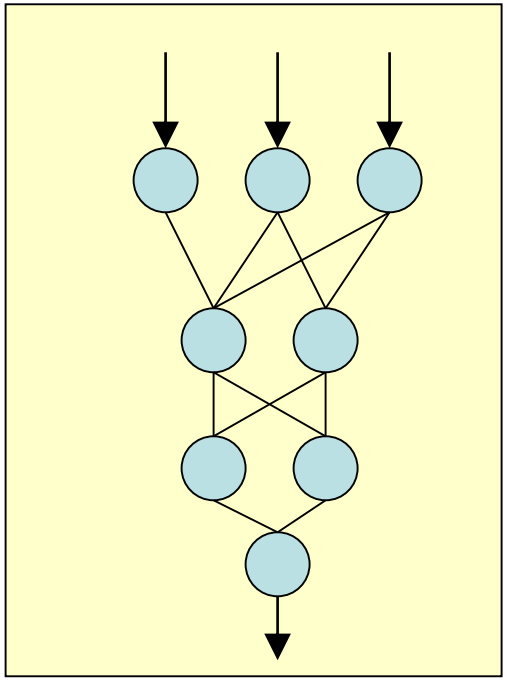
bez



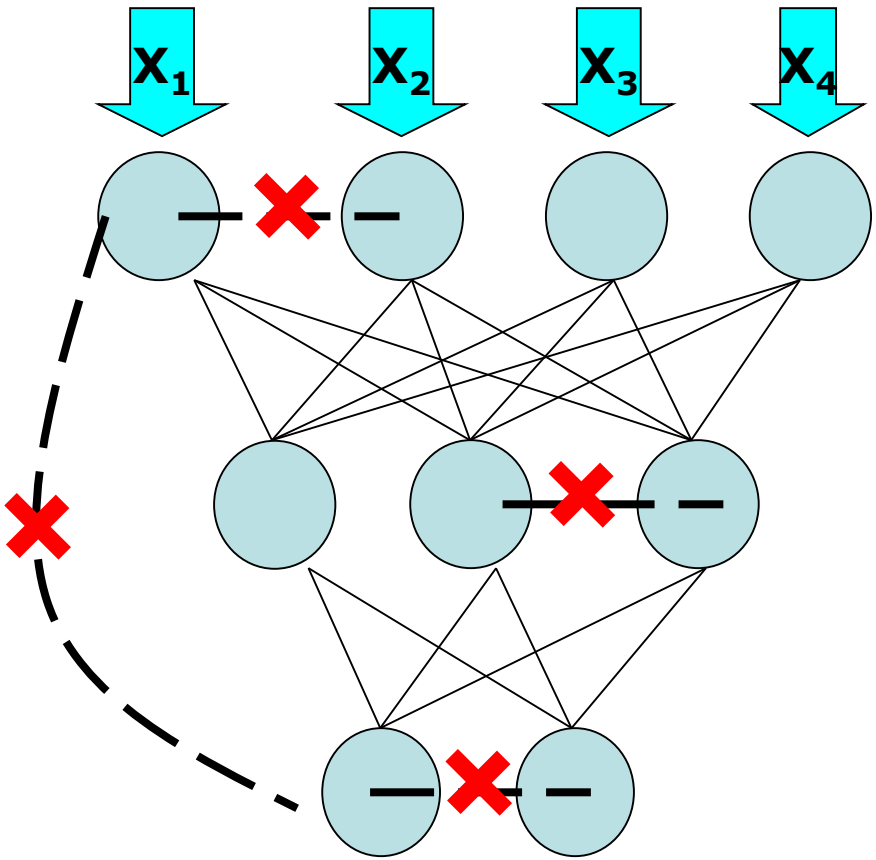
jedna



wiele



Podstawy działania



Uwaga

wewnątrz warstwy neurony nie są ze sobą połączone

neurony z danej warstwy są połączone tylko z neuronami warstwy następnej (feed-forward)

omijanie warstwy nie jest dozwolone

Podstawy działania

Połączenia elementów

Przyjmuje się, że każdy element daje addytywny wkład do elementu, z którym jest połączony:

$$i_i(t) = \sum_j w_{ij}(t) \bullet x_j(t) + z_i(t)$$

Jeśli waga w_{ij} jest dodatnia mówimy o pobudzeniu elementu *i* ze strony elementu *j*, jeśli ujemna - mówimy o powstrzymywaniu.

Podstawy działania

Aktywacja i obliczanie wyjścia

Funkcja *F* daje nową wartość aktywacji elementu *i*, na podstawie wejścia $i_i(t)$

$$o_i(t+1) = F_i(i_i(t))$$

W zależności od charakterystyki funkcji aktywacji na wyjściu elementu *i* otrzymujemy różne wartości.

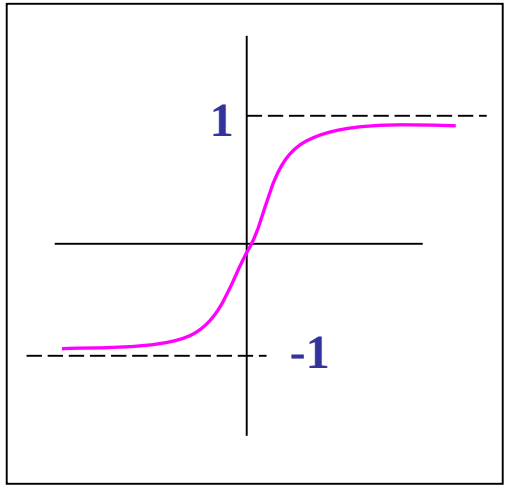
Podstawy działania

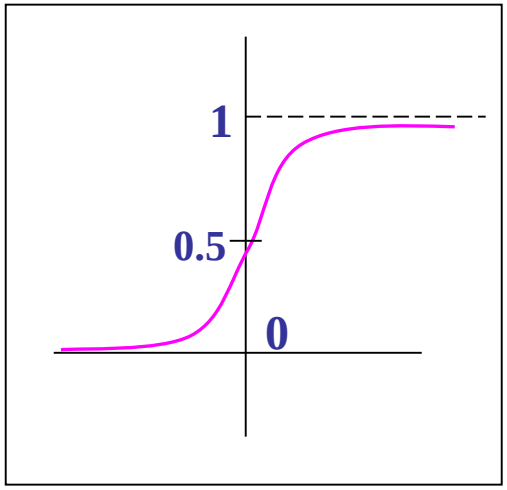
Aktywacja i obliczanie wyjścia

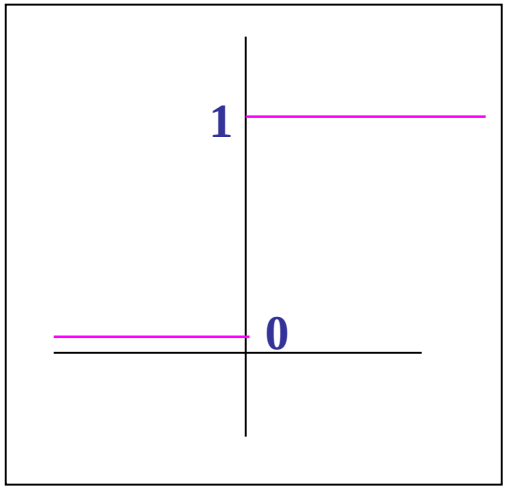
- Wybór funkcji aktywacji zależy od rodzaju problemu, jaki stawiamy do rozwiązania przed siecią. Dla sieci wielo-warstwowych najczęściej stosowane są funkcje nieliniowe, gdyż neurony o takich charakterystykach wykazują największe zdolności do nauki oraz dają możliwość odwzorowania dowolnej zależności pomiędzy wejściem a wyjściem sieci. Umożliwia to otrzymanie na wyjściu sieci informacji ciągłej, a nie tylko postaci: TAK - NIE.
- Wymagane cechy funkcji aktywacji to:
  - funkcja ciągła w zadanym przedziale,
  - łatwa do obliczenia i ciągła pochodna,
  - możliwość wprowadzenia do argumentu parametru do ustalania kształtu krzywej.

Podstawy działania

Aktywacja i obliczanie wyjścia







Tanh

$f(x) = (e^{x} – e^{-x}) / (e^{x} + e^{-x})$

Logistyczna

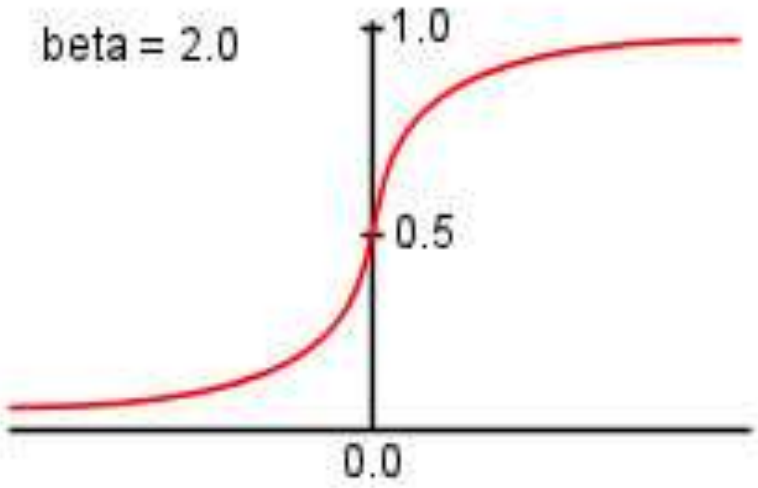
$f(x) = 1 / (1 + e^{-x})$

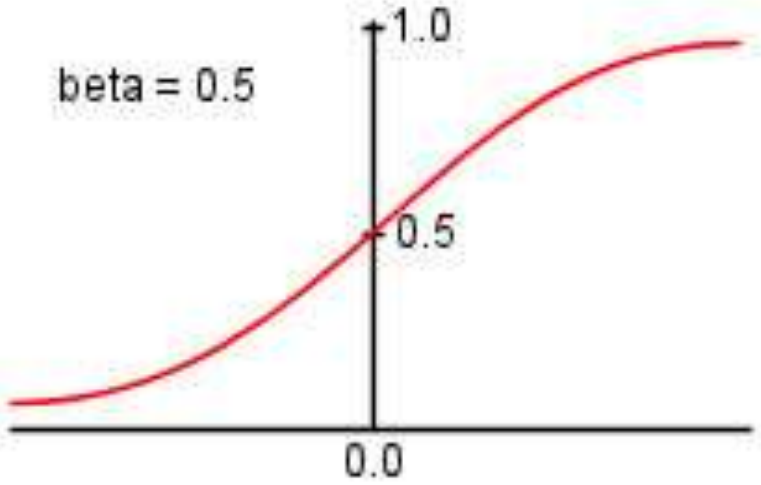
Skok jedn.

$$f(x) = \begin{cases} 0 \text{ if } x <= 0 \\ 1 \text{ if } x > 0 \end{cases}$$

Podstawy działania

Funkcja logistyczna





$$y = f(x) = \frac{1.0}{1.0 + \exp(-(\text{beta} * x))}$$

$$f'(y) = y * (1.0 - y)$$

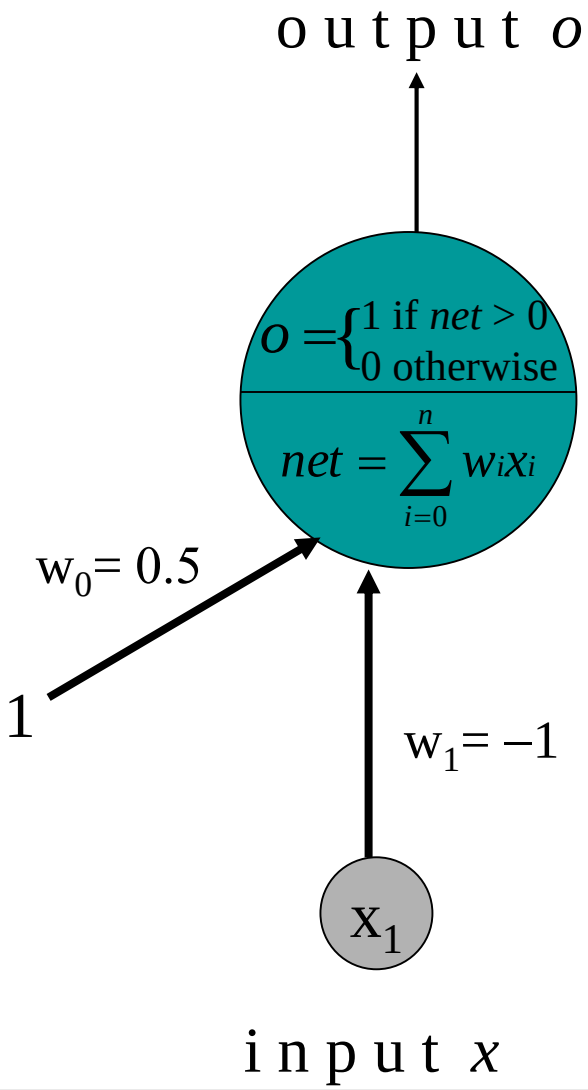
Sieć perceptronowa

- Perceptron zaproponowany przez Rosenblatta w 1957 roku to sieć jednokierunkowa złożona z elementów binarnych, realizowana wg następujących zasad:
  - elementem składowym jest sztuczny neuron opisany funkcją aktywacji *skok jednostkowy*.
  - sieć można podzielić na uporządkowane i rozłączne podzbiory elementów, zwane warstwami: wejściową i wyjściową.
  - perceptron nie zawiera połączeń między elementami tej samej warstwy.
  - połączenia między warstwami skierowane są zgodnie z ich uporządkowaniem.

