Raport Badawczy

Research Report

RB/07/2001

Sieci neuronowe

Jakub Gutenbaum

Instytut Badań Systemowych
Polska Akademia Nauk

Systems Research Institute
Polish Academy of Sciences

POLSKA AKADEMIA NAUK

Instytut Badań Systemowych

ul. Newelska 6

01-447 Warszawa

tel.: (+48) (22) 8373578

fax: (+48) (22) 8372772

Pracę zgłosił: doc. dr hab. inż. Michał Inkielman

Warszawa 2001

INSTYTUT BADAŃ SYSTEMOWYCH PAN

MiSwSE/07/2001

Jakub Gutenbaum

Sieci neuronowe

Pracownia Modelowania i Sterowania w Systemach Ekonomicznych

Kierownik Pracowni: Doc. dr hab. inż. Michał Inkielman

Nazwa zadania: Budowa scenariuszy dla ekonomicznych modeli prognostycznych

Nazwa ewentualnego podzadania:

Kierownik zadania: Doc. dr hab. inż. Michał Inkielman

Wykonawcy (zespół):

Warszawa, grudzień 2001 r

1 Wprowadzenie

Sieciami neuronowymi będziemy nazywali zalgorytmizowane modele matematyczne imitujące niektóre aspekty działania sieci nerwowych w organizmach żywych.

Sieci neuronowe (sztuczne sieci neuronowe, sieci neuronalne, sieci neuropodobne) powstawały z początkowym zamiarem modelowania elementów biologicznych systemów nerwowych. Początki tak ukierunkowanych badań sięgają lat czterdziestych XX wieku [McCulloch, Pitts, 1943]. Jednakże nie zanotowano tu znaczących osiągnięć, wydaje się, że głównie ze względu na nieporównywalność tworzywa z jakim mamy do czynienia w bioorganizmach i w układach technicznych. Sytuacja taka ulega jednak poprawie, szczególnie od czasu wykorzystania w modelowaniu obliczeń równoległych, dzięki którym następuje zwielokrotnienie zdolności operacyjnych komputerów. Duże nadzieje są obecnie wiązane ze stosowaniem pamięci biologicznej w XXI – wiecznych generacjach komputerów.

Dużo większe sukcesy osiągnięto natomiast w pokrewnej dziedzinie, a mianowicie w zastosowaniu, podpatrzonych u natury, zasad działania bioorganizmów - do budowy algorytmów, rozwiązujących wiele różnorodnych zadań praktycznych, związanych z szeroko pojętym podejmowaniem decyzji. Ten kierunek, nazywany nieco na wyrost *sztuczną inteligencją*, ma do odnotowania wiele istotnych naukowych i praktycznych osiągnięć. Ten też kierunek będzie dalej omawiany.

Mówienie, w kontekście stosowania sieci neuronowych, o sztucznej *inteligencji*, jest uzasadnione tym, że odpowiednio wykorzystane sieci tego typu są w stanie gromadzić i selekcjonować dane, uczyć się, czyli wyciągać logiczne wnioski ze zgromadzonej

teoretycznej i doświadczalnej wiedzy oraz podpowiadać nie trywialne rozwiązania w zakresie podejmowania decyzji. W określonych przypadkach sieć neuronowa wykazuje przewagę nad działaniem mózgu, wynikającą z umiejętności szybkiego przeanalizowania olbrzymiej liczby możliwych wariantów i wskazania wariantu optymalnego. W innych – najmniej sprawny mózg działa zdecydowanie lepiej, niż nawet najbardziej wyrafinowany algorytm komputerowy. Potrafi bowiem odrzucać całe podzbiory rozwiązań (decyzji) nieprzydatnych, na podstawie intuicji, bez konieczności ich szczegółowego rozpatrywania.

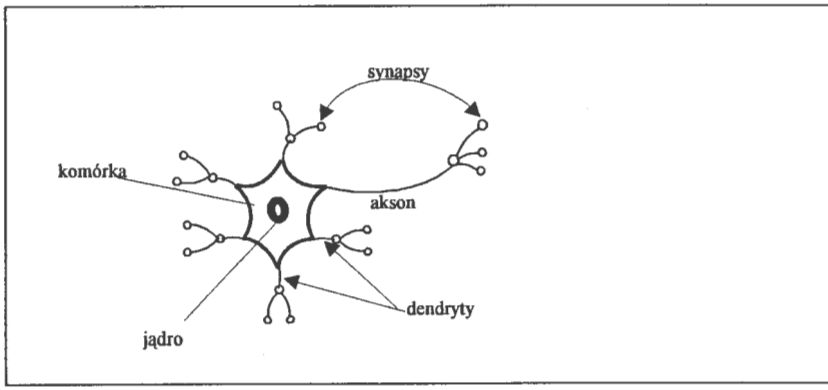
Teoria i zastosowania sieci neuronowych znajdują się w fazie intensywnego rozwoju i obejmują bardzo szeroki zakres zastosowań praktycznych w medycynie, ekonomii, wojskowości, zarządzaniu. Pobudziły też rozwój specyficznych algorytmów optymalizacyjnych, opartych na zasadach doboru naturalnego właściwego genetyce [Davidor, 1991].

Sieci neuronowe wykazują szczególną przydatność przy rozwiązywaniu zadań rozpoznawania i klasyfikacji obiektów, jak np. automatyczne czytanie tekstów (klasyfikacja znaków alfanumerycznych), identyfikacja osób na podstawie linii papilarnych, czy też sygnału mowy, diagnostyka medyczna, rozpoznawanie sytuacji awaryjnych.

