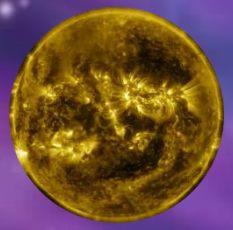
METODY INŻYNIERII WIEDZY

KNOWLEDGE ENGINEERING AND DATA MINING

UCZENIE GŁĘBOKIE I GŁĘBOKIE SIECI NEURONOWE

DEEP LEARNING AND DEEP NEURAL NETWORKS



Akademia Górniczo-Hutnicza

Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

Katedra Automatyki i Inżynierii Biomedycznej, Laboratorium Biocybernetyki

30-059 Kraków, al. Mickiewicza 30, paw. C3/205

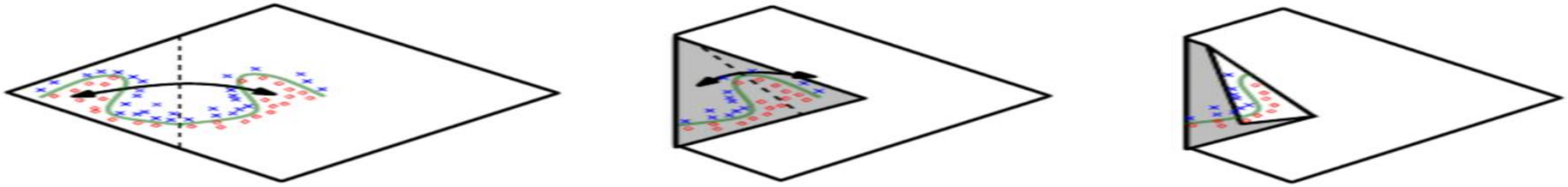
horzyk@agh.edu.pl, Google: Adrian Horzyk

Adrian Horzyk

IDEA UCZENIA GŁĘBOKIEGO

Tradycyjne uczenie sieci typu feedforward charakteryzuje się tym, iż wagi inicjalizowane są małymi losowymi wartościami, które niekoniecznie gwarantują znalezienie minimum globalnego w trakcie adaptacji algorytmów gradientowych (*gradient- based optimization algorithms*).

Uczenie głębokie umożliwia wyznaczanie wag dla poszczególnych warstw sieci stopniowo po to, by poszczególne warstwy reprezentowały cechy wspólne wzorców uczących i na tej bazie mogły tworzyć się reprezentacje bardziej skomplikowanych cech w kolejnych warstwach sieci głębokich.



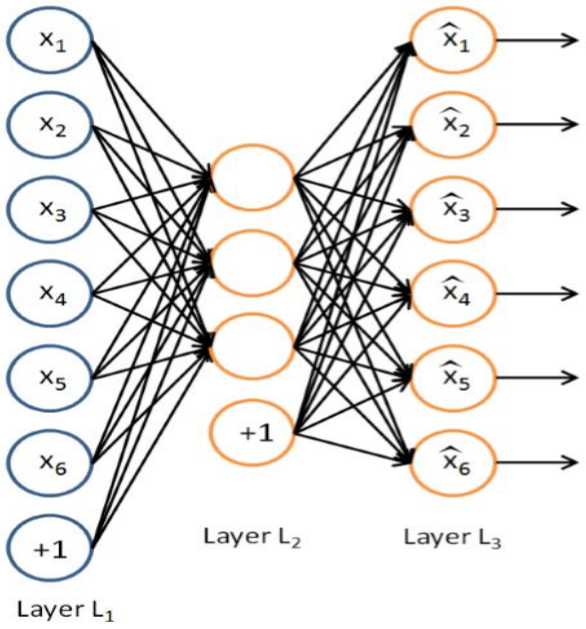
Gdy po zbudowaniu wielowarstwowej sieci głębokiej zostanie zastosowana metoda uczenia z propagacją wsteczną błędów, sieć będzie miała już wagi w poszczególnych warstwach odpowiednio zainicjowane, więc będzie miała mniejszą skłonność do utknięcia w niekorzystnych minimach lokalnych.

Ponadto wtedy metoda propagacji wstecznej zastosowana na wielu warstwach sieci głębokiej będzie próbowała bardziej dostroić poszczególne wagi takiej sieci niż szukać modelu od początku startując z losowych wartości wag, jak ma to miejsce w zwykłym modelu sieci typu MLP.

IDEA UCZENIA GŁĘBOKIEGO

Idea uczenia głębokiego wzięła się z potrzeby lepszego dopasowania parametrów modelu do danych uczących, by otrzymać lepsze zdolności uogólniające.

Zauważono, iż wielowarstwowe sieci neuronowe typu MLP uczone algorytmem propagacji wstecznej (*backpropagation*) mają tendencję do dokonywania coraz mniejszych zmian w warstwach oddalonych od wyjścia sieci w trakcie propagacji wstecznej błędów. Nie dochodzi więc do skutecznego i pożądanego procesu ekstrakcji cech.



Powstał więc pomysł „zmuszenia" sieci neuronowej do tego w taki sposób, iż sieć rozbudowywana jest stopniowo o kolejne warstwy dopiero w momencie, gdy w warstwach poprzednich takie cechy się pojawią i mogą być wykorzystane do budowy modelu.

W tym celu stosujemy np. sieci MLP typu *autoencoder*, których zadaniem jest aproksymacja identycznościowa wykorzystująca warstwę ukrytą składającą się z mniejszej ilości neuronów ukrytych niż ilość wejść lub wyjść takiej sieci.

W wyniku procesu uczenia w warstwie ukrytej z konieczności kompresji dochodzi do powstania reprezentacji wspólnych cech wzorców uczących, co działa podobnie jak inne metody redukcji wymiaru danych typu PCA lub ICA.

DOSTRAJANIE GŁĘBOKI SIECI NEURONOWYCH

Podstawowym zagadnieniem w uczeniu maszynowym jest odnalezienie modelu, który będzie w stanie nie tylko dopasować się do danych uczących, lecz również do innych danych, poprawnie na nich działając. W tym celu badane jest dopasowanie modelu na danych testowych lub walidacyjnych w celu określenia maksymalnego lub średniego błędu, który wskazuje na jakość danego modelu.

Problem dopasowania głębokich sieci neuronowych (*regularization of deep learning*) jest bardzo istotny, gdyż sieci wielowarstwowe potrafią się bardzo dobrze dopasować się do danych uczących (tzw. problem zbytniego dopasowania się – *overfitting problem*), lecz na danych testowych mogą zachowywać się niepoprawnie. Im więcej warstw tym potencjalnie tworzone są bardziej skomplikowane funkcje nieliniowe, które pogłębiają ten problem.

Istnieje wiele strategii dostrajających różne rodzaje głębokich sieci neuronowych:

- Strategie stosujące współczynniki kary i/lub nagrody
- Strategie dodające dodatkowe ograniczenia
- Strategie zmniejszające wariancję
- Strategie wczesnego zatrzymania procesu uczenia
- Strategie polegające na wybieraniu prostszych modeli lub karzących bardziej skomplikowane
- Strategie wykorzystujące metody zespołowe (*ensemble methods*)

GŁĘBOKIE SIECI NEURONOWE

Głębokie sieci neuronowe mogą być różnego typu.
Najczęściej są nimi głębokie MLP, czyli *deep multilayer perceptrons*, nazywane też *deep feedforward networks* lub *feedforward neural networks*, które aproksymują pewną funkcję *f*.

Nazwa *feedforward* bierze się stąd, iż dane (informacja) przepływa przez sieć w jedną stronę – od wejścia do wyjścia sieci.
Mówiąc inaczej, taka sieć nie posiada sprzężeń zwrotnych (*feedback connections*). Jeśli takie połączenia istnieją, takie sieci nazywamy zwykle *recurrent neural networks*.

Głębokość (*depth*) sieci bierze się z tego, iż struktura sieci typu *feedforward* składa się z wielu warstw, które są ze sobą połączone.

Stosowane funkcje matematyczne do modelowania zachowania sztucznych neuronów są konstruowane przez matematyków i inżynierów, więc ich związek z modelem biologicznego mózgu i jego funkcjami jest bardzo luźny. Możemy więc takie sieci nazywać też maszynami funkcji aproksymujących *(function approximation machines)*,
które są zaprojektowane do statystycznego uogólniania niż do modelowania funkcji mózgowych.