Sieć perceptronowa

Perceptron
odwracanie

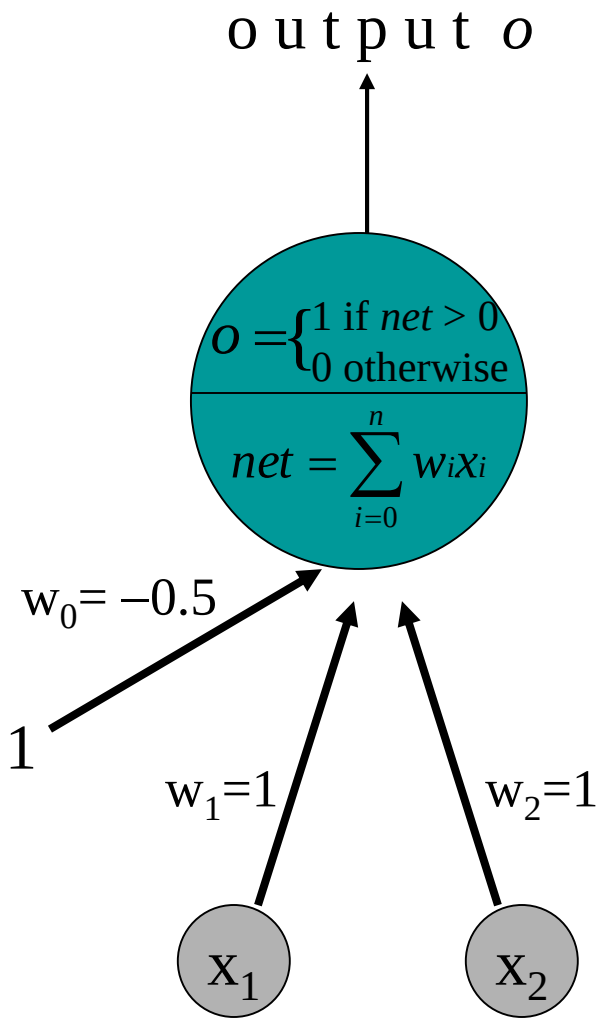
input x1	output
0	1
1	0



Sieć perceptronowa

Perceptron
OR

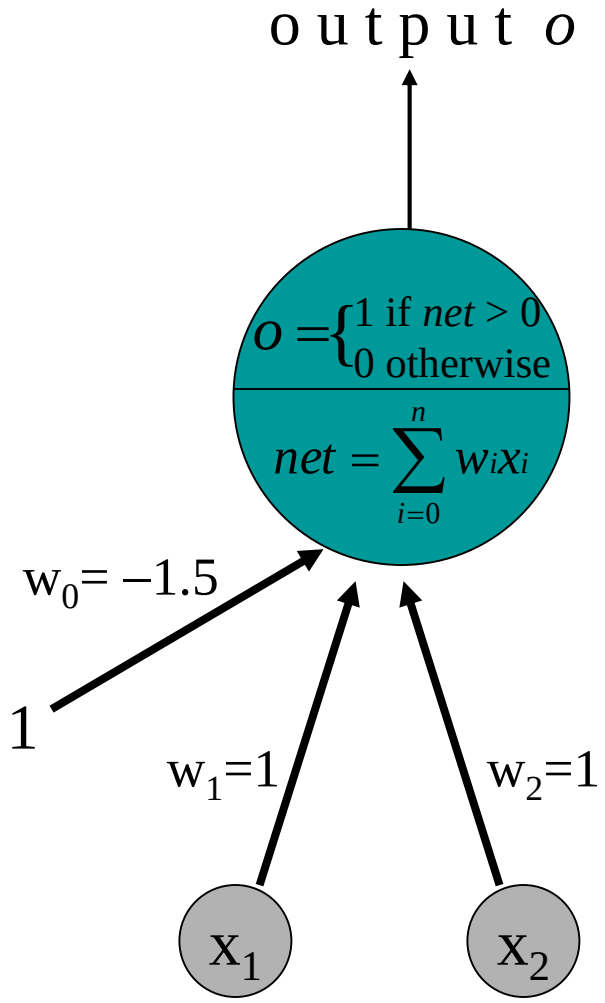
input x1	input x2	output
0	0	0
0	1	1
1	0	1
1	1	1



Sieć perceptronowa

Perceptron
AND

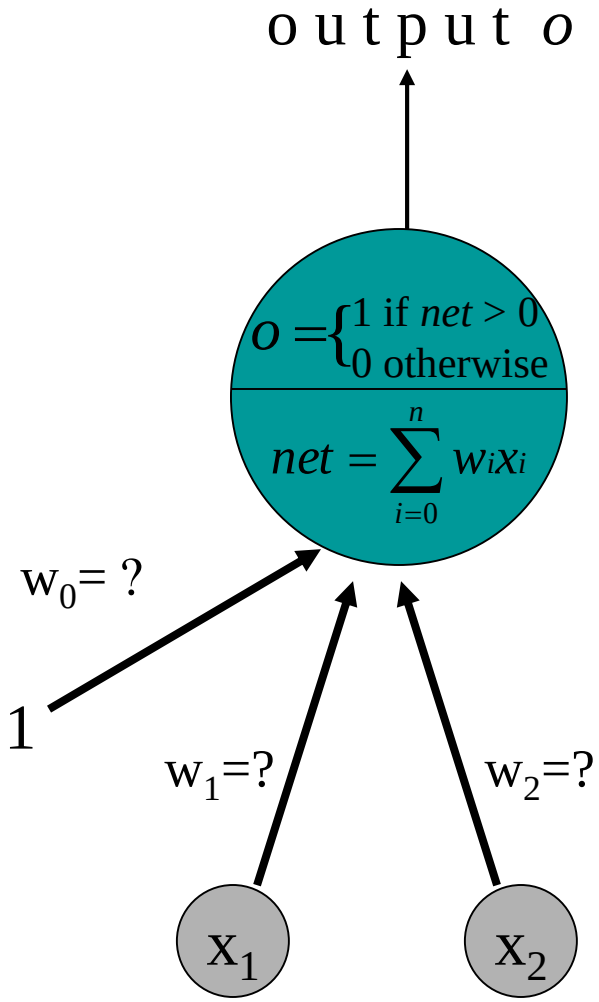
input x1	input x2	output
0	0	0
0	1	0
1	0	0
1	1	1



Sieć perceptronowa

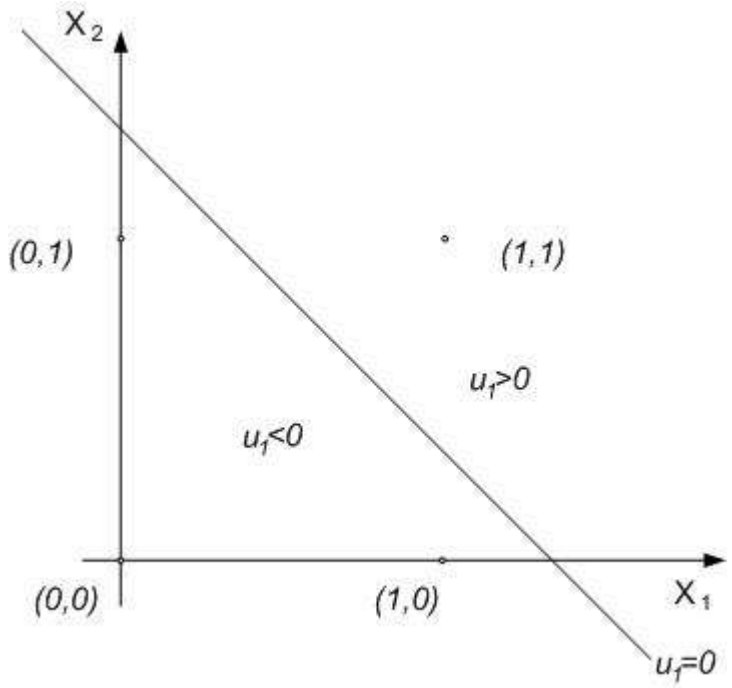
Perceptron
XOR

input x1	input x2	output
0	0	0
0	1	1
1	0	1
1	1	0

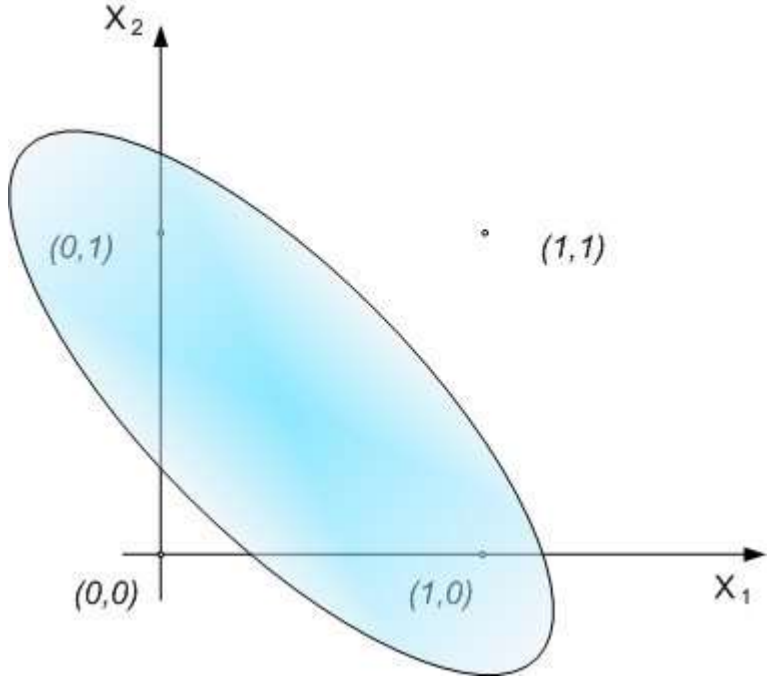


Sieć perceptronowa

Perceptron AND



Perceptron XOR



Źródło: XOR problem theory, http://home.agh.edu.pl/~vlsi/AI/xor_t/en/main.htm, dostęp: 8.11.2020

Metody uczenia

- Uczenie polega na takim wyznaczeniu wartości wag, by sieć rozwiązywała wyuczone zadanie w sposób automatyczny.
  Wyróżniamy dwie kategorie:
  - ***uczenie nadzorowane*** (supervised), gdzie dostarcza się wzorce wejściowe i odpowiadające im wyjścia.
  - ***uczenie nienadzorowane***, gdzie sieć samodzielnie próbuje odkryć statystycznie istotne cechy wzorca;
    w tej metodzie nie ma danego a priori zbioru kategorii, według których mają być klasyfikowane wzorce.

Metody uczenia

- Wszystkie metody uczenia są wariantami reguły uczenia Hebba opisanej w *Organization of Behaviour* (1949). Podstawową ideą jest tu założenie, że jeśli dwa elementy działają zgodnie, ich połączenie musi być wzmocnione. Jeżeli element *i* otrzymuje wejście od *j*, to najprostsza wersja reguły Hebba opisuje zmianę wag w_{ij} o:

$$\Delta w_{ij} = \alpha \bullet o_i \bullet o_j$$

gdzie α jest dodatnią stałą nazywaną *stałą uczenia*.