Podstawowymi elementami układu nerwowego są komórki nerwowe, czyli **neurony**. Każdy biologiczny neuron składa się (rys. 1) z komórki właściwej oraz dwóch rodzajów wypustek - licznych wypustek wejściowych (dendrytów) oraz pojedynczej - wyjściowej (aksonu). Na końcu wypustek znajdują się synapsy, stanowiące złącza nerwowe, za których pomocą realizowane są sprzężenia między neuronami. Tworzą one skomplikowaną sieć wzajemnych powiązań, wzdłuż których przenoszone są impulsy

nerwowe o naturze chemiczno - elektrycznej. Ocenia się, że mózg człowieka zawiera astronomiczną liczbę około 10^{11} neuronów, przy liczbie połączeń rzędu 10^{15}.

Sieć neuronowa przetwarza sygnał zawierający informacje o obiekcie (o stanie otoczenia, o warunkach zadania decyzyjnego) w sygnał stanowiący impuls do podjęcia określonej decyzji.



Rys.1 Biologiczny neuron

2 Model neuronu

W komórkach nerwowych dokonywana jest operacja sumowania impulsów wejściowych z odpowiednimi wagami. W synapsie, stanowiącej rodzaj przekaźnika, następuje dalsze przekształcanie otrzymanego sygnału, jego osłabienie lub wzmocnienie. Zachodzi też odpowiednie oddziaływanie na neurony sąsiednie. Taka zasada działania stanowi podstawę budowy sztucznego neuronu, który składa się z sumatora sygnałów oraz funkcji aktywacji (rys.2). Jego podstawową częścią jest **sumator sygnałów wejściowych** (1, u_1, u_2,..., u_I) z wagami w_0, w_1, w_2..., w_I, o sygnale wyjściowym x:

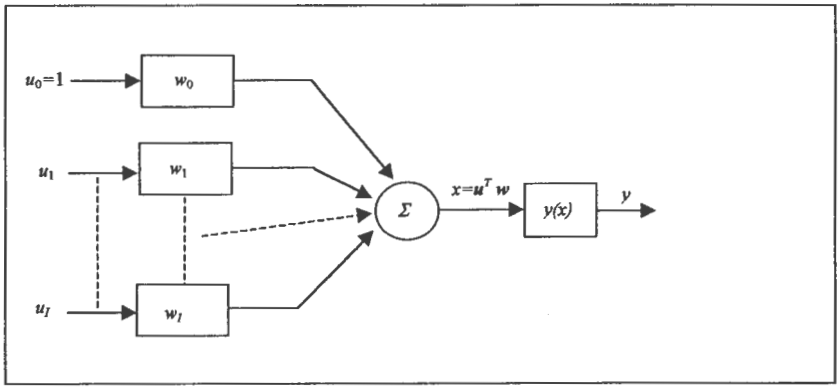
$$x = w_0 + w_1 u_1 + w_2 u_2 + ... + w_I u_I = w_0 + \sum_{i=1}^{I} w_i u_i \qquad (1)$$

lub w zapisie wektorowym:

$$x = u^T w \tag{2}$$

gdzie:

$$w^T = [w_0, w_1, w_2, \ldots, w_I] \;;\; u^T = [1, u_1, u_2, \ldots, u_I] \tag{3}$$



Rys. 2 Model sztucznego neuronu

Modelem synapsy jest tzw. **funkcja aktywacji** – nieliniowa funkcja argumentu x (rys. 3):

$$y(x) = f(u^T w) \tag{4}$$

Cechy neuronu zależą w dużym stopniu od funkcji aktywacji. Najczęściej stosowanymi funkcjami są:

- skok jednostkowy unipolarny (rys. 3a):

$$y(x) = \begin{cases} 1 \text{ przy } x \geq 0 \\ 0 \text{ przy } x < 0 \end{cases} \tag{5}$$

- skok jednostkowy bipolarny (rys. 3b):

$$y(x)=\begin{cases} 1 \text{ przy } x \geq 0 \\ -1 \text{ przy } x < 0 \end{cases} \qquad (6)$$

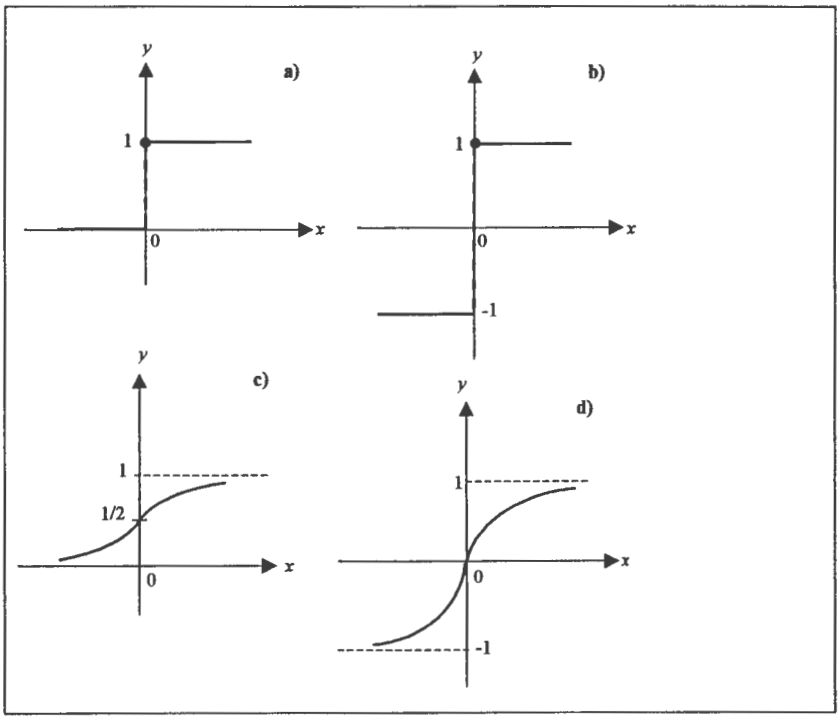
- funkcja logistyczna (rys. 3c):