PODSTAWOWE RODZAJE SIECI GŁĘBOKICH

1. **Głębokie sieci neuronowe do uczenia nienadzorowanego lub generatywnego**
   wychwytujące korelacje wyższych rzędów pomiędzy danymi w celu analizy wzorców oraz syntezy zamiarów, gdy brakuje informacji na temat klas.
2. **Głębokie sieci neuronowe do uczenia nadzorowanego**
   dostarczające dyskryminacji wzorców dla celów ich klasyfikacji, dlatego są czasami nazywane dyskryminatywnymi głębokimi sieciami neuronowymi (*discrimative deep neural networks*)
3. **Hybrydowe głębokie sieci neuronowe**
   których celem jest dyskryminacja wspomagana przez generatywne lub nienadzorowane głębokie sieci neuronowe.

GŁĘBOKIE SIECI NEURONOWE

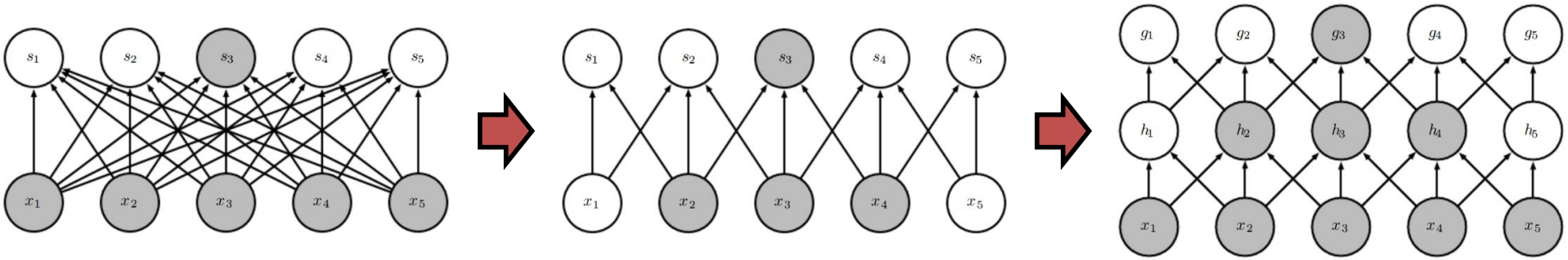
- ✓ Wykorzystywane głównie do klasyfikacji, rozpoznawania obrazów i mowy.
- ✓ Ich potęga drzemie w zdolności do ekstrakcji odpowiednich cech i równoczesnym dokonywaniu dyskryminacji. [p.19]
- ✓ Głębokie uczenie związane jest z:
  - Głębokimi sieciami neuronowymi (DNN – deep neural networks)
  - Rekurencyjnymi sieciami neuronowymi (RNN – recurrent neural networks)
  - Konwolucyjnymi sieciami neuronowym (CNN – convolutional neural networks)
  - Generatywnymi nienadzorowanymi sieciami, np. Restrykcyjna Maszyna Boltzmanna (RBM – Restricted Boltzmann Machine), DBN - Deep Belief Networks, głęboka maszyna Boltzmanna (DBM – Deep Boltzmann Machine)

EKSTRAKCJA CECH

Podstawowym zagadnieniem związanym z tworzeniem wysokiej jakości modeli neuronowych jest automatyczna ekstrakcja i reprezentacja cech, jakie występują w danych uczących. Wysokiej jakości model neuronowy jest w stanie takie cechy znaleźć i reprezentować. Ale jak to osiągnąć?

Istnieje wiele metod ekstrakcji takich cech. Metodologia głębokiego uczenia jest jedną z nich, gdyż hierarchiczne i stopniowe uczenie sprzyja odnajdywaniu w danych takich cech oraz łączeniu ich w celu budowy bardziej stabilnych i lepiej uogólniających modeli neuronowych.

Proces ekstrakcji cech można też wspierać w taki sposób, iż nie dajemy każdemu neuronowi dostępu do wszystkich danych (tzn. nie tworzymy połączeń na zasadzie każdy z każdym), lecz wybieramy połączenia selektywnie na podstawie pewnych warunków lub reguł albo ograniczamy zakres połączeń do najbliższych:



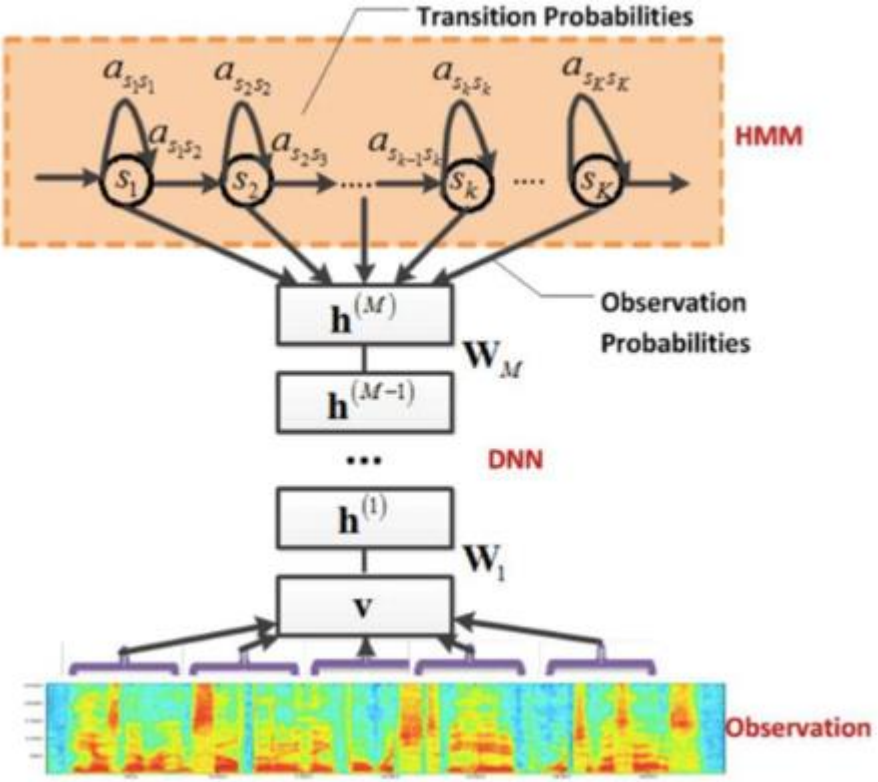
Z takimi strategiami spotykamy się np. w konwolucyjnych sieciach neuronowych (convolutional neural networks – CNN).

Głębokie uczenie (Deep Learning)

- ✓ Klasą technik uczenia maszynowego
- ✓ Wiele warstw przetwarza informacje w wielu krokach korzystając z hierarchicznego uczenia nadzorowanego w celu nienadzorowanego wyznaczenia cech dla celów analizy lub klasyfikacji.
- ✓ Celem jest wyznaczenie hierarchii cech lub form reprezentacji danych w taki sposób, iż cechy bardziej złożone są definiowane przez cechy prostsze.

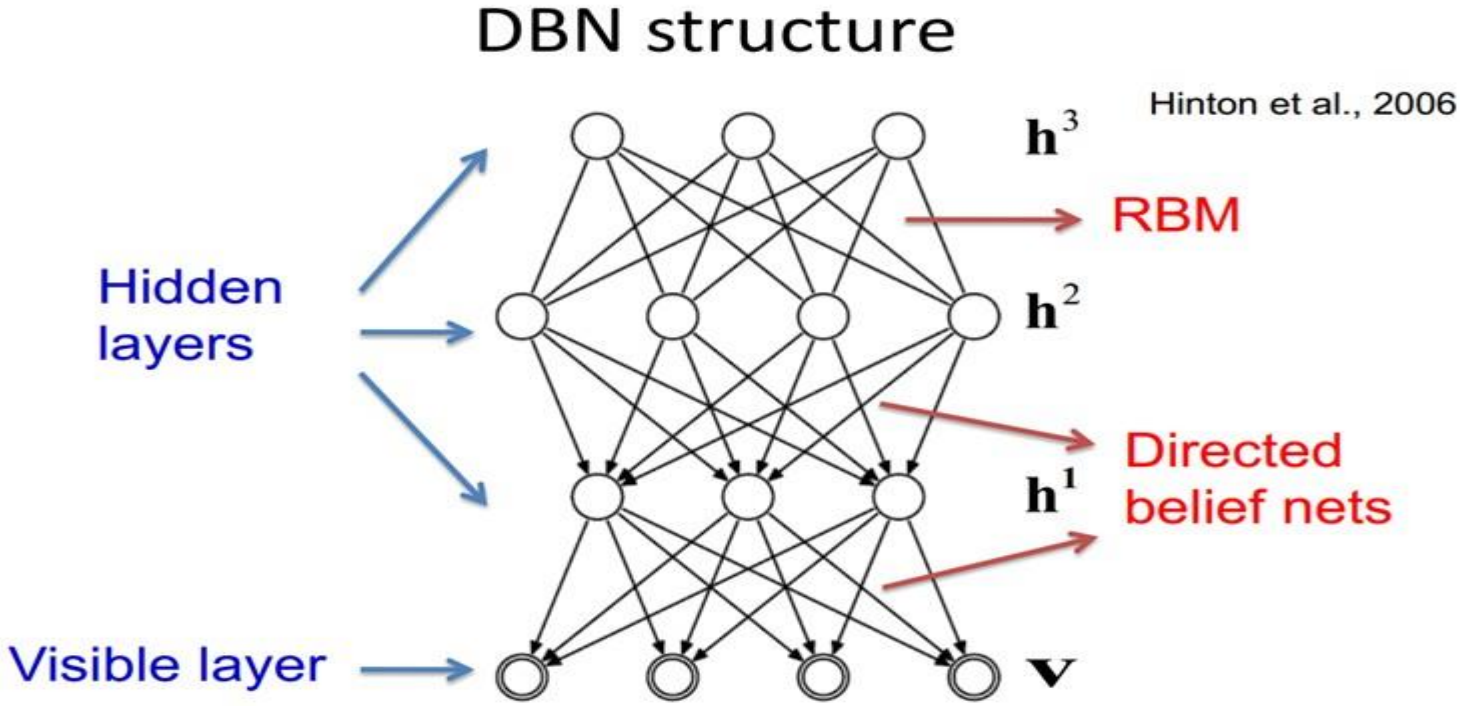
Głębokie sieci neuronowe
Deep Neural Networks (DNN)