Metody uczenia

- Inna powszechnie stosowana reguła używa nie aktualnej aktywacji elementu *i*, lecz różnicy między aktualną a pożądaną aktywacją elementu.

wagi zmieniane są tym silniej, im większy jest błąd

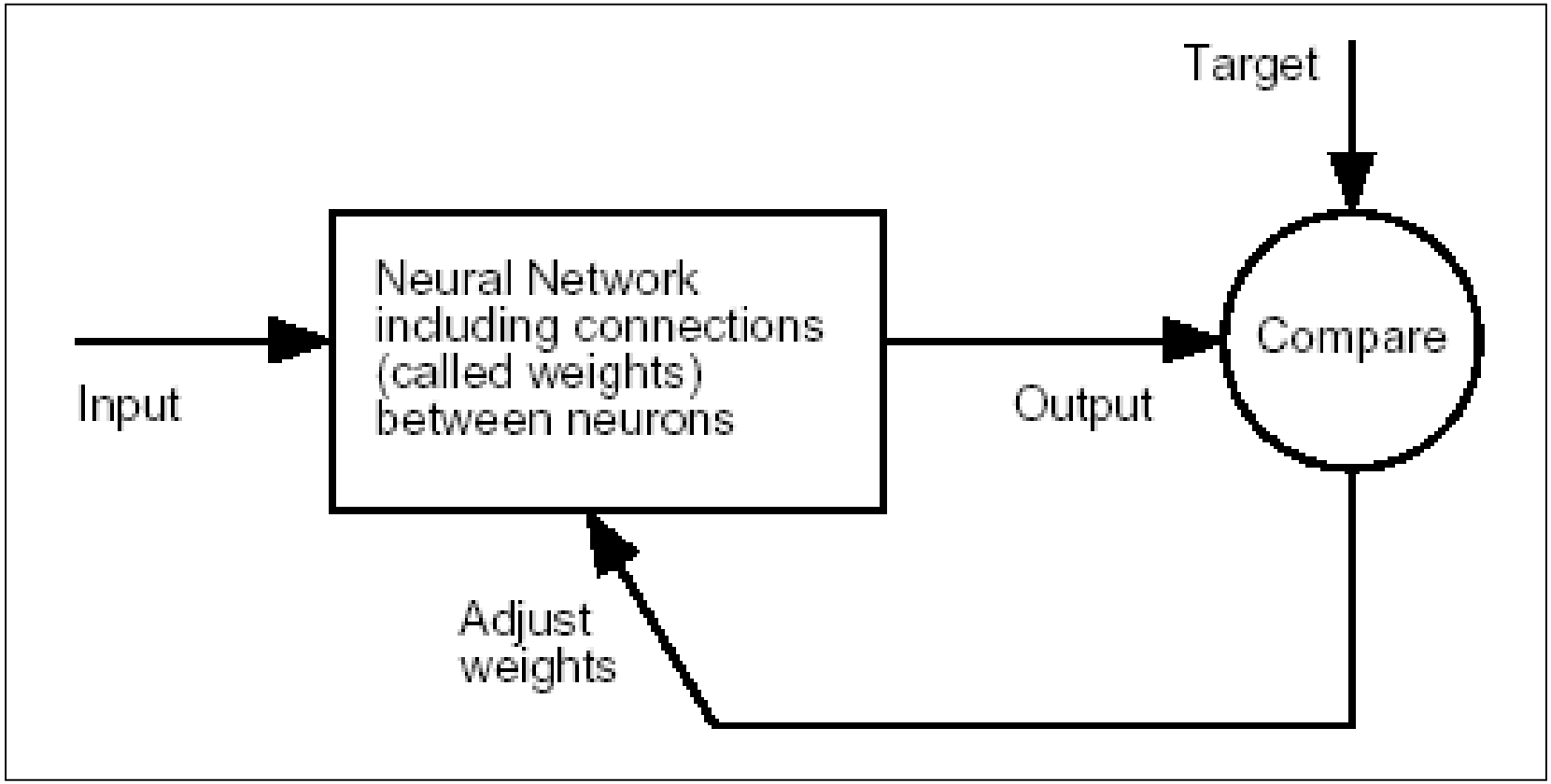
$$\Delta w_{ij} = \alpha \bullet (d_i - o_i) \bullet o_j$$

gdzie d_i jest pożądaną aktywacją, której wartość jest podawana przez nauczyciela.

Reguła ta jest nazywana regułą Widrowa-Hoffa (1960) lub regułą delty.

wagi zmieniane są proporcjonalnie do wielkości sygnałów wejściowych

Metody uczenia



Graficzna ilustracja procesu uczenia nadzorowanego

Metody uczenia

- Uczenie sieci, to proces minimalizacji błędu (odchylenia):

$$E[\vec{w}] = \frac{1}{2} \sum_{d \; \varepsilon \; D} (t_d - o_d)^2$$

D = dane treningowe

Uczenie jest zbieżne, jeśli:

… nie ma nieliniowych zależności między danymi,

… stała uczenia α ma małą wartość,

… nie ma warstw ukrytych.

Budowa modelu

Co to znaczy zbudować model ?

Input: X_1 X_2 X_3 Output: Y Model: $Y = f(X_1 X_2 X_3)$

W przypadku SSN : matematyczny zapis f(...) jest zbyt skomplikowany.

Jednakże wystarczająco charakteryzują go:

- liczba neuronów wejściowych,
- liczba warstw ukrytych,
- liczba neuronów w warstwach ukrytych,
- liczba neuronów wyjściowych,
- **wagi** wszystkich połączeń.

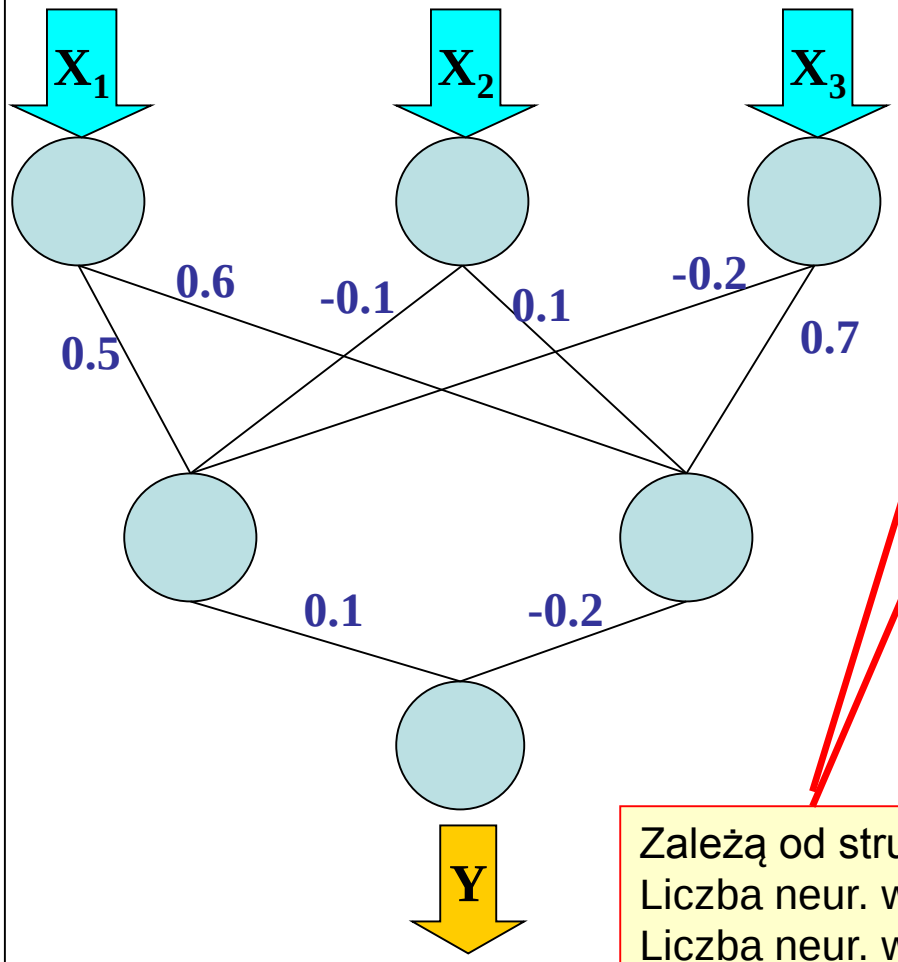
Budowa modelu SSN = określenie wartości powyższych parametrów

Ten i sześć następnych slajdów - autor: Angshuman Saha

http://www.geocities.com/adotsaha/NNinExcel.html

Budowa modelu

Input: X_1 X_2 X_3 Output: Y Model: $Y = f(X_1\ X_2\ X_3)$



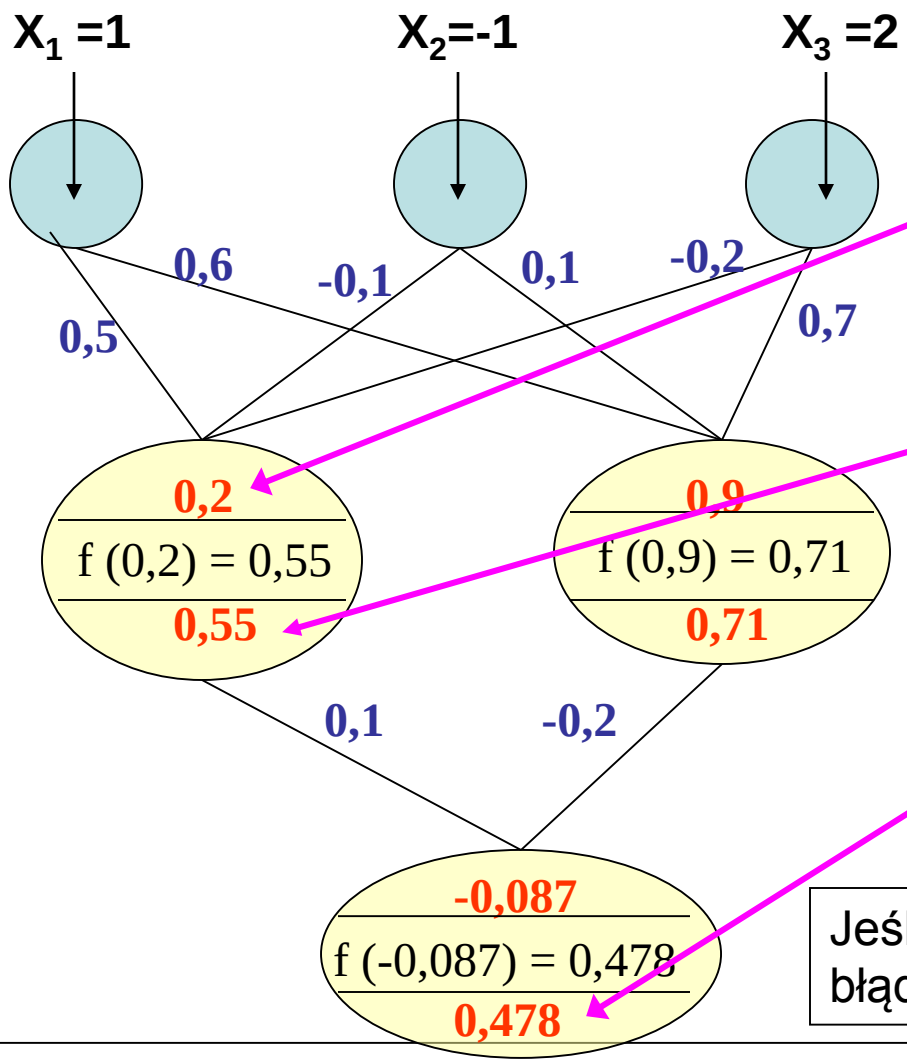
Parametry	Przykład
L. neuronów wejściowych	3
L. warstw ukrytych	1
Neurony w warstwie ukr.	2
L. neuronów wyjściowych	1
Wagi	rysunek

Zależą od struktury problemu
Liczba neur. wej.= liczba X
Liczba neur. wyj. = liczba Y

Wolne parametry

Budowa modelu

Input: X_1 X_2 X_3 Output: Y Model: $Y = f(X_1\ X_2\ X_3)$



0,2 = 0,5 * 1 - 0,1*(-1) - 0,2 * 2

$f(x) = 1 / (1 + e^{-x})$

$f(0,2) = 1 / (1 + e^{-0,2}) = 0,55$

Wyliczone Y = 0,478

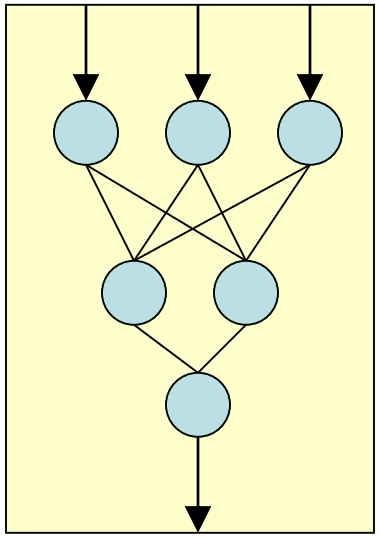
Jeśli wzorcowe Y = 2, to
błąd oszacowania = (2-0,478) =1,522

Budowa modelu

Input: $X_1\ X_2\ X_3$ **Output:** Y **Model:** $Y = f(X_1\ X_2\ X_3)$

L. neuronów wej. = liczba X = **3** L. neuronów wyj. = liczba Y = **1**

L.warstw ukrytych = ???	próbuj **1**	Nie ma reguły.
L.neuronów w warstwie = ???	próbuj **2**	Metoda prób i błędów.



Mamy architekturę … A co z wagami ???

Przy tej architekturze mamy do określenia 8 wartości wag:

$$\underline{\mathbf{W}} = (W_1, W_2, \ldots, W_8)$$

Dane treningowe: $(Y_i, X_{1i}, X_{2i}, X_{3i})$ $i = 1,2,\ldots,n$

Dla określonych wag **W**, otrzymamy oszacowane *n* wartości Y

$$(V_1, V_2, \ldots, V_n)$$

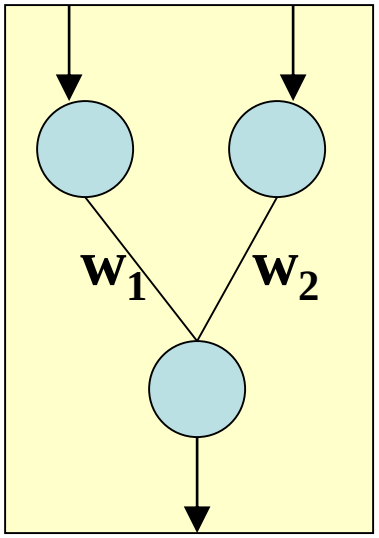
Oczywiście są one *funkcją* **W**, czyli ogólna reguła brzmi:

Dobierz tak **W**, by błąd oszacowania ***E*** był zminimalizowany

$$\boldsymbol{E} = \sum_i (Y_i - V_i)^2$$

Uczenie sieci

Przykład prostej sieci z 2 neuronami wejściowymi i 1 - wyjściowym.