$$y(x)=\frac{1}{1+\exp(-\beta x)} \qquad (7)$$

- tangens hiperboliczny (rys. 3d):

$$y(x) = \text{tgh}(\beta x) \qquad (8)$$

Skoki jednostkowe - unipolarny (*5*) oraz bipolarny (*6*) - są funkcjami nieciągłymi, co sprawia pewne kłopoty przy doborze właściwych wag. Ich odpowiednikami są funkcje ciągłe – funkcja logistyczna (*7*) i tangens hiperboliczny (*8*). Stromość tych funkcji można zmieniać przez wybór współczynnika β: im większe β, tym bardziej funkcja logistyczna zbliża się do skoku jednostkowego, a tangens hiperboliczny do skoku bipolarnego.



Rys. 3 Funkcje aktywacji

3 Uczenie pojedynczego neuronu

Przy doprowadzeniu do wejścia neuronu sygnału $\boldsymbol{u}$ na jego wyjściu pojawi się sygnał $y(\boldsymbol{u})$:

$$y(\boldsymbol{u}) = f(\boldsymbol{u}^T \boldsymbol{w}) \,;\; f: R^{I+1} \rightarrow R^1 \qquad (9)$$

Proces uczenia (trenowania) neuronu polega na takim doborze wag neuronu $\boldsymbol{w}(t)$, drogą kolejnych iteracji ($t = 0,1,...$), aby na wyjściu uzyskać sygnał $y^*(\boldsymbol{u})$, stanowiący wzorzec klasy, do której należy $\boldsymbol{u}$:

$$\lim_{t \to \infty} f[\boldsymbol{u}^T \boldsymbol{w}(t)] = y^*(\boldsymbol{u}) \qquad (10)$$

Funkcja $y^*(\boldsymbol{u})$ nie musi być zadana w postaci jawnej. Zakładamy jedynie, że dla każdego wektora $\boldsymbol{u}$, który pojawi się na wejściu, znana jest pożądana dla tego $\boldsymbol{u}$ wartość sygnału wyjściowego $y^*(\boldsymbol{u})$. Sposób uczenia zakładający znajomość $y^*(\boldsymbol{u})$ w procesie uczenia nazywany jest uczeniem z nauczycielem (uczeniem nadzorowanym). O innych sposobach uczenia będzie mowa w X.5.

Liniowy model neuronu z algorytmem realizującym iteracyjny proces uczenia nazywa się Adaline (Adaptive Linear Element). Zawiera on, poza sumatorem (1), detektor błędu $\delta(t)$:

$$\delta(t) = y^*(\boldsymbol{u}) - [\boldsymbol{u}^T \boldsymbol{w}(t)] \qquad (11)$$

oraz algorytm nastrajania wag:

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + \eta(t)\,\delta(t)\,\boldsymbol{u} \qquad (12)$$

gdzie $\eta(t)$ – stała, lub malejąca z wzrostem t, długość kroku.

Zapoczątkowanie procedury nastrajania wag wymaga arbitralnego określenia warunku początkowego $\boldsymbol{w}(0)$. Często stosowana jest metoda doboru losowego.

Można pokazać [Tadeusiewicz,1993], że algorytm (*12*) jest równoważny iteracyjnemu algorytmowi gradientowemu:

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) - \eta(t)\,[\partial q(\boldsymbol{w})\partial \boldsymbol{w}]^T \qquad (13)$$

gdzie $q(\boldsymbol{w})$ jest funkcją błędu o postaci:

$$q[w(t)] = \tfrac{1}{2}\sum_{n=1}^{N}[y^*(u) - y(t)]^2 \qquad (14)$$

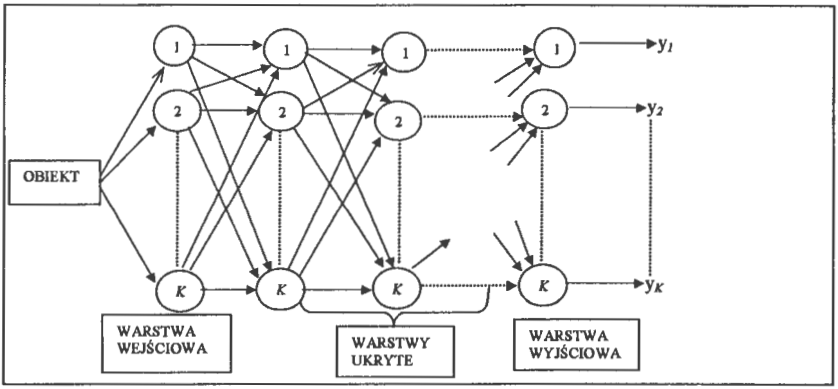
$$y(t) = u^T w(t) \qquad (15)$$

Ponieważ funkcja $q(w)$ jest ściśle wypukła, ma jedno dobrze określone minimum, to proces iteracyjny $w(t)$ jest zbieżny do wektora wag w^*, które zapewniają minimum funkcji celu (*14*).

4 Struktury sieci

Połączone w określoną strukturę sztuczne neurony tworzą sieć neuronową, zazwyczaj budowaną warstwowo. Warstwa wejściowa składa się z neuronów, do których dociera sygnał z zewnątrz sieci. Warstwa wyjściowa składa się z neuronów, które „wyprowadzają" sygnał poza sieć. Warstwy pośrednie, jeśli istnieją, są nazywane warstwami ukrytymi, bowiem do wejść i wyjść tworzących je neuronów nie ma bezpośredniego dostępu.

Struktura połączeń międzywarstwowych może być bardzo różna. W **sieci jednokierunkowej** sygnały przepływają od wejścia do wyjścia (rys. 4).