- ✓ To wielowarstwowy perceptron z wieloma warstwami ukrytymi, pomiędzy którymi połączenia są na zasadzie każdy z każdym.
- ✓ Wagi tych połączeń są często (aczkolwiek nie zawsze) inicjowane z wykorzystaniem wstępnego uczenia nadzorowanego lub nienadzorowanego.



Deep Belief Networks (DBN)

- ✓ To probabilistyczne modele generatywne składające się z wielu stochastycznych warstw zawierającymi ukryte zmiennie.
- ✓ Najwyższe dwie warstwy zawierają nieskierowane (dwustronne) symetryczne połączenia pomiędzy ich neuronami.
- ✓ Niższe warstwy mają połączenia z wyższą warstwą.



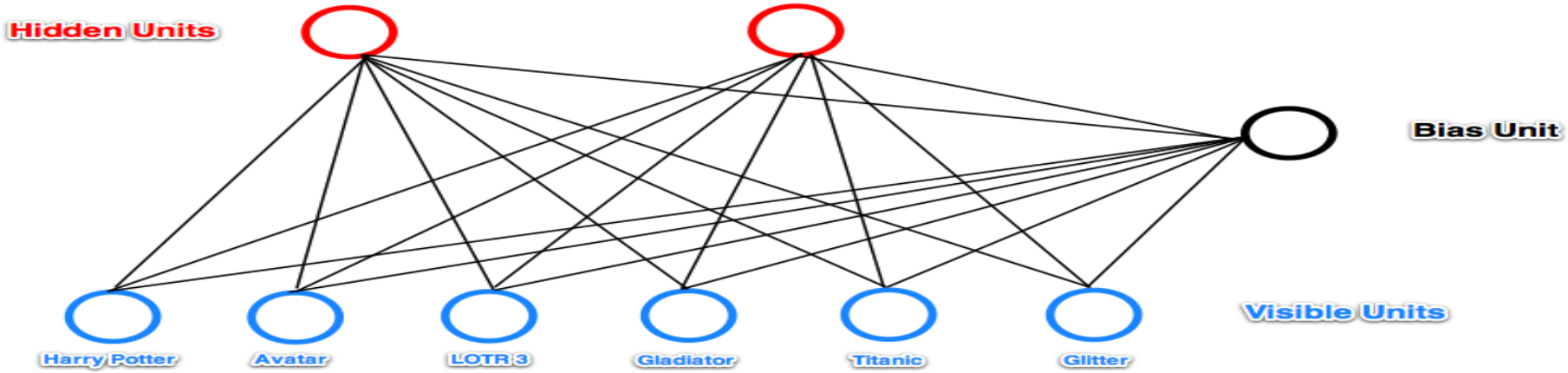
$$P(\mathbf{v},\mathbf{h}^1,\mathbf{h}^2,\ldots,\mathbf{h}^l) = P(\mathbf{v}\,|\,\mathbf{h}^1)P(\mathbf{h}^1\,|\,\mathbf{h}^2)\ldots P(\mathbf{h}^{l-2}\,|\,\mathbf{h}^{l-1})P(\mathbf{h}^{l-1},\mathbf{h}^l)$$

Maszyna Boltzmanna (Boltzmann machine BM)

- ✓ Sieć dwuwarstwowa posiadająca symetryczne połączenia pomiędzy węzłami (neuronami) tych warstw.
- ✓ W węzłach podejmowane są decyzje stochastyczne o włączeniu lub wyłączeniu.

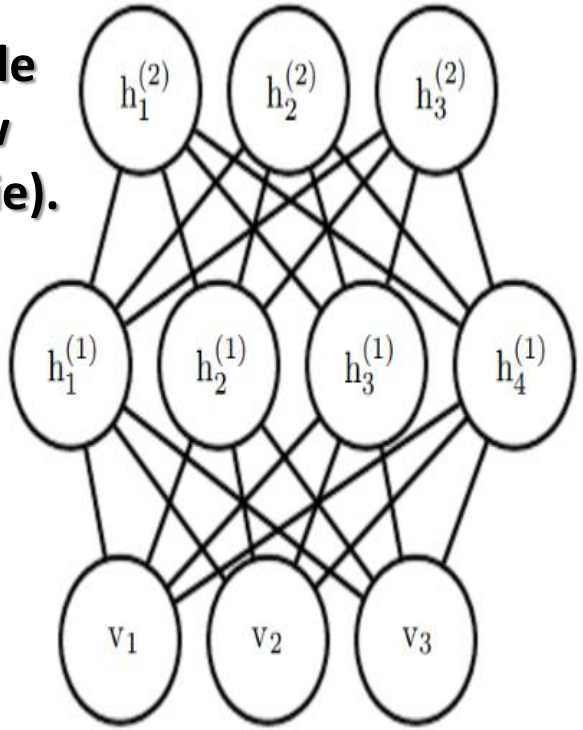
Restrykcyjna Maszyna Boltzmanna (Restricted Boltzmann machine RBM)

- ✓ Specjalny rodzaj maszyny Boltzmana zawierającej warstwę widocznych węzłów oraz warstwę węzłów ukrytych.



RESTRICTED BOLTZMAN MACHINE RBM

- ✓ **Jest specjalnym typem losowych pól Markowa, które posiadają jedną ukrytą warstwę stochastyczną (zwykle Bernouliego) oraz jedną warstwę widocznych węzłów stochastycznych (zwykle Bernouliego lub Gaussowskie).**



- ✓ **RBM jest grafem dwudzielnym, w którym połączenia występują pomiędzy każdym węzłem ukrytym i widocznym, natomiast nie ma żadnych połączeń pomiędzy węzłami ukrytymi oraz żadnych pomiędzy węzłami widocznymi.**

- ✓ **Do wyznaczenia wag stosujemy różnicę pomiędzy energią danych i energią modelu reprezentowaną przez układ:**