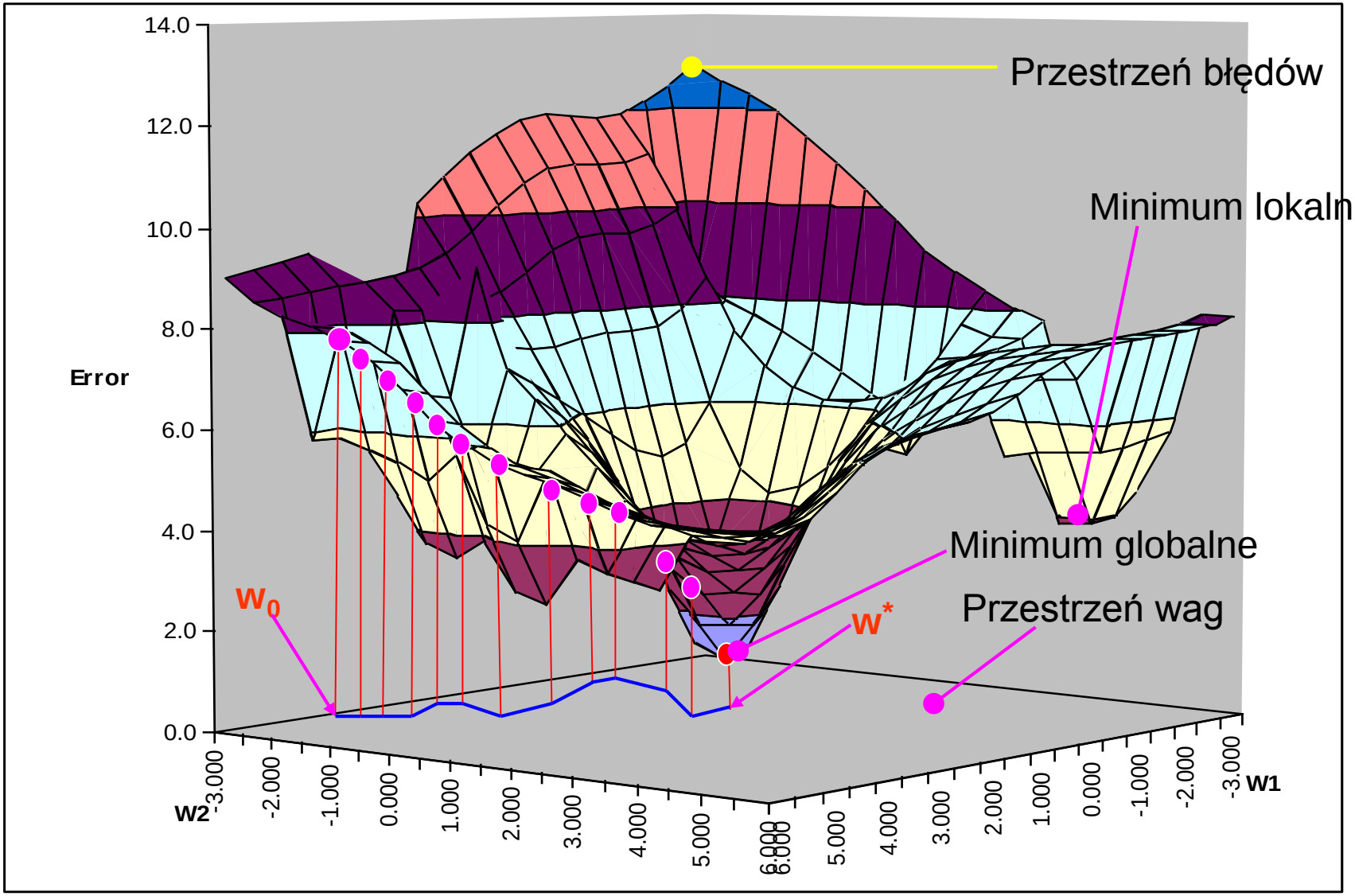
$$E(w_1, w_2) = \Sigma [Y_i - V_i(w_1, w_2)]^2$$

- Para (w_1, w_2) jest punktem w 2-wymiarowej płaszczyźnie.
- Dla każdej takiej pary mamy wartość ***E***.
- Funkcja E=f(w_1, w_2) wyznacza - **'przestrzeń błędów'**
- Cel: znaleźć taką parę wag, dla której ***E*** jest minimalny.

Algorytm hill climbing

1. Zacznij od losowej pary (w_1, w_2)
2. Przesuń się do pary (w'_1, w'_2), dla której błąd jest mniejszy.
3. Kontynuuj krok 2 aż dojdziesz do pary (w^*_1, w^*_2), gdzie ***E*** jest minimalny.

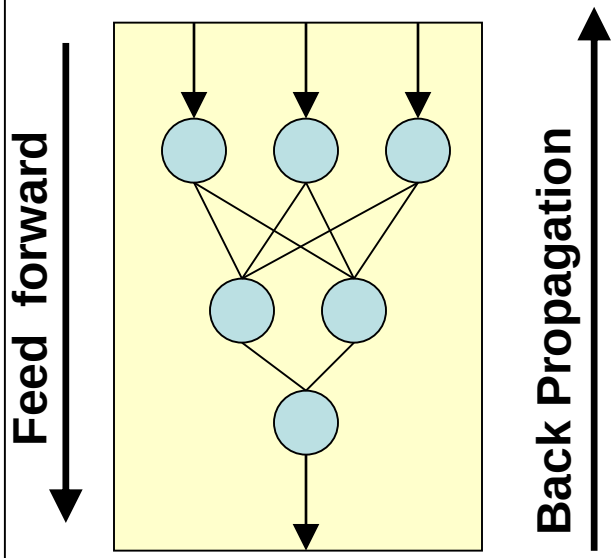
Uczenie sieci



Uczenie sieci

Algorytm wstecznej propagacji błędu Backpropagation Algorithm

uogólnienie reguły delty dla złożonych sieci



$$E = \Sigma (Y_i - V_i)^2$$

1. Zacznij z wylosowanymi wartościami wag.
2. Zaprezentuj sieci pierwszą obserwację
   $\mathbf{X_1}$ -> sieć -> $\mathbf{V_1}$; Błąd $= (\mathbf{Y_1} - \mathbf{V_1})^2$
3. Zmodyfikuj wagi tak, by zminimalizować błąd (sieć dopasowuje się do 1. obserwacji)
4. Zaprezentuj sieci kolejną obserwację. Zmodyfikuj wagi tak, by zminimalizować błąd.
5. Powtarzaj krok 3. aż do ostatniej obserwacji.
6. To kończy jeden cykl treningowy.
7. Wykonaj wiele takich cykli (epok), aż ***E*** osiągnie małą (minimalną?) wartość.

Uczenie sieci

Parametry uczenia:

- stała uczenia
- bezwładność (momentum)

Im wyższy współczynnik uczenia tym sieć uczy się szybciej, ale może wpaść w oscylacje

$$\Delta w_{ji}(t) = \alpha \delta_j(t) o_i(t) + \gamma \Delta w_{ji}(t-1)$$

$$\delta_j = -\frac{\partial E}{\partial o_j}\frac{\partial o_j}{\partial net_j} = -\frac{\partial E}{\partial o_j} f'(net_j)$$

Momentum - pozwala usunąć szybkie oscylacje

Sposób prezentacji danych:

- losowa prezentacja – element stochastyczny, uczenie on-line.
- ustalona kolejność.

Sposób modyfikacji wag (wsadowy vs. przyrostowy).

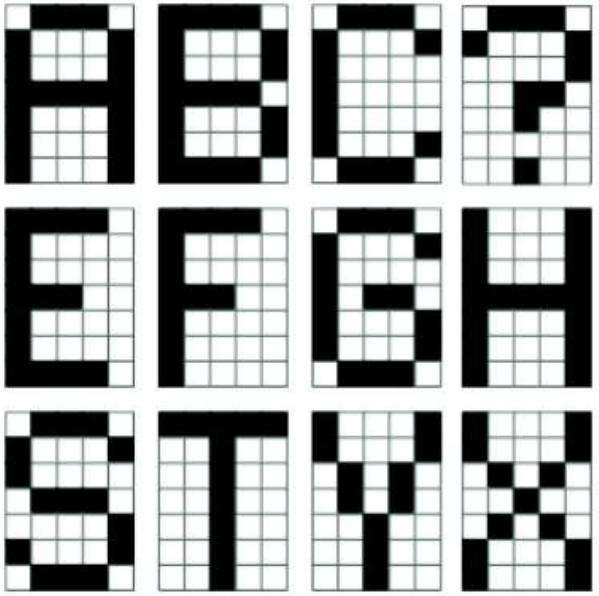
Metody uczenia

- **Sposób modyfikacji wag**
- **Uczenie przyrostowe**, w tej odmianie minimalizuje się funkcję błędów dla każdej pary wektorów zmiennych wejściowych i wzorców osobno; modyfikacja wag następuje każdorazowo po takiej operacji.
- **Uczenie wsadowe**, w tej odmianie minimalizuje się błąd średniokwadratowy wyznaczony dla wszystkich próbek ze zbioru uczącego, również korekta wag następuje po zaprezentowaniu całego zbioru uczącego.

Uczenie sieci – przykład*

- Uczenie metodą propagacji wstecznej jednowarstwowej sieci klasyfikującej dwanaście znaków, reprezentujących odrębne klasy (zajmują wierzchołki 35-wymiarowej kostki i są liniowo separowalne, a tym samym nauczona sieć jednowarstwowa powinna je poprawnie klasyfikować).

- Wyniki klasyfikacji odczytywano przyporządkowując sygnałom na wyjściach neuronów liczbę 1, gdy przekraczają wartość 0,5 , a liczbę 0 w przeciwnym razie. Badana sieć zawierała 12 neuronów o logistycznych funkcjach aktywacji z parametrami $\beta = 1$.

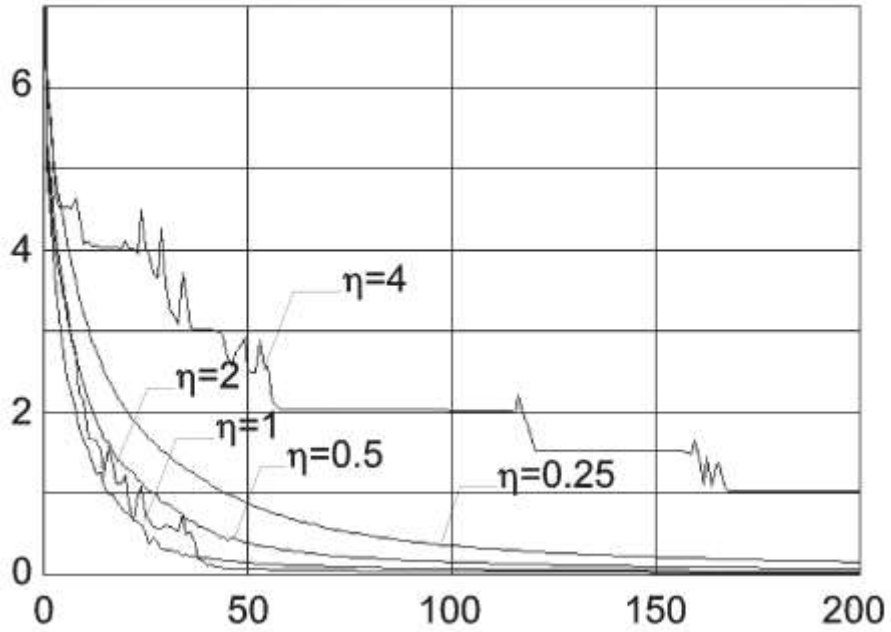


*Za: W. Jędruch, Sztuczna inteligencja, Politechnika Gdańska, 2010.

Uczenie sieci – przykład

- Przeprowadzono 200 cykli uczenia sieci dla kilku wartości współczynnika η, po których uzyskano następujące błędy łączne: $\eta = 4$, $E = 1{,}0040$ $\eta = 2$, $E = 0{,}0138$ $\eta = 1$, $E = 0{,}0294$ $\eta = 0{,}5$, $E = 0{,}0651$ $\eta = 0{,}25$, $E = 0{,}1469$

Z rysunku widać wpływ współczynnika η na szybkość uczenia. Zbyt duża wartość (η=4) powoduje duże oscylacje i przeregulowania i w efekcie duży błąd końcowy. Pośrednie wartości (η=2 lub 1) powodują, pomimo niewielkich oscylacji, szybkie osiąganie małych wartości błędu. Wreszcie zbyt małe wartości (η=0,5 lub 0,25) powodują systematyczne i bez oscylacji, ale powolne malenie błędu.