Rys. 4 Sieć jednokierunkowa

Operację realizowaną przez sieć jednokierunkową z jedną warstwą ukrytą można przedstawić następująco.

Niech górny indeks m dotyczy neuronów w warstwie m-tej, przy czym m= 0, 1, 2 – dotyczy warstw: wejściowej, ukrytej i wyjściowej - odpowiednio. Załóżymy ponadto, że każda warstwa zawiera tę samą liczbę K neuronów, z których każdy ma tę samą liczbę wejść równą I+1.

Pojawienie się sygnału $\boldsymbol{u} = [1, u_1, ..., u_I]^T$ na wejściu sieci spowoduje, że na wejściu k-ego neuronu warstwy ukrytej powstanie sygnał y_k^1 równy:

$$y_k^1 = f_k^0 [(\boldsymbol{u})^T \boldsymbol{w}_k^0] \quad ; \quad k = 1, 2, ..., K \qquad (16)$$

gdzie f_k^0 – funkcja aktywacji k-ego neuronu warstwy wejściowej, $\boldsymbol{w}_k^0$ – wektor wag k-ego neuronu warstwy wejściowej.

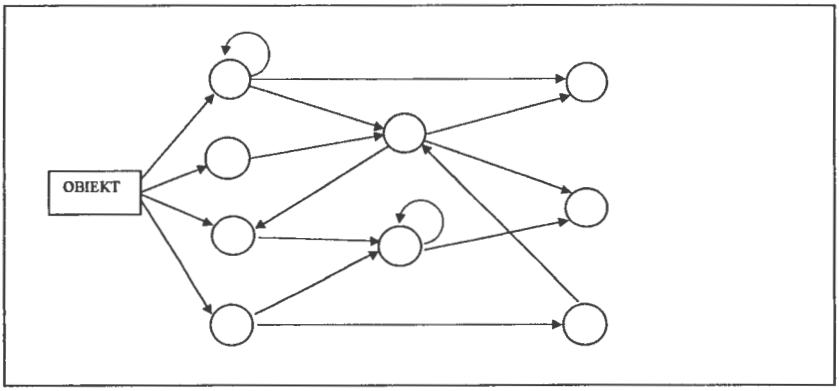
Wektor $\boldsymbol{y}^1 = \boldsymbol{u}^1$ na wejściu każdego z neuronów warstwy ukrytej spowoduje pojawienie się na wyjściu k-ego neuronu warstwy ukrytej sygnału y_k^2 równego:

$$y_k^2 = f_k^1 [(\boldsymbol{u}^1)^T \boldsymbol{w}_k^1] \; : \; k = 1, 2, ..., K \qquad (17)$$

gdzie f_k^1 – funkcja aktywacji k-ego neuronu warstwy ukrytej, w_k^1 – wektor wag k-ego neuronu warstwy ukrytej. Na wyjściu sieci pojawi się sygnał:

$$y_k^3 = f_k^2[(u^2)^T w_k^2] \qquad (18)$$

Często spotykaną jednokierunkową siecią neuronową jest tzw. **perceptron** [Rosenblatt, 1962] , który ma pełną strukturę oraz jednakowe funkcje aktywacji poszczególnych neuronów (sieć homogeniczna).



Rys. 5 Sieć rekurencyjna

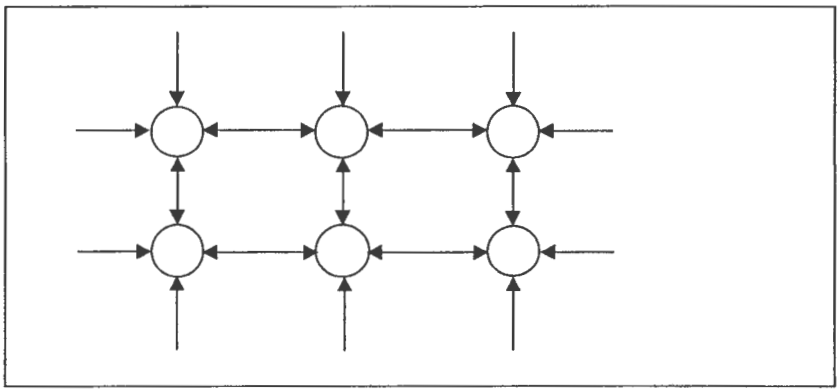
W sieciach rekurencyjnych (rys. 5) możliwy jest również wsteczny kierunek przepływu sygnałów, w tym sprzężenia zwrotne – lokalne lub obejmujące całą sieć. W tym ostatnim przypadku sygnały z warstwy wyjściowej podawane są z powrotem do warstwy wejściowej. Sieć taka może być autonomiczna lub pobudzana wymuszeniem zewnętrznym. Najprostsza sieć rekurencyjna, tzw. sieć Hopfielda, składa się z jednej warstwy, objętej sprzężeniem **zwrotnym**. W dyskretnej sieci Hopfielda składającej się z K neuronów jednowejściowych, równanie różnicowe opisujące dynamikę k-ego neuronu ma postać:

$$y_k(t+1) = \sum_{j=1}^{K} w_j{}^k f^j[y_j(t)] + I_k(t)\ ;\ k = 1, 2, ..., K \qquad (19)$$

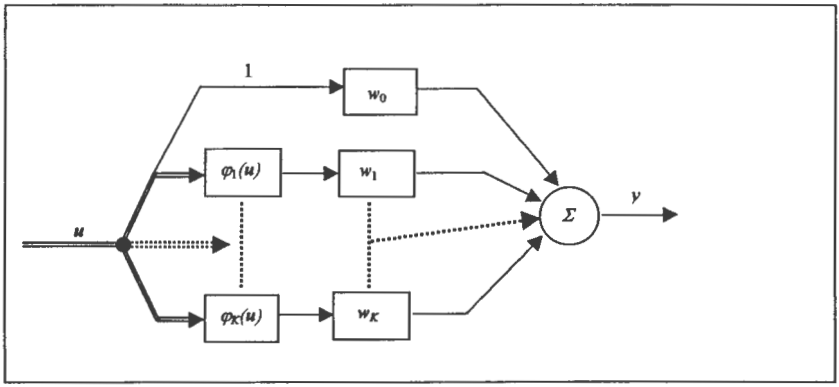
gdzie: $I_k(t)$ – wymuszenie zewnętrzne doprowadzone do k-tego neuronu.