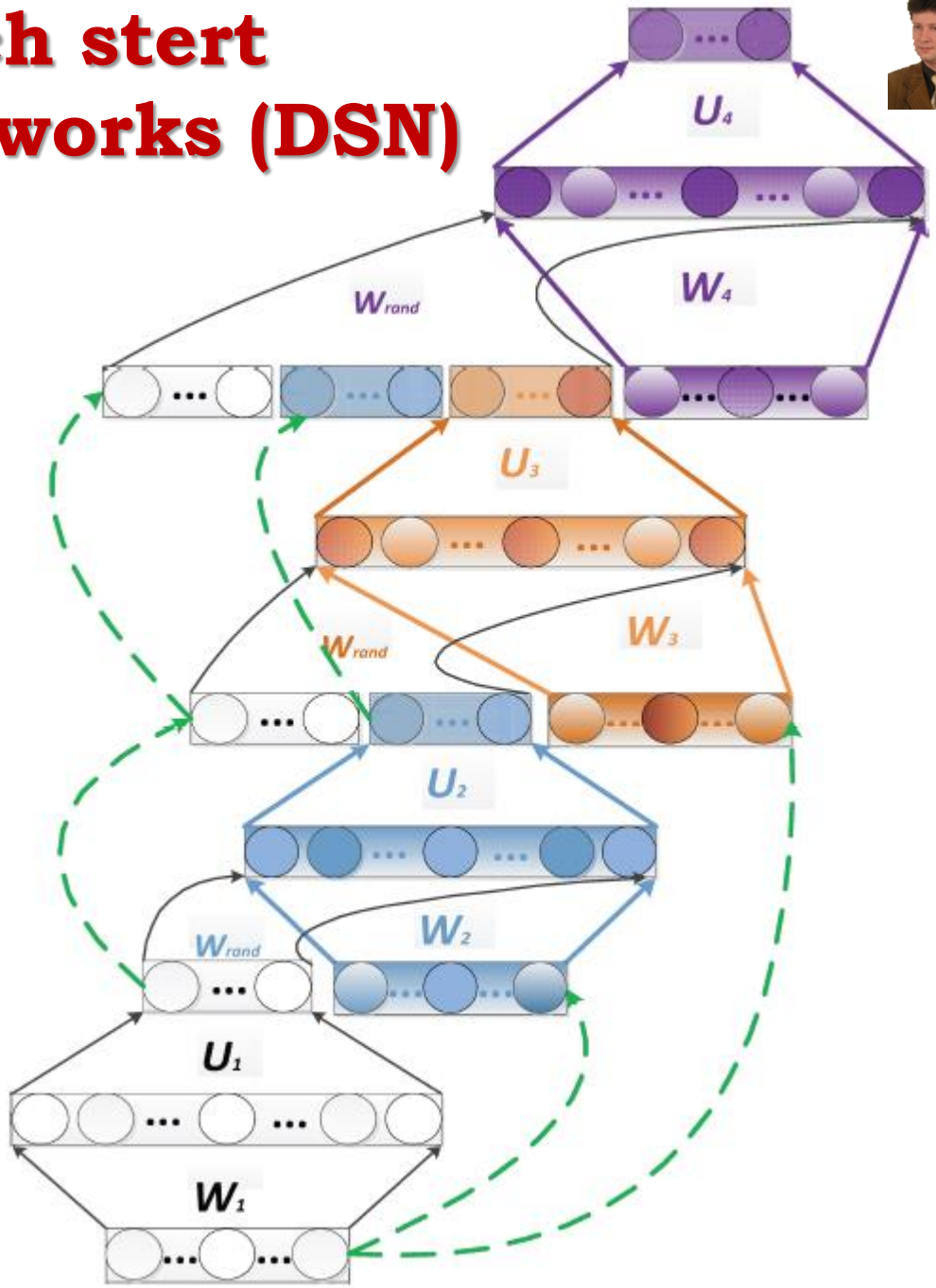
$$\Delta w_{ij} = E_{\text{data}}(v_i h_j) - E_{\text{model}}(v_i h_j)$$

- ✓ **Dla połączeń Bern-Bern:** $E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i=1}^{I}\sum_{j=1}^{J} w_{ij} v_i h_j - \sum_{i=1}^{I} b_i v_i - \sum_{j=1}^{J} a_j h_j$

- ✓ **Dla połączeń Gauss-Bern:** $E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i=1}^{I}\sum_{j=1}^{J} w_{ij} v_i h_j - \frac{1}{2}\sum_{i=1}^{I}(v_i - b_i)^2 - \sum_{j=1}^{J} a_j h_j$

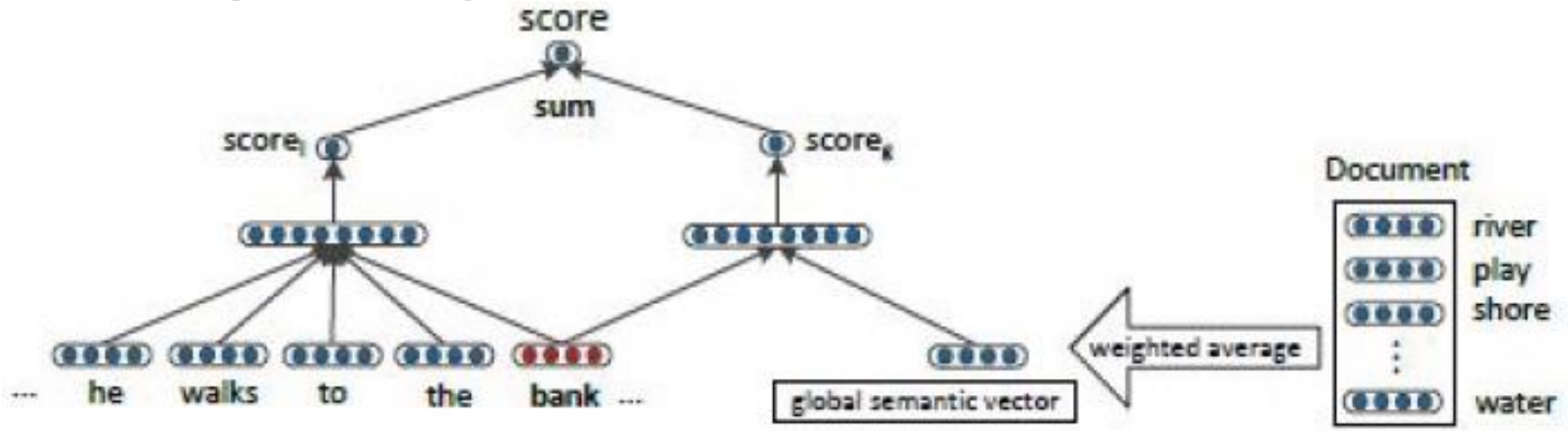
Sieci głębokich stert
Deep Stacking Networks (DSN)

- ✓ Wykorzystują zarówno dane wejściowe jak i wyjściowe do dalszej adaptacji w kolejnych warstwach sterty sieci.
- ✓ Poszczególne podsieci na stercie sieci uczone są osobno.



Głębokie uczenie w przetwarzaniu języka naturalnego (NLP – natural language processing)

- ✓ Celem jest określenie prawdopodobieństwa wystąpienia pewnej sekwencji słów lub innych elementów lingwistycznych (np. liter, znaków, fonemów).
- ✓ Stosowane jest do korekty, translacji, parsowania, klasyfikacji elementów języka.
- ✓ W NLP oraz do modelowania języka (*LM – language models*) stosowane są najczęściej N-gramy (czyli N elementowe sekwencje słowne)



PRZYKŁAD STRATEGII UCZENIA GŁĘBOKIEGO

Załóżmy, że mamy do rozwiązania pewne zadanie aproksymacji, regresji lub klasyfikacji oraz pewien zbiór danych uczących, definiujących wektory wejściowe oraz pożądane wyjścia. Poszukujemy najlepszego głębokiego modelu sieci neurnowej:

1. Posiadamy pewien zbiór różnych funkcji nieliniowych (sigmoidalne, trygonometryczne, logarytmiczne, wielomianowe, radialne itd.)
2. W kolejnych krokach adaptacji i rozbudowy głębokiej dodajemy do sieci neuronowej zestaw nowych neuronów podłączonych do wszystkich poprzednich, z których każdy reprezentuje inną funkcję nieliniową.
3. Uczymy te neurony (np. metodą propagacji wstecznej) tak długo, dopóki neurony nie wykształcą reprezentacji pewnych cech/klas/aproksymacji reprezentujących możliwie największą ilość wzorców uczących.
4. W trakcie uczenia można usuwać połączenia, których wartości wag fluktuują lub zbliżają się do zera, gdyż zadaniem tych neuronów jest reprezentacja pewnych wspólnych cech lecz zwykle nie dla wszystkich poprzednich neuronów.
5. Z ostatnio dodanych do sieci neuronów pozostawiamy w sieci tylko jeden neuron (ew. kilka), któremu/którym udało się reprezentować największą ilość wzorców uczących, a wszystkie pozostałe usuwamy.
6. Powracamy do punktu 2. tak długo, dopóki nie będziemy posiadali zbioru neuronów prawidłowo reprezentujących pożądane wyjścia (regresyjne, aproksymujące lub klasyfikujące) dla wszystkich danych uczących.