Uczenie sieci – przykład

- Dla ilustracji reakcji sieci poniżej przedstawiono sygnały wyjściowe neuronów nauczonej sieci (dla $\eta = 2$) odpowiadającej na przykładowy obraz wejściowy (znak A), gdzie w nawiasach przy liczbach podano znaki reprezentowane przez poszczególne neurony:

$$\begin{array}{cccccccc} (A) & 0,9799, & (B) & 0,0039, & (C) & 0,0113, & (?) & 0,0103 \\ (E) & 0,0001, & (F) & 0,0100, & (G) & 0,0020, & (H) & 0,0123 \\ (S) & 0,0002, & (T) & 0,0014, & (Y) & 0,0001, & (F) & 0,0086 \end{array}$$

Zalety SSN

1. Rozwiązuje problemy, które klasyczne algorytmy nie są w stanie rozwiązać, mówimy tutaj przede wszystkim o zadaniach, w których skład wchodzi kojarzenie czy przewidywanie.
2. Nie potrzebują specjalnego algorytmu, całym zadaniem jest zaprojektowanie odpowiedniej struktury sieci odpowiadającej problemowi oraz pokierowanie procesem uczenia się sieci. Niepotrzebne są charakterystyki modelu, założenia, ograniczenia. Wystarczy podać cel i dużo przykładów praktycznego osiągnięcia tego celu. Wprawdzie sieć nie poda reguł kierujących określonym zjawiskiem, ale poda rozwiązanie problemu.

Zalety SSN

3. Działają jako całość, każdy element ma pewien wkład w realizację zadania, dlatego też mogą działać pomimo błędów (np. struktury czy też błędów w danych). Klasyczny algorytm w sytuacji błędu np. w strukturze algorytmu lub w zestawie danych najczęściej „wywraca się”, uniemożliwiając działanie programu. Natomiast SSN dopiero w przypadku ilości błędów powyżej pewnego granicznego poziomu nie jest w stanie poprawnie działać.
4. Umiejętność uogólniania wiedzy. *Znaczy to dokładnie tyle, że jeśli sieć nauczy się, powiedzmy rozpoznawać kolory: czerwony i żółty, to rozpozna również różowy i bladożółty, czyli kolory podobne do znanych*.

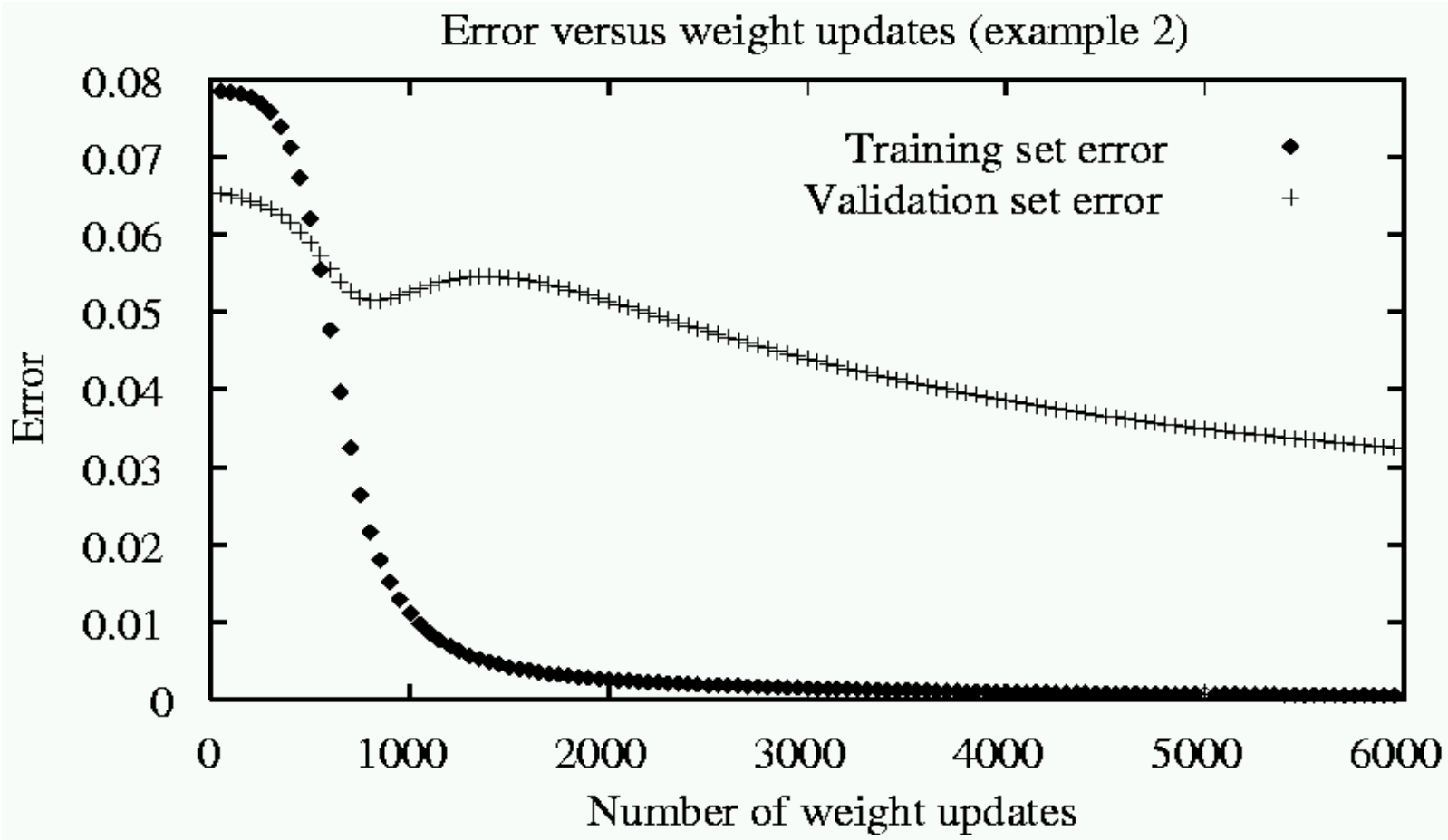
Korbicz J., Obuchowicz A., Uciński D. - "Sztuczne sieci neuronowe. Podstawy i zastosowania", Akademicka Oficyna Wydawnicza PLJ, Warszawa 1994

Wady SSN

1. Operują pojęciami rozmytymi, nie określają dokładnie, lecz w przybliżeniu, podobnie zresztą jak ludzki mózg. Dlatego nie mogą być stosowane tam, gdzie potrzebna jest precyzyjna wartość.
2. Nie radzą sobie z rozumowaniem wieloetapowym, gdy z jednych wniosków trzeba wysnuć inne, a z tych - kolejne itd. Wynika to z faktu, że sieć neuronowa działa jednoetapowo, tylko w jednym kroku.
3. Proces uczenia się sieci może trwać bardzo długo; wszystko zależy od stopnia skomplikowania sieci oraz złożoności problemu do rozwiązania.
4. Trudno wysuwać wnioski co do istoty zależności między danymi wyjściowymi a wejściowymi: sztuczna sieć neuronowa działa jak „czarna skrzynka”, do której wrzucamy dane i otrzymujemy wyniki, natomiast logiki systemu i schematów myślenia nie jesteśmy w stanie odkryć.
5. Łatwo „przeuczyć sieć”, tzn. sieć nauczy się na pamięć podawanych przykładów w fazie uczenia i w momencie testowania na innych przykładach jest bezużyteczna, nie potrafi właściwie rozwiązać zadania (nie ma zdolności do generalizacji).

Wady SSN

- **Uczenie sieci - przetrenowanie**



Cykl życia SSN

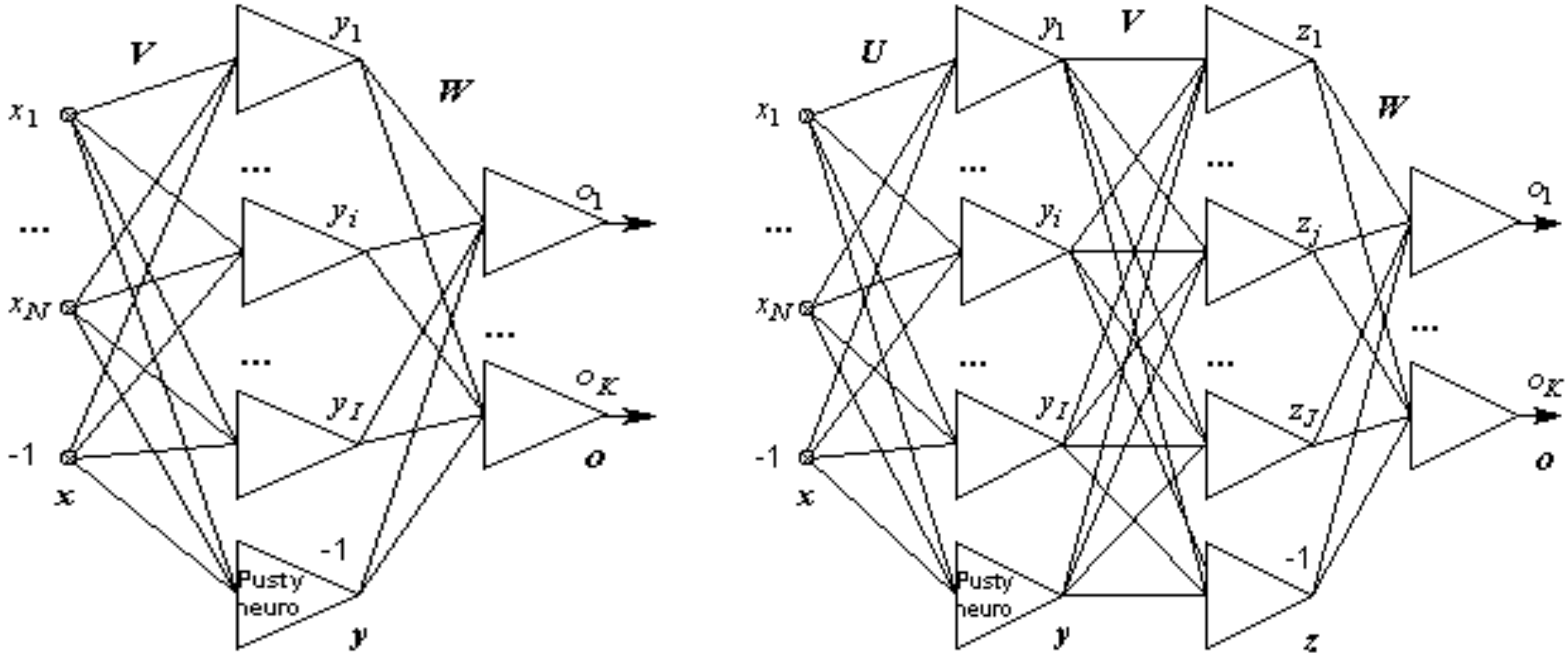
1. Określenie zmiennej wyjściowej oraz zmiennych wejściowych.
2. Gromadzenie danych.
3. Wstępne przetwarzanie danych (preprocessing).
4. Podział zbioru danych.
5. Dobór odpowiedniej architektury sieci.
6. Uczenie sieci.
7. Testowanie i weryfikacja.
8. Zastosowanie sieci.

Topologia SSN

- Ze względu na rodzaj powiązań między elementami sieci wyróżniamy:
  - *sieci jednokierunkowe* (feed-forward), w których dane przepływają w jednym kierunku od elementów wejściowych do wyjściowych.
  - *sieci rekurencyjne* zawierające połączenia ze sprzężeniami zwrotnymi.

Topologia SSN

Sieci jednokierunkowe



Wartości wyjść zależą od aktualnych wartości wejść do neuronu, jest to więc statyczna zależność funkcyjna.