Sieci Hopfielda mogą być stosowane do rozwiązywania zadań optymalizacyjnych [Mandziuk, 2000].

Połączenia między neuronami sąsiednich warstw mogą być „każdy z każdym" lub mieć charakter selektywny. Jeśli w sieci występują również połączenia między sąsiednimi neuronami tej samej warstwy - mówimy o **sieci komórkowej** (rys. 6). W zadaniach aproksymacji stosowane są **sieci radialne** (rys. 7), w których zakłócony sygnał wejściowy, generowany przez aproksymowaną funkcję o nieznanej postaci, podawany jest do wszystkich neuronów warstwy wejściowej, a sygnał wyjściowy jest ważoną sumą aproksymujących funkcji. W charakterze funkcji aproksymujących stosowane są funkcje nieliniowe, symetryczne względem środka, tzw. funkcje radialne.



Rys. 6 Sieć komórkowa



Rys. 7 Sieć radialna

W praktyce stosuje się zazwyczaj sieci bez warstw ukrytych lub o niewielkiej ich liczbie, ze względu na trudności związane z nastrajaniem wag i wyborem struktury sprzężeń w procesie uczenia przy większej liczbie warstw.

5 Zastosowania sieci neuronowych

5.1 Zastosowania sieci liniowej do klasyfikacji obiektów

W przypadku sieci neuronowej jednowarstwowej o K neuronach I+1 wejściowych, na wyjściu sumatorów otrzymujemy wektor $x = [x^1, x^2, ..., x^K]^T$:

$$x = W u \tag{20}$$

gdzie: W - macierz wag sieci:

$$W = \begin{bmatrix} w_0^1 & w_1^1 & w_2^1 & \cdots & \cdots & \cdots & w_I^1 \\ w_0^2 & w_1^2 & w_2^2 & \cdots & \cdots & \cdots & w_I^2 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ w_0^K & w_1^K & w_2^K & \cdots & \cdots & \cdots & w_I^K \end{bmatrix} \tag{21}$$

Sieć o K neuronach w warstwie wyjściowej realizuje odwzorowanie I- wymiarowej (formalnie I+1-wymiarowej) przestrzeni cech obiektu w K-wymiarową przestrzeń klas. Taka sieć może służyć do rozwiązywania różnorodnych zagadnień decyzyjnych: zadania klasyfikacji i identyfikacji obiektów, filtracji, predykcji, podejmowania decyzji optymalnych.

Najczęściej spotykane zastosowania liniowych sieci neuronowych dotyczą zadań klasyfikacji obiektów, np. rozpoznawania znaków alfanumerycznych. Stosowany do tego celu układ wymaga odpowiednio wyposażonej kamery telewizyjnej oraz co najmniej jednowarstwowej sieci o liczbie neuronów równej liczbie rozróżnianych klas.

Przy pojawieniu się rozpoznawanego obiektu przed kamerą powstaje obraz, przetwarzany w wektor liczbowy (wektor cech), w szczególności zero-jedynkowy, charakterystyczny dla tego właśnie obiektu.

W większości zastosowań sieci neuronowych, w tym przy opisywanym tu rozpoznawaniu obiektów, można rozróżnić dwie fazy działania sieci: fazę uczenia (treningu) i fazę pracy właściwej, w tym przypadku - decyzję o przynależności do określonej klasy, już bez informacji pochodzącej od nauczyciela.

W fazie uczenia (z nauczycielem) pokazuje się kamerze pojedyncze obiekty. Każdy obiekt generuje właściwy mu wektor cech $\boldsymbol{u} = [1, u_1, u_2 ..., u_I]^T$, stanowiący sygnał wejściowy sieci. Jednocześnie nauczyciel informuje układ decyzyjny o przynależności obiektu do określonej klasy. Uczenie polega na takim ustawieniu wag, aby w trakcie właściwej pracy sieć była w stanie, możliwie jak najlepiej, rozpoznawać klasę obiektu na podstawie wektora cech.

Załóżmy początkowo, że każdorazowe pojawienie się obiektu, należącego do klasy k generuje jednoznacznie określony wektor (wektor wzorcowy) u^{*k}, przy czym:

$$u^{*k} \neq u^{*j} \quad \text{przy } k \neq j, (k, j) = 1, 2, ..., K \tag{22}$$

Przy pojawieniu się obiektu klasy k sygnał o nim w postaci wektora u^{k*} doprowadzany jest do wejść wszystkich neuronów. Na wyjściu m – tego neuronu (m = 1,2,..., K) pojawi się sygnał x^k_m równy iloczynowi skalarnemu:

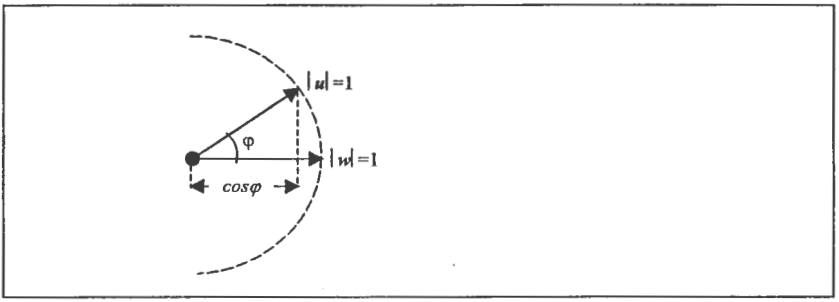
$$x^k_m = (u^{*k})^T w^m; \ (k, m) = 1, 2, ..., K \tag{23}$$

gdzie w^m – wektor wag m-tego neuronu:

$$w^m = [w_0^m, w_1^m, ..., w_I^m]^T$$

Przedstawiona dalej metoda rozpoznawania obiektów opiera się na tym, że iloczyn skalarny dwóch wektorów u oraz w jest równy iloczynowi ich długości (modułów) przemnożonemu przez cosinus kąta między nimi (rys. 8):

$$u^T w = |u||w| \cos \phi \tag{24}$$



Rys. 8 Iloczyn skalarny wektorów znormalizowanych

Iloczyn skalarny (*24*) przy znormalizowanych wektorach u oraz w :

$$|u| = 1 \ , |w| = 1 \qquad (25)$$

osiąga wartość maksymalną, równą jedności, przy zgodności kierunków obu wektorów ($\cos \phi = 0$):

$$u = w \qquad (26)$$

Jeśli zatem, zgodnie z założeniem, każdy obiekt należący do klasy k generuje zawsze ten sam wektor cech $u = u^{*k}$ (wektor wzorcowy), przy czym różniący się od wektora wzorcowego obiektów należących do innych klas, zgodnie z (*22*), to zadanie rozpoznawania jest banalne. Numerujemy neurony i klasy obiektów od 1 do K. Faza uczenia polega na tym, że każdemu k-temu neuronowi nadajemy wektor wag w^k równy wektorowi wzorcowemu k-tej klasy:

$$w^k = u^{*k}; \qquad k = 1, 2, ..., K \qquad (27)$$

W fazie rozpoznawania właściwego obiekt jest zaliczany do klasy odpowiadającej neuronowi, na którego wyjściu pojawi się sygnał **większy**, niż sygnały na wyjściach neuronów pozostałych. Wiemy też, że maksimum, które wystąpi przy całkowitej zgodności wektora cech obiektu z wektorem wzorcowym danej klasy, jest równe jedności. I tak obiekt należy do klasy k, jeśli przy pojawieniu się wektora cech u zachodzą nierówności:

$$u^T w^k > u^T w^m, \quad \text{dla każdego } m = 1, 2, ..., K; \quad m \neq k \qquad (28)$$

oraz:

$$u^T w^k = 1 \qquad (29)$$

gdzie u oraz w – wektory znormalizowane

Założenie, że każdy obiekt należący do klasy k generuje identyczny wektor u^{k*}, można przyjąć np. w przypadku automatycznego odczytywania pisma maszynowego. Jednakże przy odczytywaniu pisma ręcznego, poszczególne znaki należące do określonej klasy, nawet pisane przez tę samą osobę, różnią się między sobą, a więc generują różne wektory cech.

Człowiek, czytając pismo ręczne, nie kieruje się tylko kształtem liter. Przecież często nie można rozróżnić np. litery *a* od *o*, czy też *u* od *n*. Do ich rozróżnienia uwzględnia się kontekst, w którym litery te występują. Mózg bezwiednie dokonuje analizy lingwistycznej i semantycznej, a to wymaga już inteligencji. Sieci neuronowe mogą również być wyposażone w odpowiednie inteligentne algorytmy, dokonujące dogłębnej analizy odczytywanego tekstu w celu uniknięcia pomyłek.

Opracowano wiele metod rozpoznawaniu obiektów generujących wektory cech różniące się między sobą, nawet przy obiektach należących do tej samej klasy. Przedstawimy trzy takie metody.

Metoda wektorów wzorcowych. Etap uczenia polega na wyznaczeniu wektorów wzorcowych dla każdej z klas, a następnie - nastrojeniu wag na poziomie tych wektorów, zgodnie z (*27*). Stosuje się przy tym wielokrotne podawanie tego samego ciągu uczącego. Wybór tego ciągu jest odrębnym problemem. Powstaje bowiem sprzeczność. Jeśli rozrzut cech obiektów tej samej klasy, demonstrowanych podczas uczenia, jest zbyt duży, na wzorcowy wektor cech mogą mieć zbyt duży wpływ obiekty, których wektory cech różnią się znacząco od wzorcowego, ale które pojawiają się stosunkowo rzadko. Z drugiej strony, jeśli w fazie uczenia ograniczymy się do pokazywania obiektów danej klasy mało różniących się między sobą, to wektor wzorcowy nie

będzie reprezentatywny dla obiektów odbiegających od stereotypu, co też może być źródłem błędów.

Wektorem wzorcowym dla danej klasy może być wektor, który w sensie wybranej miary odległości, jest w średnim najbliższy wszystkim wektorom cech generowanych w fazie uczenia przez obiekty należące do tej samej klasy. I tak, jeśli przyjąć, że w trakcie uczenia klasa k była prezentowana N_k razy, to wektor wzorcowy dla tej klasy u^{k*} powinien zapewniać minimum uśrednionego wskaźnika odchylenia:

$$q(u^{k*}) = \min \frac{1}{N_k} \sum_{n=1}^{N_k} \left\| u^{kn} - u^{k*} \right\| \; ; \; k = 1, 2, ..., K \qquad (30)$$

gdzie: $||\,.\,||$ - norma wektora, u^{kn} - wektor cech obiektu należącego do k – tej klasy w n – tej prezentacji.