UCZENIE SIECI WIELOWARSTWOWYCH

Sieci wielowarstwowe uczymy zwykle metodami nadzorowanymi (tzn. z nauczycielem):

1. **Podajemy na wejście sygnał wejściowy (zwykle w postaci wektora lub macierzy danych)**
2. **Obliczamy wartości wyjściowe neuronów w 1. warstwie i poprzez ich połączenia z neuronami w 2. warstwie podajemy te wartości jako sygnały wejściowe dla neuronów w 2. warstwie.**
3. **Obliczamy wartości wyjściowe w kolejnych warstwach tym samym sposobem.**
4. **Wartości wyznaczone w ostatniej warstwie stanowią zarazem odpowiedź sieci na podany sygnał wejściowy.**
5. **Sygnał ten porównujemy z wzorcowym (określonym przez nauczyciela) i wyznaczamy błąd.**
6. **Korzystając metod gradientowych propagujemy błąd wstecz przez sieć i dokonujemy korekty wartości wag połączeń synaptycznych.**

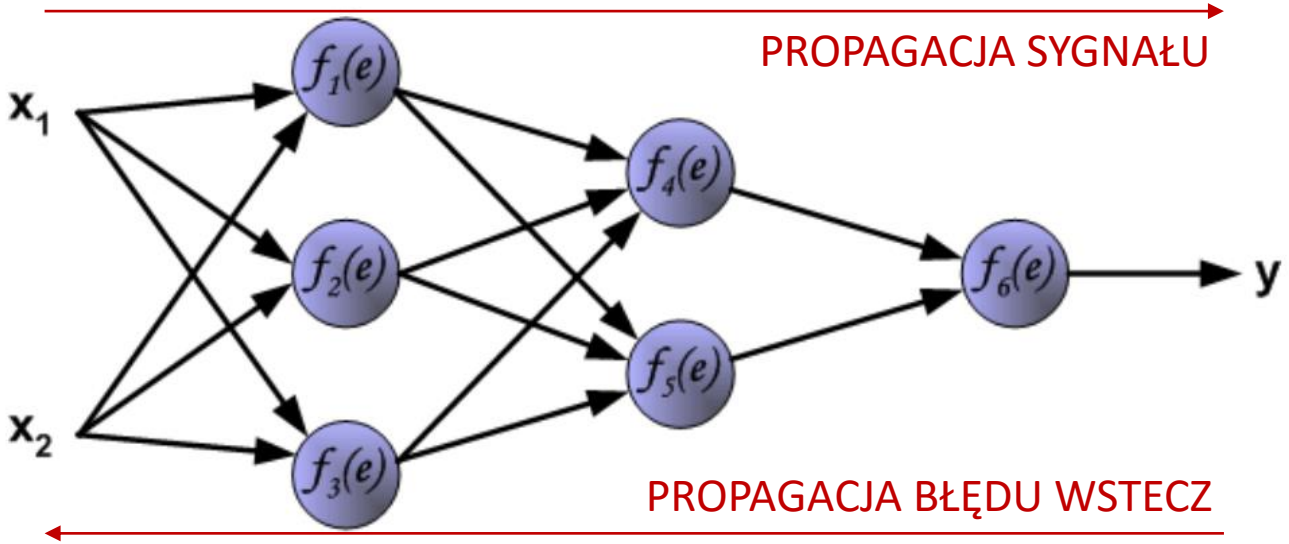
ALGORYTM WSTECZNEJ PROPAGACJI BŁĘDÓW

Pierwszym i zarazem najbardziej popularnym algorytmem do uczenia sieci wielowarstwowych jest algorytm wstecznej propagacji błędów (*backpropation algorithm*), którego działanie oparte jest na regule delta.

Wyznaczamy średniokwadratową funkcję błędu dla sieci Q(w), a następnie dążymy do znalezienia minimum tej funkcji względem wektora w.

Uczenie składa się z dwóch naprzemiennych faz:

1. **Fazy propagacji sygnału od wejść (wektora x) do wyjścia.**
2. **Fazy wstecznej propagacji błędu od wyjścia y w kierunku wejść sieci.**

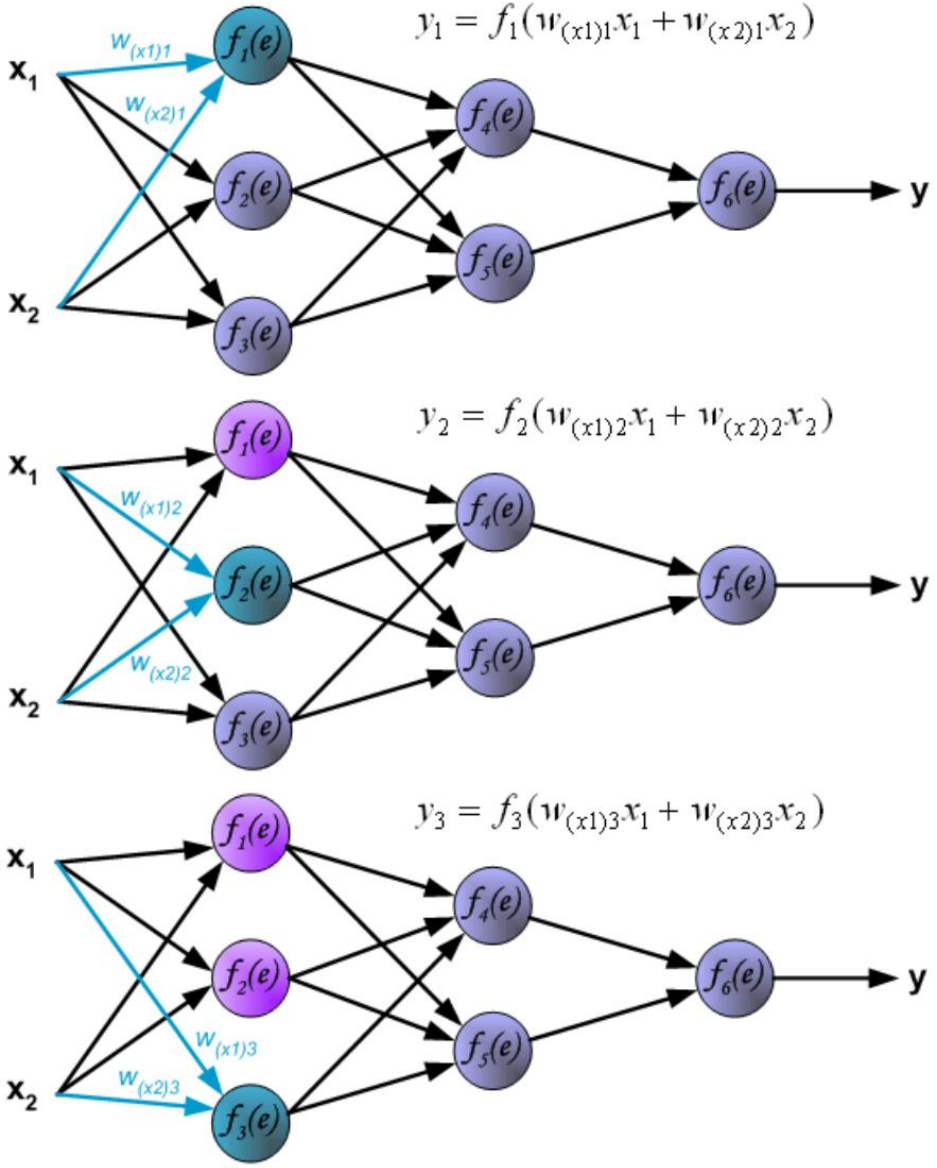


PROPAGACJA SYGNAŁU PRZEZ PIERWSZĄ WARSTWĘ SIECI

Sieć wielowarstwową pobudzamy sygnałami wejściowymi (x_1, x_2) kolejno warstwami neuronów.

Najpierw pobudzane są neurony w 1. warstwie i obliczana jest ich wartość wejściowa y_1, y_2 i y_3.

Wartości wyjściowe następnie są wykorzystane jako sygnały wejściowe w kolejnej warstwie neuronów.