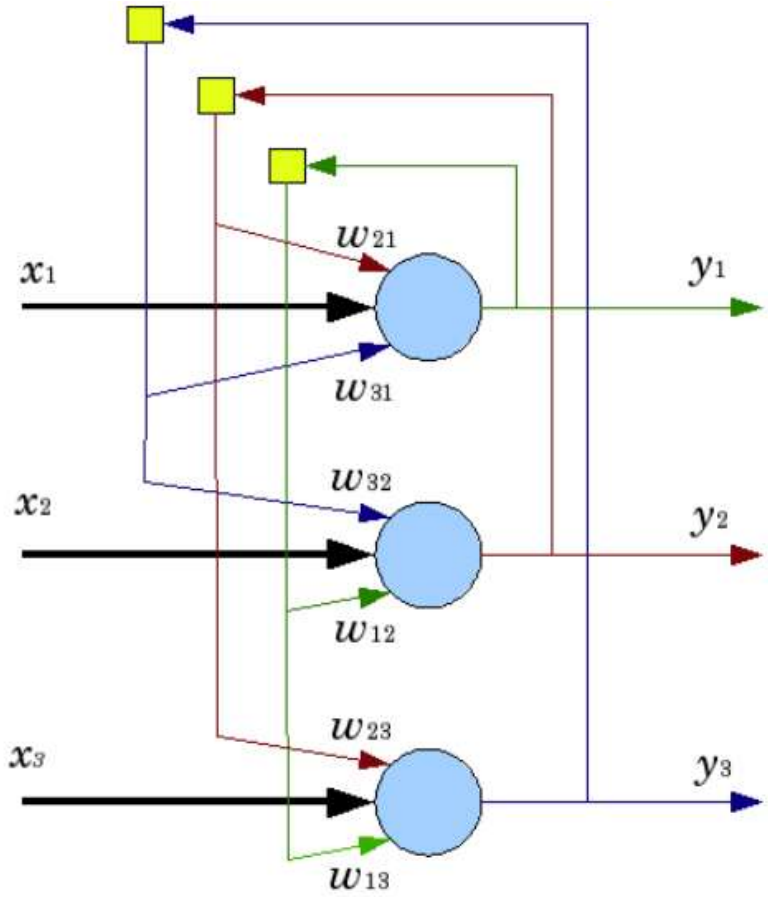
Topologia SSN

Sieci rekurencyjne

- W swoje strukturze posiadają sprzężenia zwrotne, czyli wyjście warstwy późniejszej jest bezpośrednio lub pośrednio połączone z wejściem warstwy wcześniejszej lub połączone są wyjścia neuronów z ich wejściami. Powoduje to, że sieć staje się dynamiczna. Wartości na jej wyjściach zależą nie tylko od wartości wejściowych, ale również od wcześniejszych wyjść.
- Przykładem sieci rekurencyjnych są sieci Jordana-Elmana oraz Hopfielda. W strukturach takich sieci sygnały wyjściowe z warstwy wyjściowej lub ukrytej opóźniane są o jeden cykl i ponownie podawane na jej wejście.
- W praktyce najczęściej stosowane są sieci częściowo rekurencyjne, w których tylko niektóre neurony mają sprzężenia zwrotne.

Topologia SSN

Sieć rekurencyjna Hopfielda



Topologia SSN

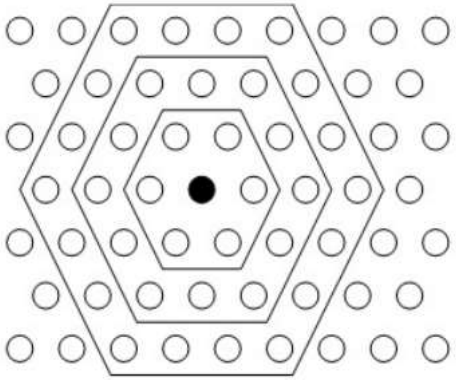
Sieć rekurencyjna Hopfielda

- Stabilność procesów osiągnięto dzięki zastosowaniu trzech zabiegów:
  - wprowadzono bardzo regularną (i prostą do realizacji, zarówno w formie programu komputerowego, jak i w postaci specjalizowanych układów elektronicznych lub optoelektronicznych) strukturę wewnętrzną sieci, polegającą na tym, że w całej sieci neurony są łączone na zasadzie „każdy z każdym”,
  - zabroniono sprzężeń zwrotnych obejmujących pojedynczy neuron. Oznacza to, że sygnał wyjściowy z danego neuronu nie może być bezpośrednio podawany na jego wejście. Nie wyklucza to sytuacji, w której sygnał wyjściowy z danego neuronu wpływa na swoją własną wartość w przyszłości, ponieważ możliwe jest sprzężenie zwrotne poprzez dodatkowe neurony pośredniczące, jednak wpływ tych dodatkowych neuronów jest zdecydowanie stabilizujący,
  - wprowadzane współczynniki wagowe muszą być symetryczne, tzn. jeśli połączenie od neuronu o numerze *i* do neuronu o numerze *j* charakteryzuje się pewnym współczynnikiem wagi w_{ij}, to dokładnie taką samą wartość ma współczynnik wagowy połączenia biegnącego od neuronu o numerze *j* do neuronu o numerze *i*.

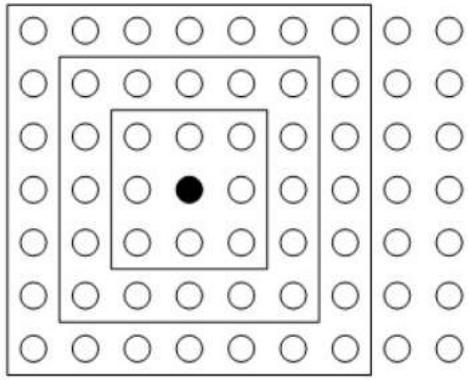
Topologia SSN

Self-Organizing Maps

- Podstawy teoretyczne Map Samoorganizujących się stworzył Teuvo Kohonen (1982 r.), stąd też często używany termin - sieć Kohonena.
- Mapę tworzy siatka komórek o stałym rozmiarze, siatka ma najczęściej budowę heksagonalną lub prostokątną:



Siatka heksagonalna

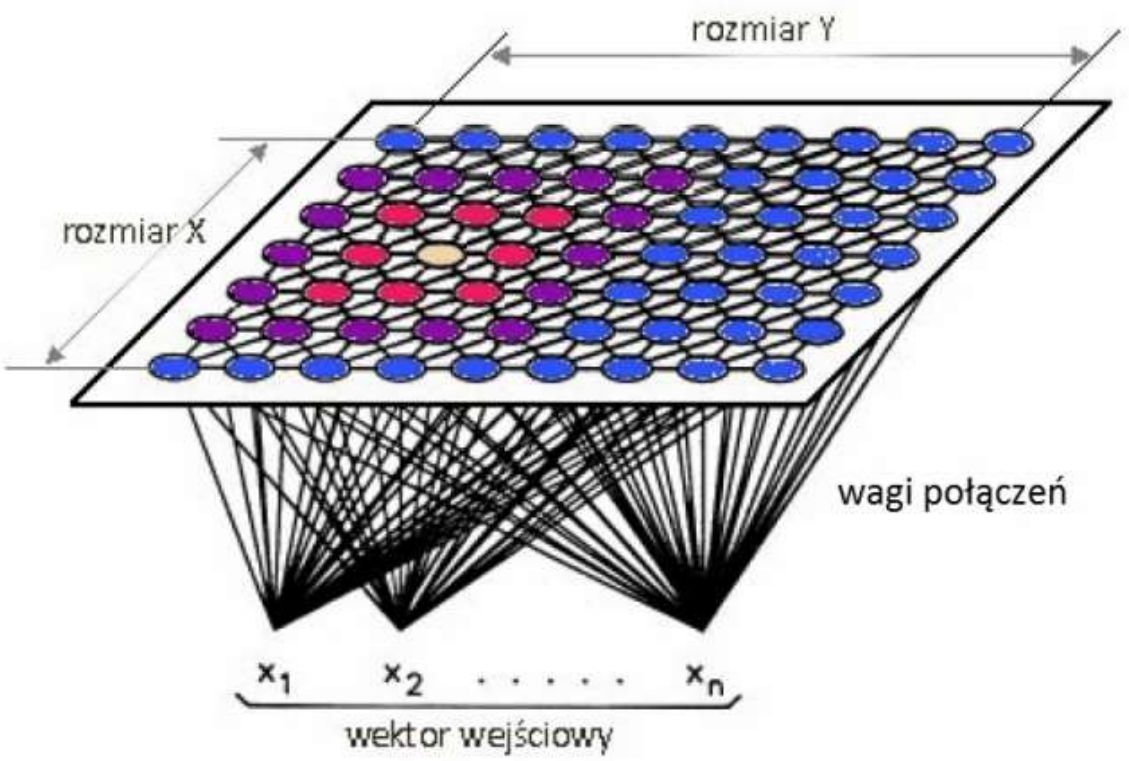


Siatka prostokątna

- Sieć ucząca się bez nadzoru.
- Cel: odwzorowanie wielowymiarowych danych za pomocą dwuwymiarowej mapy w taki sposób, aby obiekty sobie bliskie w przestrzeni obiektów były sobie bliskie na płaszczyźnie.
- Znajomość liczby grup nie jest wymagana.

Topologia SSN

Self-Organizing Maps

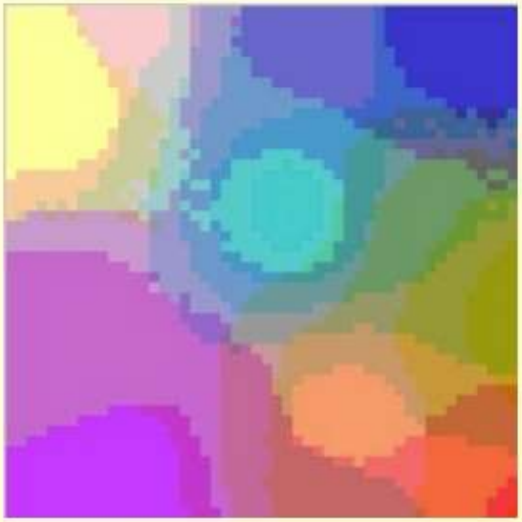


Topologia SSN

Self-Organizing Maps

Przykład grupowania kolorów

Mapa wstępna



Mapa po 100 iteracjach

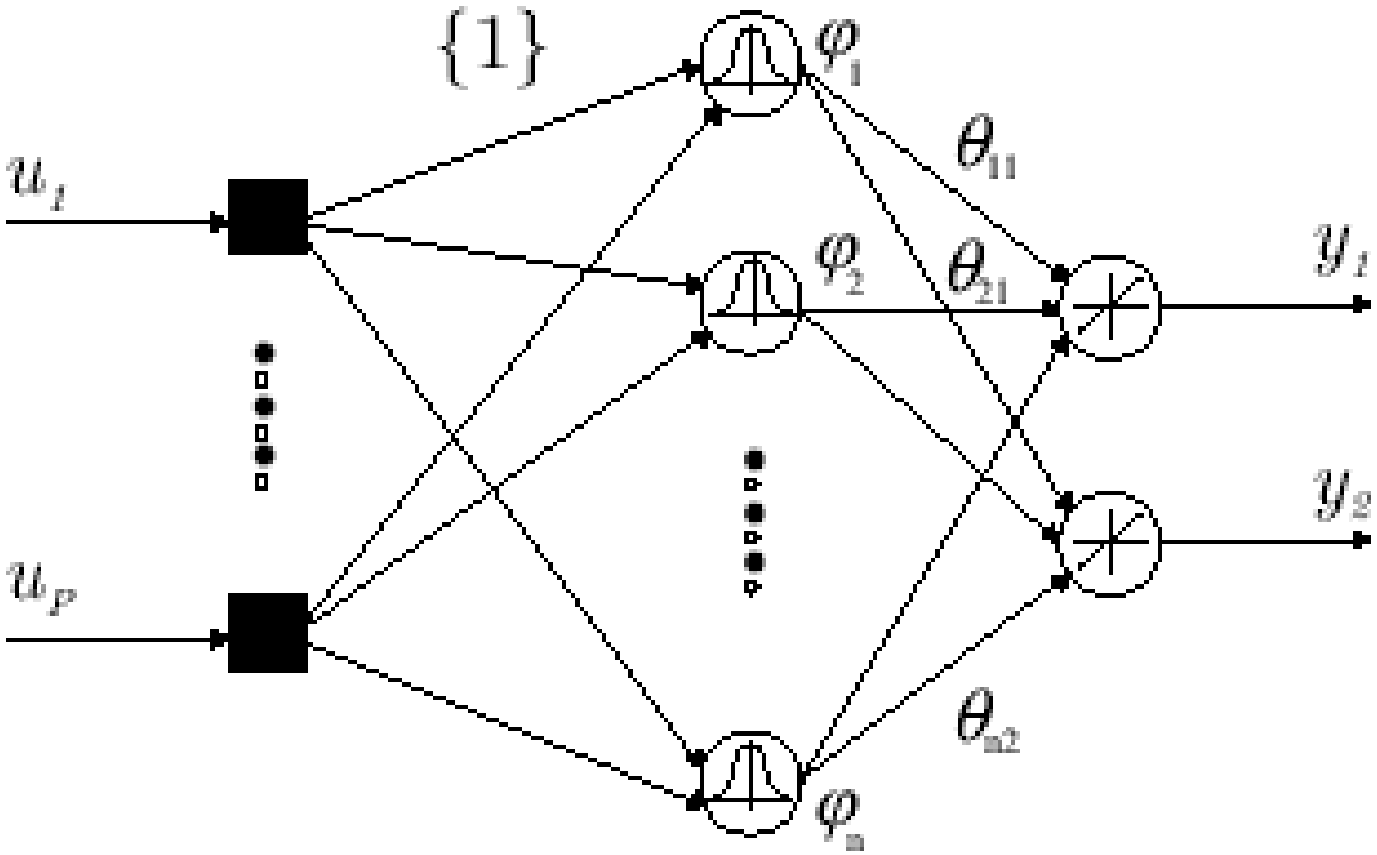
Topologia SSN

Sieć RBF (Radial Basis Function)- o radialnych funkcjach bazowych

- Są to sieci trzywarstwowe, złożone z warstwy wejściowej, ukrytej składającej się z neuronów radialnych o funkcji aktywacji najczęściej w postać funkcji Gaussa i wyjściowej o neuronach posiadających liniową bądź logistyczną funkcję aktywacji.
- Każdy neuron w sieci RBF wyznacza na swoim wejściu kwadrat odległości aktualnego wektora wejściowego od swojego wektora wag. Ponieważ zbiory sygnałów wejściowych o jednakowych wartościach funkcji wejściowej mają w tym przypadku formę hiperkul (na płaszczyźnie dwuwymiarowej będą to koła) - w związku z tym mówi się w tym przypadku o funkcjach radialnych (czasem używane jest też określenie "funkcje o symetrii radialnej" lub "funkcje o symetrii kołowej"). Funkcje te tworzą podstawę (bazę) do wyznaczania sygnału wyjściowego z sieci - stąd ich nazwa: radialne funkcje bazowe.
- Zaletą sieci jest krótki czas uczenia sieci oraz krótki czas potrzebny na wybór właściwej struktury sieci.