Wagi neuronów w^k nastawia się na poziomie wektora wzorcowego dla danej klasy, wynikającego z (*30*):

$$w^k = u^{k*}; \qquad k = 1, 2, ..., K \qquad (31)$$

Rozpoznawanie właściwe polega na zaliczaniu obiektu do klasy k, której odpowiada neuron o największej wartości sygnału wyjściowego.

Metoda przyrostów. Metoda, podobnie jak poprzednia, polega na takim iteracyjnym nastrajaniu wag, aby przy rozpoznawaniu klasa obiektu była określona neuronem, na którego wyjściu pojawi się maksymalny sygnał wyjściowy, ale bez konieczności uprzedniego określenia wektorów wzorcowych. W tej metodzie wektor wag m – tego neuronu w $(t+1)$-ej iteracji, przy pojawieniu się obiektu należącego do klasy k-tej, dany jest wzorem:

$$w^m(t+1) = w^m(t) + \eta\, \delta_m^k(t)\, x^k(t)\ ;\ m = 1, 2, ..., K \qquad (32)$$

gdzie: $x^k(t) = [x_1{}^k(t), x_2{}^k(t), ..., x_K{}^k(t)]^T$ – wektor sygnałów wyjściowych przy pojawieniu się obiektu należącego do klasy k:

$$x_m^k(t) = [w^m(t)]^T u^k \qquad (33)$$

Przedstawimy dalej jeden z możliwych sposobów iteracyjnego nastrajania współczynników δ_m^k.

Oznaczmy przez J^k podzbiór indeksów poszczególnych neuronów takich, że przy pojawieniu się obiektu klasy k zachodzi:

$$J^k = \{m \in \{1, 2, ..., K\}: x_m^k \geq x_k^k\ ,\ m \neq k\} \qquad (34)$$

Niech na wejściu sieci z ponumerowanymi dowolnie neuronami pojawi się obiekt należący do klasy k generujący sygnał u^k. Na wyjściu m-ego neuronu pojawi się wówczas sygnał x_m^k ; m= 1, 2, ..., K. Jeśli sygnał, na wyjściu k – tego neuronu jest większy od wszystkich innych sygnałów wyjściowych, to sieć właściwie rozpoznała obiekt i korekta wag nie następuje:

$$\text{przy } J^k = \varnothing \quad \delta_m^k(t) = 0\ ;\ m = 1, 2, ..., K \qquad (35)$$

Jeśli natomiast przy pojawieniu się obiektu należącego do klasy k znajdą się neurony, których sygnał wyjściowy x_m^k jest nie mniejszy (większy lub równy) od sygnału wyjściowego neuronu k-tego, należy odpowiednio skorygować wagi. Polega to na zmniejszeniu wag neuronów o sygnale wyjściowym nie mniejszym, niż sygnał wyjściowy neuronu k-tego oraz zwiększeniu wag neuronu k-tego. Zatem:

przy $J^k \neq \varnothing$

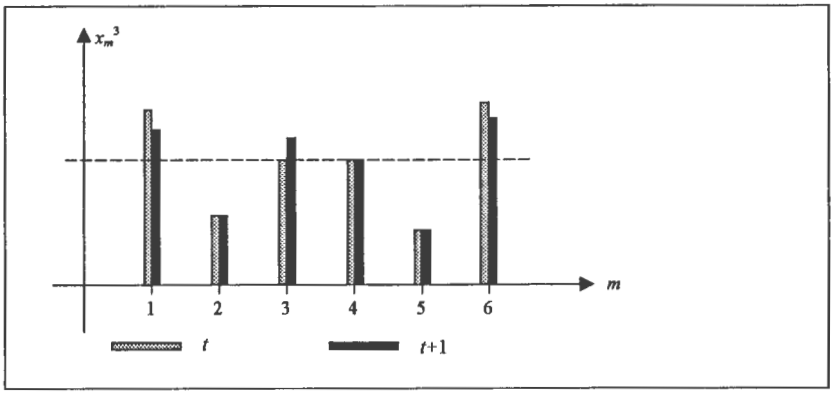
$$\delta_m^k(t) = \begin{cases} \Delta_m < 0 \text{ dla } m \in J^k \\ 0 \text{ dla } m \notin J^k \\ \Delta_x > 0 \text{ dla } m = J^k \end{cases} \qquad (36)$$

Metodę ilustruje rys. 9, na którym pokazano wagi w takcie t oraz w takcie $t+1$, w przypadku $m = 6$, przy pojawieniu się obiektu należącego do klasy 3-ej. Z rysunku wynika:

$$J^3 = \{1, 4, 6\};\ \Delta_1 < 0,\ \Delta_3 > 0,\ \Delta_6 < 0,\ \Delta_2 = \Delta_4 = \Delta_5 = 0$$

Procedura taka jest powtarzana do uzyskania pustych zbiorów

J^k; $k = 1,2,...,K$



Rys. 9 Korekta wag metodą przyrostów

Metoda gradientowa. Uczenie sieci neuronowej, przeznaczonej do rozpoznawania obiektów, polega na iteracyjnym, nastrajaniu wag zgodnie z wzorem:

$$w^m(t+1) = w^m(t) - \eta[\partial q/\partial w^m]^T;\ m = 1, 2, ..., K \qquad (37)$$

gdzie: $q = \sum_{m=1}^{K} \sum_{n=1}^{N} [[y^{*m}(u_n) - y_n^m(t)]^2$ (38)

$$y_n^m(t) = (u_n)^T w^m(t) \quad (39)$$

$y^{*m}(u_n)$ – pożądany sygnał na wyjściu neuronu m – tego przy pojawieniu się obiektu u_n, N – liczba eksperymentów.