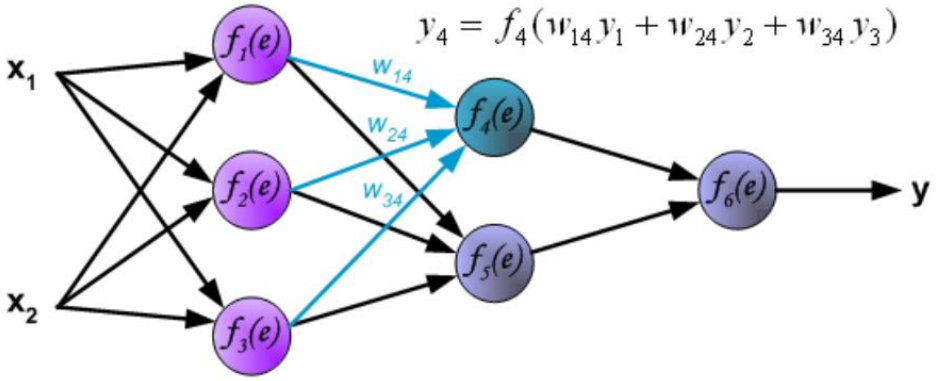


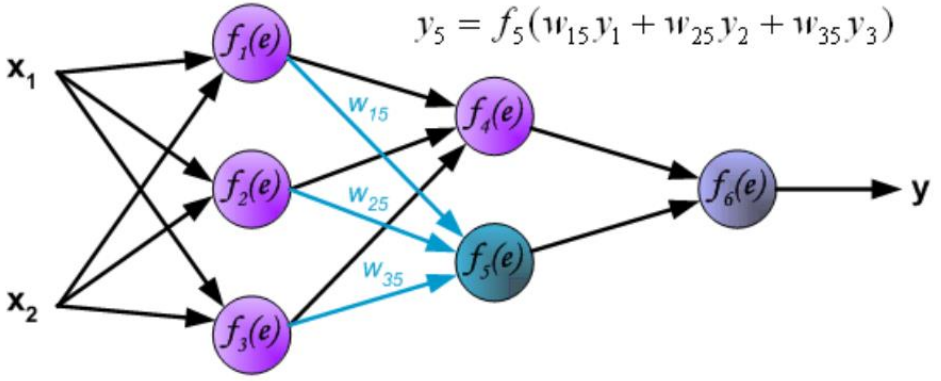
PROPAGACJA SYGNAŁU PRZEZ DRUGĄ WARSTWĘ SIECI

Drugą (ukrytą) warstwę neuronów sieci wielowarstwowej pobudzamy sygnałami wyjściowymi warstwy pierwszej (y_1, y_2 i y_3) obliczonymi w poprzednim kroku.

Na tej podstawie wyznaczane są wartości wyjściowe neuronów w warstwie drugiej (y_4 i y_5).

Obliczone wartości wyjściowe następnie są wykorzystane jako sygnały wejściowe w ostatniej warstwie neuronów, tzw. Warstwie wyjściowej sieci, która w naszym przypadku zawiera tylko 1 neuron.

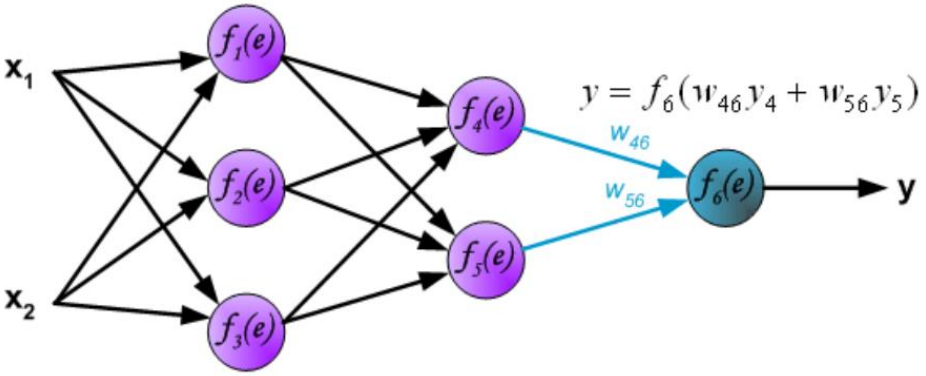




PROPAGACJA SYGNAŁU PRZEZ TRZECIĄ WARSTWĘ SIECI

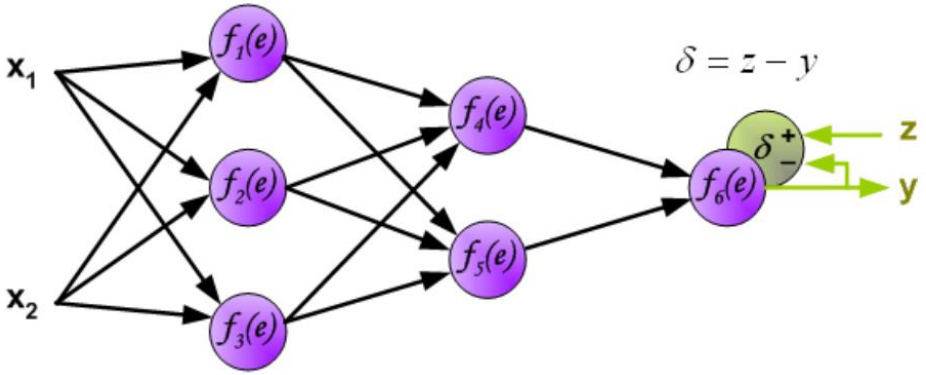
Trzecia warstwa neuronów sieci wielowarstwowej zawierająca pojedynczy neuron pobudzamy sygnałami wyjściowymi warstwy drugiej (y_4 i y_5) obliczonymi w poprzednim kroku.

Na tej podstawie wyznaczana jest wartość wyjściowa neuronu w warstwie trzeciej (y_6).



Obliczona wartość wyjściowa jest porównywana z wartością pożądaną (z) wyznaczoną w zbiorze uczącym przez nauczyciela, a różnica pomiędzy wartością pożądaną i uzyskaną ($\delta = z - y$) stanowią o dalszych krokach działania algorytmu.

Wartość błędu jest propagowana wstecz, ważona zgodnie z wagą połączenia między neuronami i sumowana w tych neuronach celem wyznaczenia ich błędu.



PROPAGACJA WSTECZNA BŁĘDU DO DRUGIEJ WARSTWY

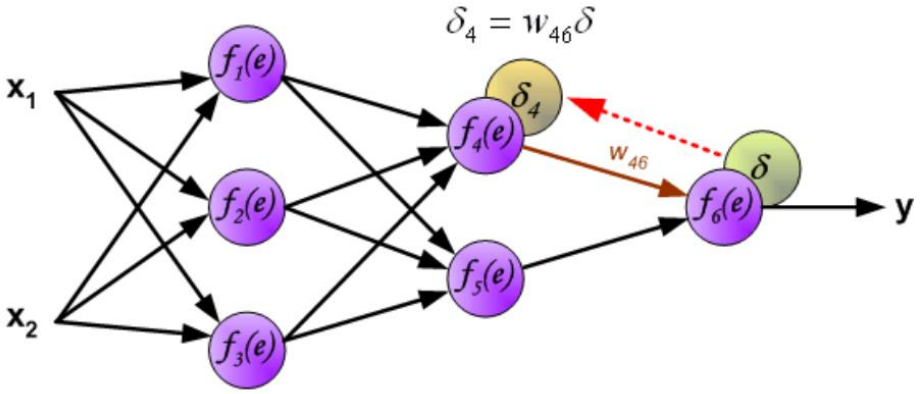
Przechodząc z błędem wstecz z trzeciej warstwy do drugiej, błąd jest ważony zgodnie z aktualną wartością wagi połączenia pomiędzy neuronem warstwy 3 i odpowiednim neuronem warstwy drugiej.

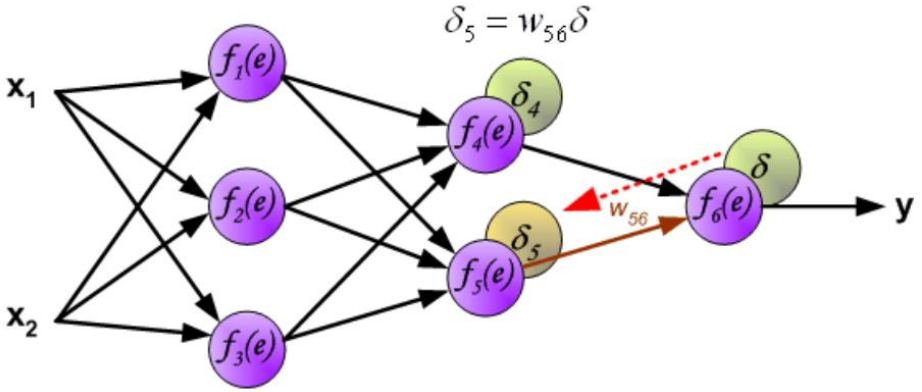
Neurony w warstwie drugiej sumują ważone sygnały błędów dochodzących do nich z warstwy trzeciej. Tutaj ze względu na to, iż w warstwie 3 występuje tylko jeden neuron, sumy składają się tylko z jednego członu:

$$\delta_i = \sum_{j=1}^{k} w_{ij}\delta_j$$

$$\delta_4 = w_{46}\delta_6$$

$$\delta_5 = w_{56}\delta_6$$



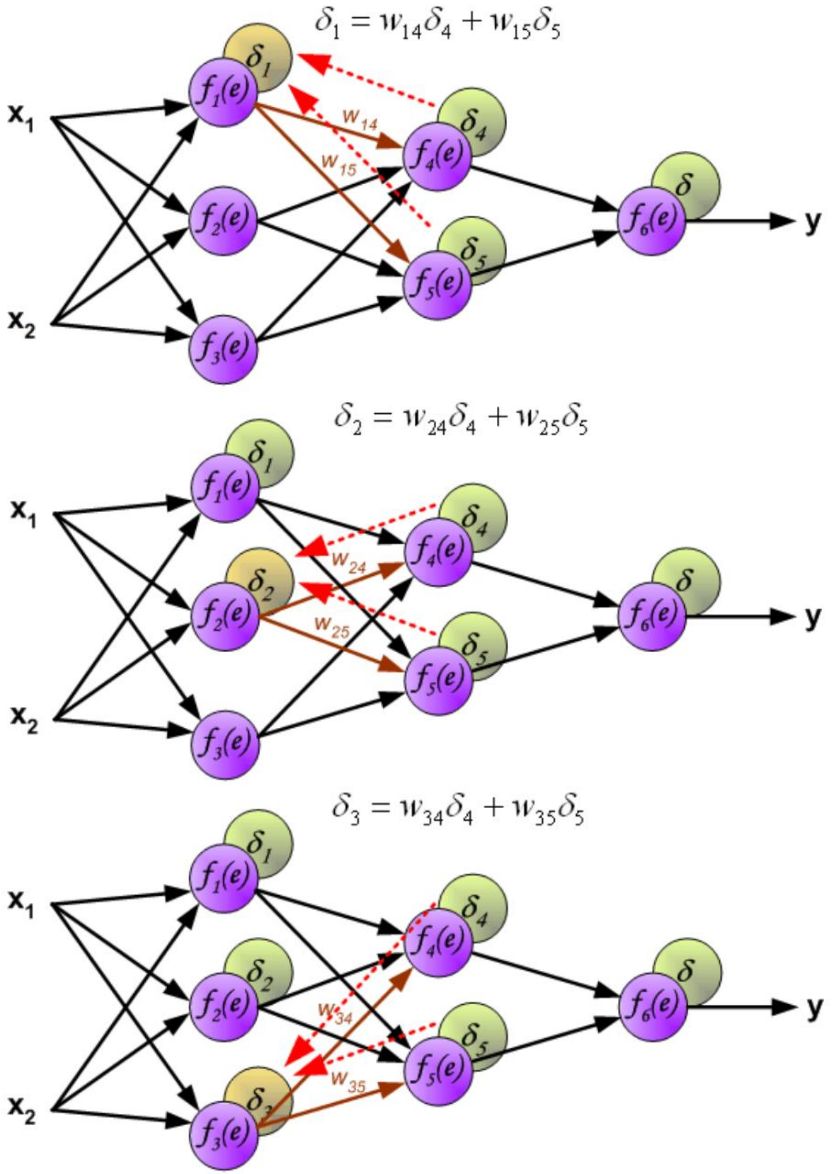


PROPAGACJA WSTECZNA BŁĘDU DO PIERSZWEJ WARSTWY

Przechodząc z błędem wstecz z drugiej do pierwszej warstwy sieci błędy obliczone dla drugiej warstwy (δ_4 i δ_5) są ważone zgodnie z aktualną wartością wag połączeń pomiędzy neuronami warstwy drugiej i pierwszej, a następnie sumowane neuronach warstwy pierwszej zgodnie z następującą zależnością:

$$\delta_i = \sum_{j=1}^{k} w_{ij}\delta_j$$

Następnie dokonywana jest korekta wartości wag sieci.



KOREKTA WAG SIECI W TRAKCIE PROPAGACJI WSTECZNEJ BŁĘDU

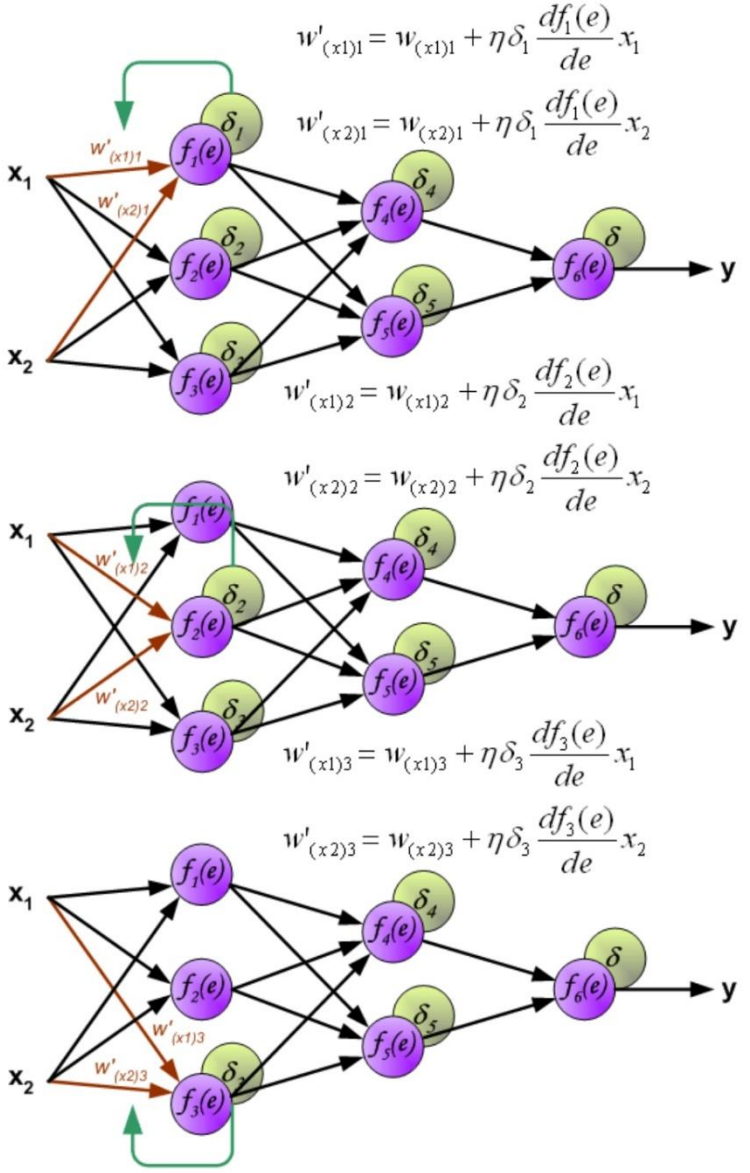
Dokonujemy korekty wag sieci tak, żeby zmniejszyły błąd średniokwadratowy, jaki był obliczony na wyjściu sieci.

W tym celu korzystamy z uogólnionej reguły delta, korzystając z pochodnej cząstkowej funkcji aktywacji oznaczonej jako $df_i(e) / de$.

Korekty wag możemy dokonywać:

od razu dla każdego wzorca uczącego (tzw. *on-line training*)

dopiero po zakończeniu propagacji błędów dla całego zbioru uczącego, sumując je dla wszystkich wzorców, a na końcu obliczając ich średnią (tzw. *off-line training* lub *batch training*).



KOREKTA WAG SIECI W TRAKCIE PROPAGACJI WSTECZNEJ BŁĘDU

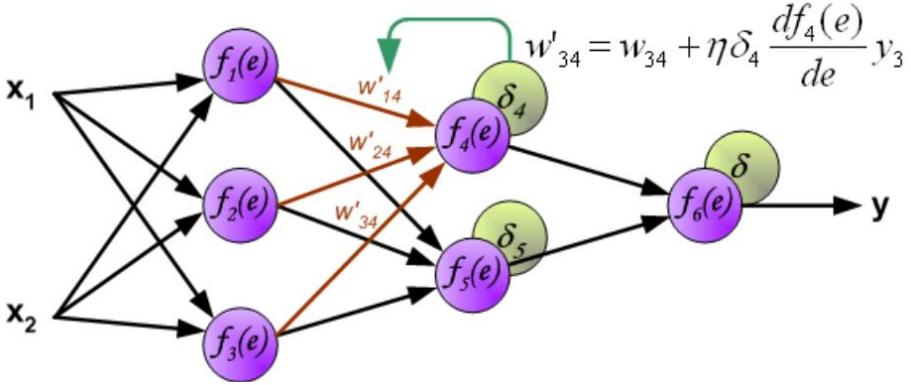
Podobnie dokonujemy korekty wag w drugiej i trzeciej warstwie sieci.

Współczynnik η służy stopniowej adaptacji sieci oraz określa szybkość uczenia się. Na początku jest zwykle duży, a następnie jego wartość jest stopniowo zmniejszana. Warunkuje on możliwość przejścia od minimów lokalnych do minimum globalnego.