Topologia SSN

Sieć RBF- o radialnych funkcjach bazowych



Zastosowania SSN

- Klasyczne zastosowania SSN w zarządzaniu i ekonomii to sieci jednokierunkowe, wielowarstwowe (MLP) do klasyfikacji, rozpoznawania wzorców i predykcji.
- Marketing
  - SSN jest elementem systemu ekspertowego do śledzenia i rezerwowania miejsc w samolotach.
  - SSN służą do klasyfikowania klientów np. pod względem ryzyka handlowego.
  - Prognozowanie sprzedaży.
  - Analiza sentymentu rynkowego.
  - Rekomendacje produktowe i personalizacja treści.
  - Segmentacja rynku.
  - Prognozowanie trendów rynkowych.

Zastosowania SSN

- Wytwarzanie
  - Przewidywanie zużycia energii w procesach produkcyjnych: analizując dane historyczne związane z produkcją, temperaturą, ilością surowców i innymi czynnikami, systemy oparte na ANN mogą dostarczać prognozy zużycia energii, co pozwala na lepsze zarządzanie zasobami i optymalizację kosztów energetycznych.
  - Kontrola jakości produkcji w czasie rzeczywistym: analizując dane z sensorów i systemów wizyjnych, ANN mogą wykrywać defekty w produktach, klasyfikować wady i automatycznie dostosowywać parametry procesów produkcyjnych w celu poprawy jakości.
  - Optymalizacja (sterowanie) procesów produkcyjnych: używane do optymalizacji parametrów procesów produkcyjnych. Mogą analizować dane dotyczące wielu czynników, takich jak temperatura, ciśnienie, prędkość, aby zoptymalizować wydajność i minimalizować odpady.

Zastosowania SSN

- Wytwarzanie
  - Zarządzanie łańcuchem dostaw: wytwarzanie niejednokrotnie obejmuje zarządzanie łańcuchem dostaw. ANN mogą analizować dane dotyczące zamówień, dostaw, poziomów magazynowych, prognoz popytu itp., co pomaga w optymalizacji logistyki, minimalizacji kosztów i skracaniu czasu dostawy.
  - Diagnostyka i utrzymanie ruchu: wykorzystywane do prognozowania awarii maszyn i urządzeń w procesach produkcyjnych.
  - Autonomiczne roboty w produkcji: w środowiskach produkcyjnych rosną możliwości wykorzystania autonomicznych robotów. Sztuczne sieci neuronowe są wykorzystywane do nauki i dostosowywania zachowań robotów w celu optymalizacji ich pracy, reagowania na zmienne warunki i współpracy z ludźmi.

Zastosowania SSN

- Kontrola jakości
  - Sieć może szacować wpływ różnych zmiennych i ich kombinacji na parametry produktu (ćwiczenie).
  - Przemysł motoryzacyjny:
    - Rozpoznawanie wad w karoserii: do analizy obrazów powierzchni karoserii samochodowej w celu wykrywania niedoskonałości, wgnieceń czy zadrapań.
    - Kontrola spoin w spawaniu: analizują obrazy spawanych połączeń, identyfikując nieprawidłowości w spoinach, takie jak pęknięcia czy nadmierna porowatość.
  - Przemysł spożywczy (nie tylko):
    - Sortowanie i klasyfikacja produktów: systemy wizyjne klasyfikacji na podstawie wyglądu, kształtu czy koloru.
    - Identyfikacja wad w opakowaniach: ANN mogą analizować obrazy opakowań, pomagając w wykrywaniu wad, takich jak uszkodzenia, niewłaściwe etykiety czy braki zamknięcia.
    - System monitorowania prawidłowego napełnienia butelek.

Zastosowania SSN

- Kontrola jakości
  - Przemysł chemiczny:
    - Kontrola jakości surowców: sieci są stosowane do analizy danych sensorycznych i chemicznych w celu oceny jakości surowców używanych w produkcji chemicznej.
    - Identyfikacja składu substancji: ANN mogą analizować spektra chemiczne w celu identyfikacji składników chemicznych i utrzymania odpowiedniego składu produktów.
  - Przemysł elektroniczny:
    - Testowanie płytek drukowanych: wykorzystywane do analizy obrazów płytek drukowanych w celu identyfikacji wad, takich jak zwarcia, przerwy, pęknięcia, czy niewłaściwie umieszczone elementy.

Zastosowania SSN

- Kontrola jakości
  - Przemysł tekstylny:
    - Kontrola jakości tkanin: mogą analizować strukturę tkanin i identyfikować wady, takie jak węzły, dziury, czy nieprawidłowości w kolorze.
    - Sortowanie i klasyfikacja odzieży: ANN są wykorzystywane do sortowania odzieży na podstawie różnych kryteriów jakościowych, takich jak rozmiar, kolor czy jakość szwu.
  - Przemysł farmaceutyczny:
    - Inspekcja tabletek i kapsułek: mogą analizować obrazy tabletek i kapsułek w celu identyfikacji nieprawidłowości, takich jak ubytki, pęknięcia czy niewłaściwie umieszczone znaki.

Zastosowania SSN

- Bezpieczeństwo
  - Monitorowanie bagażu na lotniskach.
  - Śledzenie ludzi w systemach monitoringu, wykrywanie niestandardowych zachowań.
  - Systemy rozpoznawania twarzy.
  - Analiza ruchu drogowego (np. www.hayden.ai/applications).
  - Systemy detekcji intruzów, w tym przetwarzanie obrazów termicznych.
  - Detekcja oszustw i cyberbezpieczeństwo.

Zastosowania SSN

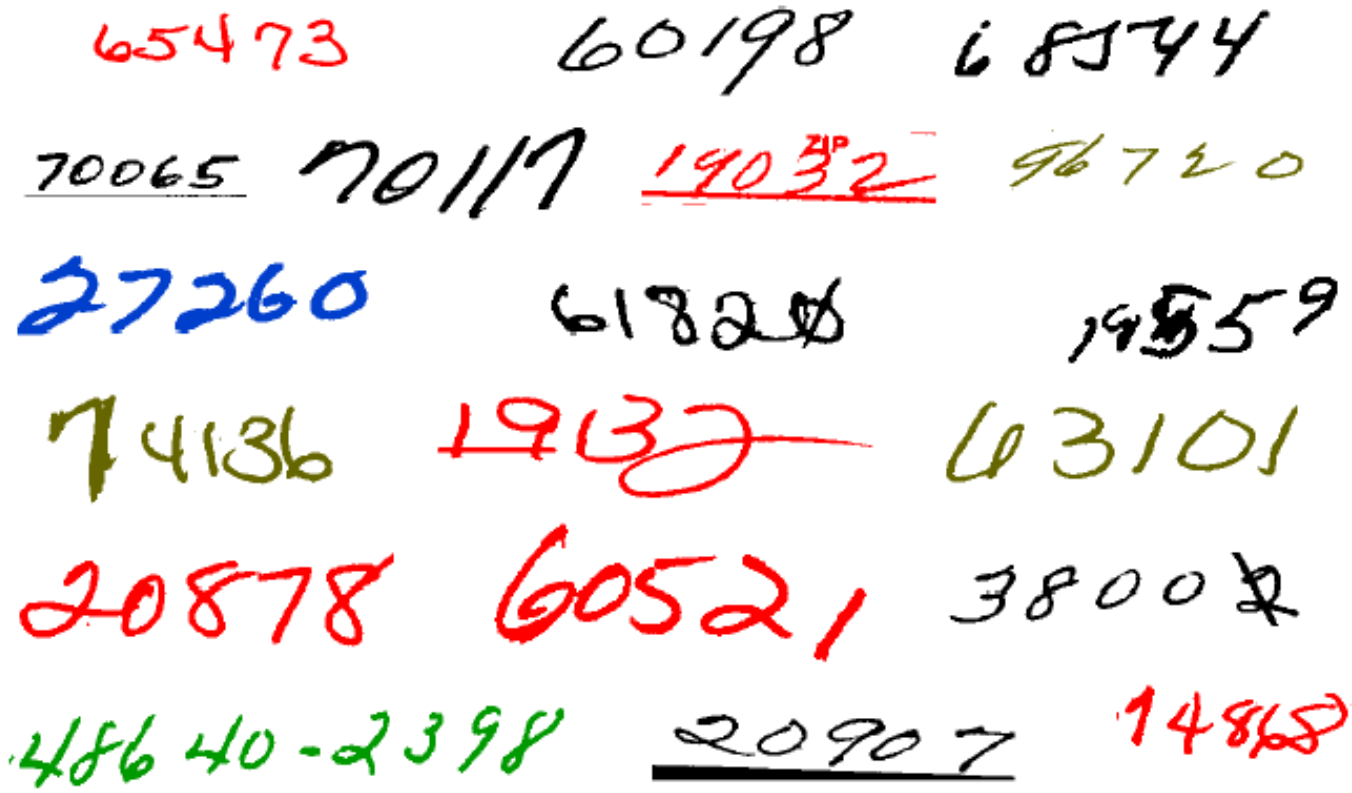
- Diagnostyka - wykorzystywane są do analizy danych z sensorów, monitorowania parametrów pracy, oraz w celu wczesnego wykrywania usterek czy awarii w różnych rodzajach maszyn i urządzeń.
  - Maszyn, urządzeń, pojazdów: sieci neuronowe są wykorzystywane do analizy danych z różnych czujników w maszynach. Mogą identyfikować nieprawidłowości w pracy maszyn, prognozować potencjalne awarie, oraz wspierać planowanie konserwacji.
  - Silników: ANN mogą analizować dane z czujników monitorujących silniki, takie jak wibracje, temperatury czy prądy. Pomagają w diagnozowaniu stanu technicznego silników, identyfikacji uszkodzeń czy zużycia elementów.
  - Urządzeń medycznych: ANN mogą być stosowane do monitorowania i diagnozowania stanu technicznego urządzeń medycznych; pomagają w utrzymaniu sprzętu w dobrym stanie oraz w zapewnianiu bezpieczeństwa pacjentów.
  - Systemów hydraulicznych i pneumatycznych: sieci mogą analizować parametry pracy systemów hydraulicznych i pneumatycznych w maszynach. Pomagają w wykrywaniu przecieków, uszkodzeń w układach ciśnienia oraz w identyfikacji problemów z elementami układów.

Zastosowania SSN

- Prognozowanie
  - Prognoza rozwoju rynku energetycznego.
  - Prognoza zapotrzebowania na energię.
  - Prognoza cen surowców naturalnych (ropa, metale, itp.) i rolnych.
  - Prognozowanie wskaźników gospodarczych.
- Rynek akcji
  - Prognoza cen akcji.
  - Kursy wymiany walut.
  - Prognozowanie indeksów giełdowych.
- **"Using Neural Networks to Predict Stock Prices" — Don't Be Fooled.** Many claim they can predict stock and crypto-currency prices using machine learning; but no one can prove a profit on live data. What's the catch?
- https://towardsdatascience.com/using-neural-networks-to-predict-stock-prices-dont-be-fooled-c43a4e26ae4e

Zastosowania SSN

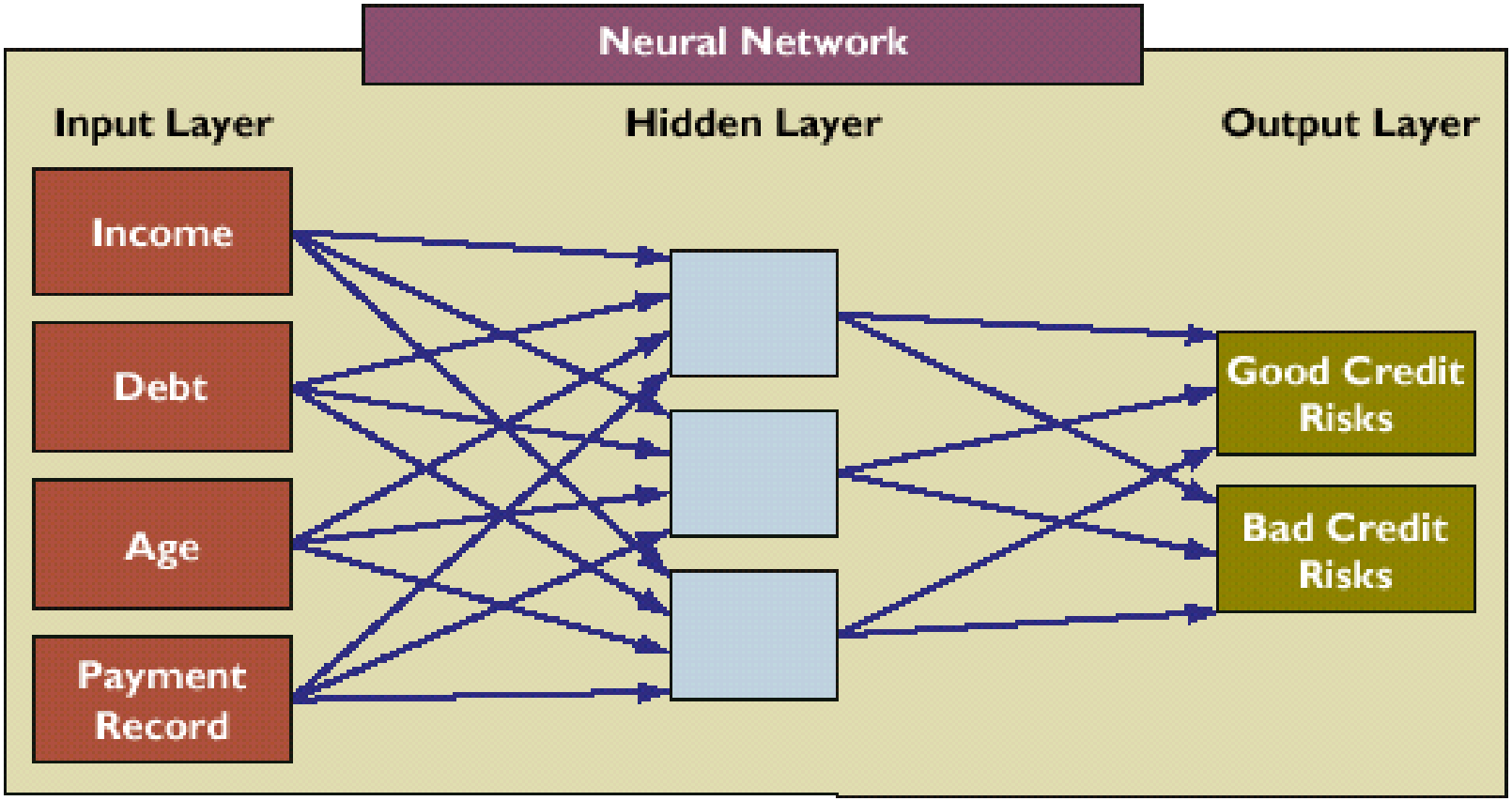
- Rozpoznawanie pisma ręcznego



Przykłady ręcznie pisanych kodów pocztowych
(*źródło*: baza danych US Postal Service)

Zastosowania SSN

- Ocena ryzyka kredytowego



Ocena kondycji finansowej

- Kondycja finansowa: stan finansowy w określonym przedziale czasowym
  - zdolność do zachowania wypłacalności (spłaty zadłużenia),
  - zdolność do przynoszenia zysków,
  - zdolność do powiększania majątku.
- Zła kondycja finansowa najczęściej kończy się bankructwem przedsiębiorstwa.
- Stosowane metody:
  - jakościowe – sposób opisowy,
  - ilościowe:
    - deterministyczne – proste wskaźniki;
    - stochastyczne:
      1. statystyczne – analiza trendu,
      2. dyskryminacyjne – wielowymiarowa analiza statystyczna;
    - sieci neuronowe, algorytmy ewolucyjne.

Ocena kondycji finansowej

- Analiza dyskryminacyjna
- Funkcję dyskryminacyjną można określić wzorem:

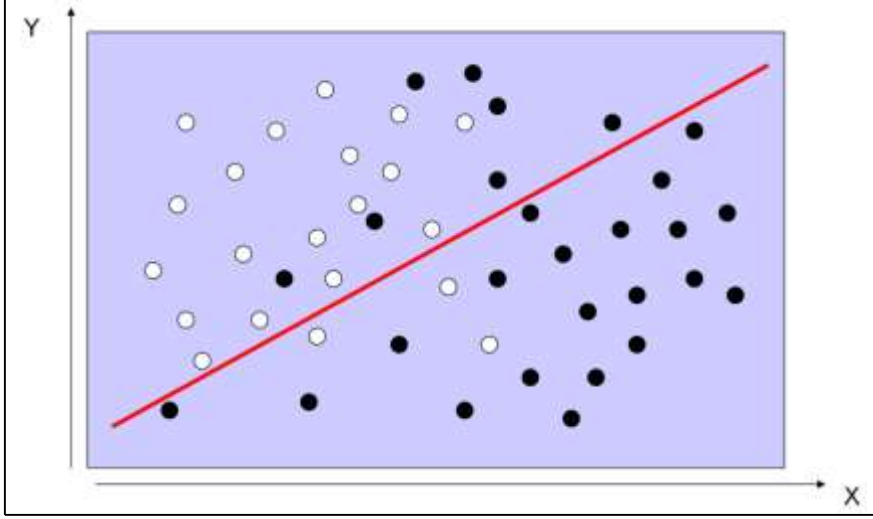
$$Z = W_1 X_1 + W_2 X_2 + \ldots + W_n X_n$$

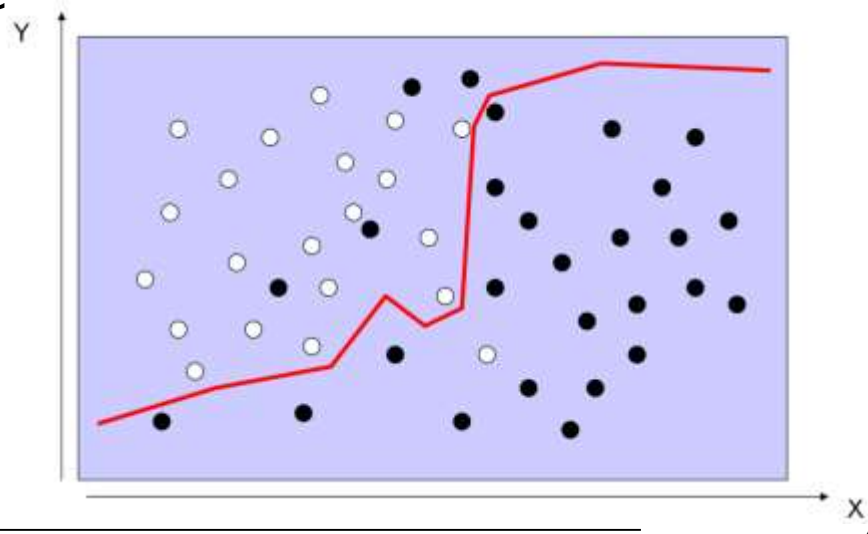
gdzie:

Z – wartość funkcji dyskryminacyjnej,

W_i – wagi i-tej zmiennej (np. wskaźników finansowych),

X_i – zmienne objaśniające modelu





Ocena kondycji finansowej

- Analiza dyskryminacyjna - model Altmana (1968)
- Z = 6,56 * X1+ 3,26 * X2 + 6,72 * X3 + 1,05 * X4
- X1 = majątek obrotowy / aktywa ogółem
- X2 = zysk netto / aktywa ogółem
- X3 = EBIT / aktywa ogółem
- X4 = kapitał własny / zobowiązania ogółem
- wartości progowe: 1,10 i 2,60

Ocena kondycji finansowej

- **M. Wudarczyk, D. Kieszkowski** - Sieci neuronowe w analizach finansowych. *Prognozowanie bankructw* (2006) Wyniki:
- Wszystkie sieci osiągają błąd klasyfikacji rzędu 20%-30%.
- Dla porównania, model Altmana dla tych danych ma błąd rzędu 25% - 40%.

Ocena ryzyka kredytowego

- **M. D. Odoma i R. Shardy (1990)**
  Odoma i Shardy jako pierwsi użyli sztucznych sieci neuronowych do klasyfikacji kredytobiorców pod względem ryzyka kredytowego. Zastosowali oni perceptron trójwarstwowy, z jednym neuronem w warstwie wyjściowej, pięcioma neuronami w warstwie ukrytej i pięcioma neuronami w warstwie wejściowej tj.:
  - kapitał pracujący / aktywa ogółem,
  - skumulowane zyski reinwestowane / aktywa ogółem,
  - (zysk brutto + odsetki) / aktywa ogółem,
  - wartość rynkowa kapitału własnego / wartość księgowa kapitału obcego,
  - przychód ze sprzedaży / dochód ogółem.

Ocena ryzyka kredytowego

- Do badań została wybrana próba złożona z 64 kredytobiorców wypłacalnych i 65 niewypłacalnych. W trakcie uczenia wykorzystany został algorytm wstecznej propagacji błędów ze współczynnikiem momentum. Została przyjęta klasyfikacja dychotomiczna (0;1):
  - dla wartości wyjścia $\geq 0{,}5$ oznaczało to przedsiębiorstwo wypłacalne,
  - dla wartości wyjścia $< 0{,}5$ oznaczało to przedsiębiorstwo niewypłacalne.
- Parametrem zmienianym w analizie był stosunek przedsiębiorstw wypłacalnych do niewypłacalnych w próbie treningowej.

Ocena ryzyka kredytowego

- **A. Bechler (2003)**
  Przeprowadził porównanie efektywności modeli ekonometrycznych i sieci neuronowych we wspomaganiu decyzji kredytowych dotyczących kredytów konsumenckich na zakup samochodu. Analizowana próba zawierała 200 zestawów danych, z czego 176 decyzji było pozytywnych, natomiast pozostałe 24 to decyzje negatywne. Zestaw cech został poddany wstępnej analizie poprzez pozbycie się ze zbioru cech rzadkich oraz silnie skorelowanych z innymi cechami, ostateczny zbiór zawierał 35 cech diagnostycznych, które znormalizowano otrzymując wartości z przedziału [0,1].
- W analizie wykorzystano trzywarstwową sieć jednokierunkową, z liczbą neuronów wejściowych wahającą się od 3 do 34, natomiast liczba neuronów ukrytych wyniosła od 2 do 17. Na funkcję aktywacji została wybrana zarówno dla warstwy wyjściowej jak i ukrytej funkcja logistyczna, ze współczynnikiem kształtu $\beta = 1$. Warstwa wyjściowa składała się z jednego neuronu przyjmującego wartości 0, gdy kredyt nie został udzielony i 1 - gdy został udzielony, z liniową funkcją aktywacji.
- Skorzystano z programu STATISTICA Neural Networks (SSN) 3.0. Jako algorytmu treningowego użyto metody wstecznej propagacji błędu ze współczynnikiem uczenia $\eta = 0{,}6$ oraz momentum $\lambda = 0{,}3$.

Ocena ryzyka kredytowego

- Analizując wyniki autor doszedł do wniosku, że zarówno modele ekonometryczne, jak i sieci neuronowe są efektywnym narzędziem pomocnym przy ocenie ryzyka kredytowego (wskaźnik trafień dla większości modeli przekroczył 90%), zwłaszcza, że analiza ta nie jest prosta - opiera się także na subiektywnych miarach (czynnik ludzki) oraz zawiera element przypadkowości. Autor zauważył pewną przewagę sieci neuronowych nad modelami ekonometrycznymi tj. zdolność do wykrywania głęboko ukrytych związków we wzorcach empirycznych, wadą natomiast jest brak interpretacji współczynników wagowych oraz słabo poznane zasady funkcjonowania. Dlatego też celowe jest stosowanie obu metod: powszechną opinia jest taka, iż mając do wyboru kilka modeli klasyfikacji potencjalnych kredytobiorców, najlepsze rezultaty daje łączne uwzględnienie wyników generowanych przez te modele.

Najdziwniejsze zastosowania

- ANALIZA MOŻLIWOŚCI PROGNOZOWANIA PRZEMIESZCZEŃ GLEBY PODCZAS ORKI ZA POMOCĄ KLASYCZNYCH METOD STATYSTYCZNYCH ORAZ SZTUCZNYCH SIECI NEURONOWYCH - Inżynieria Rolnicza 2(90)/2007
- OCENA STOPNIA ZUŻYCIA TECHNICZNEGO WYBRANEJ GRUPY BUDYNKÓW MIESZKALNYCH ZA POMOCĄ SZTUCZNYCH SIECI NEURONOWYCH – Zastowania metod statystycznych w badaniach naukowych, StatSoft Polska 2003

Inne zastosowania

Examples of ANNs developed by UGA students

1) **Egg defect detection system**
   Inputs were based on computer vision images of eggs. Histograms of the pixel intensities were generated for egg images and used to train the ANNs.
2) **Aflatoxin prediction** in peanuts.
3) **Diagnosing heart disease** from Electrocardiogram output.
4) **Rate of bioconversion** in a composting system. Inputs were *temperature*, *moisture content* and *time*.
5) **Predicting Commodity Price**.
6) **Insect pest management** decision support.
7) **Estimation** of daily maximum and minimum temperature and solar radiation.

Inne zastosowania

Examples of ANNs developed by UGA students

8) **Prediction of dielectric properties** of a material.
9) **Classifying music** into categories of Baroque, Classical and Romantic.
10) **Determining land use** using aerial images.
11) **Frost Damage Risk Management** by
    a) predicting air temperature on hourly basis for the next twelve hours.
    b) classifying the next twelve hours as having a frost event or not.
12) **Interpreting CT (Computed Tomography) scan data** to non-destructively identify defects in apples.