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

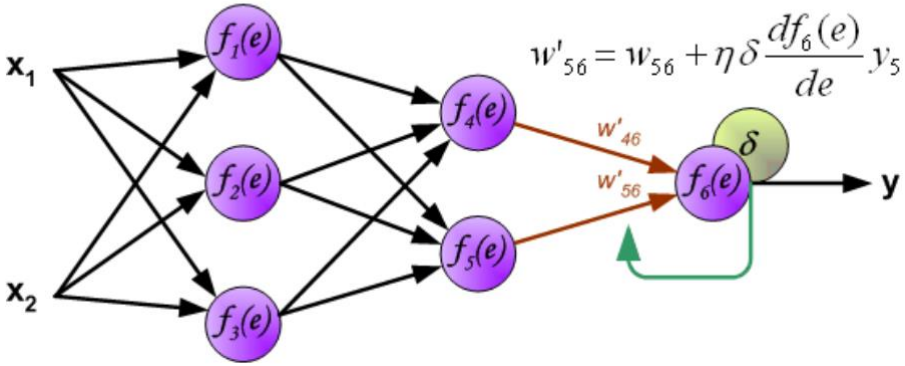
$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



$$w'_{46} = w_{46} + \eta\delta \frac{df_6(e)}{de} y_4$$

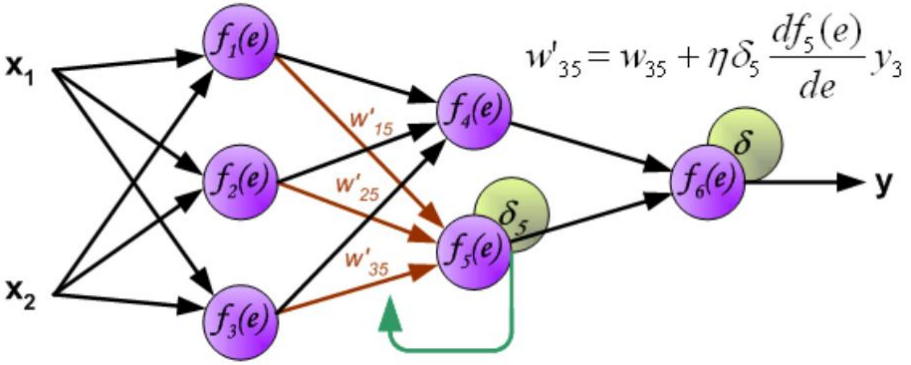
$$w'_{56} = w_{56} + \eta\delta \frac{df_6(e)}{de} y_5$$



$$w'_{15} = w_{15} + \eta\delta_5 \frac{df_5(e)}{de} y_1$$

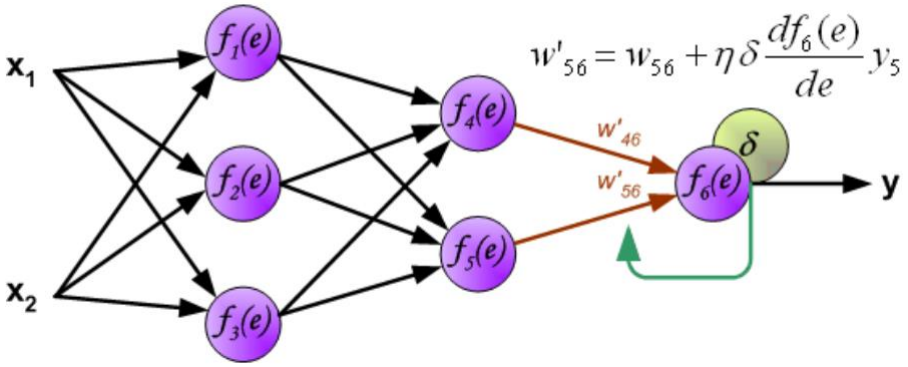
$$w'_{25} = w_{25} + \eta\delta_5 \frac{df_5(e)}{de} y_2$$

$$w'_{35} = w_{35} + \eta\delta_5 \frac{df_5(e)}{de} y_3$$



KOREKTA WAG SIECI W TRAKCIE PROPAGACJI WSTECZNEJ BŁĘDU

Wyznaczenie wartości pochodnej we wzorach na aktualizację wag zależne jest od postaci funkcji aktywacji f.



Dla najczęściej stosowanych funkcji:

sigmoidalnej:

$$f(x) = \frac{1}{1 + e^{-\beta * x}}$$

$$f'(x) = \beta * f(x) * \left(1 - f(x)\right) = \beta * y * (1 - y)$$

tangensa hiperbolicznego:

$$f(x) = tgh(\beta * x)$$

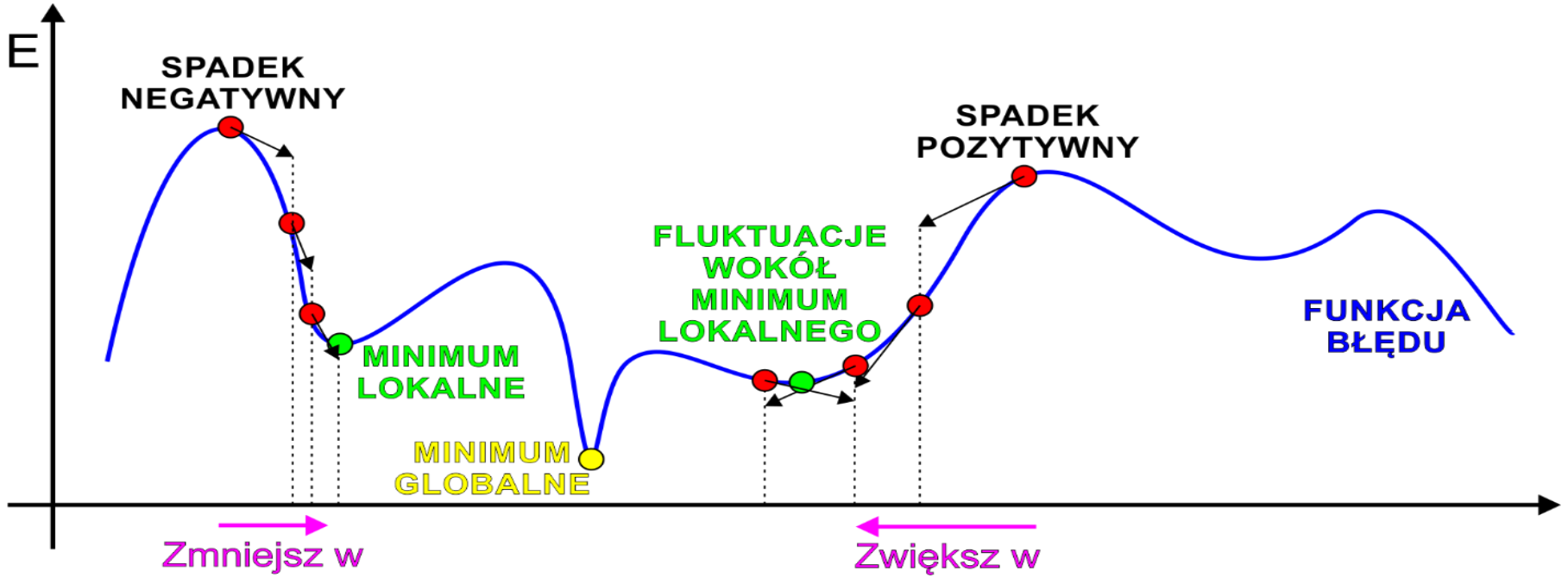
$$f'(x) = \beta * \left(1 - tgh^2(\beta * x)\right) = \beta * y\left(1 - y^2\right)$$

TRUDNOŚCI ZWIĄZANE Z ADAPTACJĄ SIECI NEURONÓW

Najczęściej spotykane metody adaptacji sieci neuronowych wykorzystują metody optymalizacji gradientowej, wyznaczając kierunek spadku gradientu funkcji błędu.

Metody te jednak są narażone na utknięcie w lokalnych minimach, co jest podstawową piętą Achillesa tych metod, nie dając gwarancji znalezienia globalnego minimum – a więc optymalnego rozwiązania z punktu widzenia określonej architektury sieci neuronowej oraz przyjętych funkcji aktywacji neuronów.

Dlatego właśnie metody uczenia głębokiego, które budują i dopasowują strukturę sieci hierarchicznie oraz stopniowo w oparciu o utworzone wcześniej reprezentacje cech dominujących wzorców uczących dają lepsze wyniki oraz lepszą jakość uogólniania.



BIBLIOGRAFIA

Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, MIT Press book, 2016 - http://www.deeplearningbook.org/ :