Taki algorytm uczenia jest zbieżny [Tadeusiewicz,1993], przy odpowiednio dobranym, niezbyt dużym współczynniku η. Oznacza to, że w kolejnych iteracjach wagi zmierzają do określonych wartości W^*, przy których sieć dokonuje właściwej klasyfikacji obiektów. Szybkość zbieżności algorytmu (*37*) zależy od doboru warunków początkowych $W(0)$ oraz od ciągów uczących u_n.

W fazie pracy właściwej wektor cech u doprowadzany jest do wejścia każdego z K neuronów o wagach W^*. Dalej można stosować regułę, zgodnie z którą obiekt generujący ten wektor należy do klasy, która odpowiada neuronowi o największym sygnale wyjściowym.

5.2 Zastosowanie sieci nieliniowych do klasyfikacji obiektów

Aby przybliżyć podstawowe idee zawarte w stosowaniu nieliniowych funkcji aktywacji przedstawimy metodę rozwiązania zadania podziału obiektów o wektorach cech $u \in R^{I+1}$ na dwie klasy: $u \in S_1$ oraz $u \notin S_1$, gdzie $S_1 \subset R^{I+1}$. Rozpoczniemy od zadania, w którym podzbiór S_1 jest wypukły i ograniczony hiperpłaszczyznami, tzn. określony nierównościami liniowymi.

W sieci neuronowej jednowarstwowej o K neuronach z jednakowymi funkcjami aktywacji $f(x^k)$ przy pojawieniu się na wejściu wektora cech $\boldsymbol{u}$, na wyjściu k-tego neuronu pojawia się sygnał (rys. 10):

$$y^k = f(x^k); \quad k = 1, 2, ..., K \tag{40}$$

gdzie:

$$x^k = \boldsymbol{u}^T \boldsymbol{w}^k; \quad k = 1, 2, ..., K \tag{41}$$

W zapisie wektorowym (*40*) przyjmuje postać:

$$\boldsymbol{y} = [f(x^1), f(x^2), ..., f(x^K)]^T \tag{42}$$

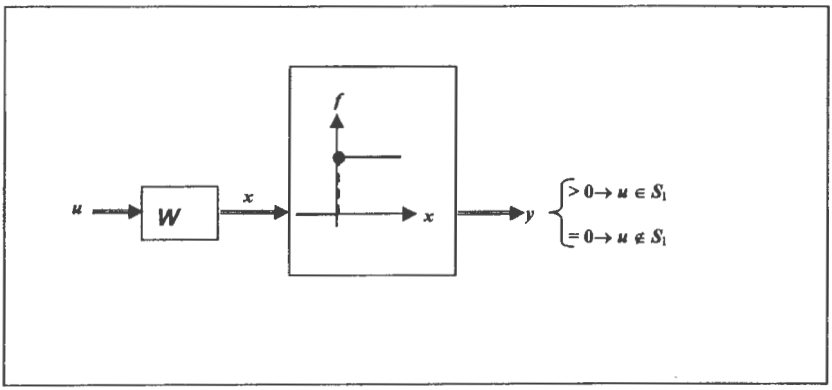
gdzie: $\boldsymbol{y} = [y_1, y_2, ..., y_K]^T$ – wektor sygnałów wyjściowych.

Zastosujemy unipolarne skokowe funkcje aktywacji (rys. 3a). Przy pobudzeniu sygnałem $\boldsymbol{u}$ na wyjściach neuronów otrzymamy

$$y^k = f(x_k) = \begin{cases} 1 \text{ gdy } x^k \geq 0 \\ 0 \text{ gdy } x^k < 0 \end{cases}; \quad k = 1,2,\ldots,K \tag{43}$$

gdzie :

$$\boldsymbol{w}^k = [w^k_0, w^k_1 \; w^k_2 \ldots w^k_I]^T \tag{44}$$



Rys. 10 Sieć jednowarstwowa

W (*44*) indeks k oznacza numer neuronu, indeks dolny – numer wejścia.

W postaci wektorowej (*43*) przyjmuje postać (*20*):

$$\boldsymbol{x} = \boldsymbol{W}\boldsymbol{u} \tag{45}$$

gdzie:

$$\boldsymbol{x} = [x^1, x^2, ..., x^K]^T \tag{46}$$

Odwzorowania (*43*) $f: R^1 \rightarrow \{0, 1\}$ dzielą przestrzeń $(u_0, u_1, u_2, ..., u_I)$ na dwie części, których granice wyznaczone są K hiperpłaszczyznami o równaniach:

$$\boldsymbol{W}\cdot\boldsymbol{u} = \boldsymbol{0} \tag{47}$$

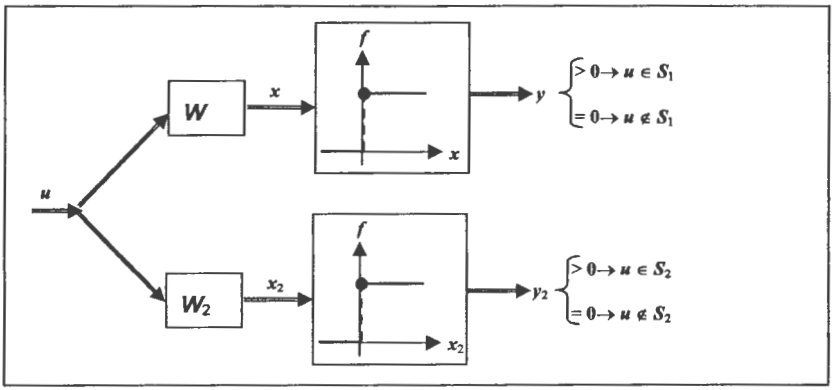
W przypadku $I = 2$ są to proste o postaci:

$$w^k{}_0 + u_1 w^k{}_1 + u_2 w^k{}_2 = 0\ ;\ k = 1, 2, ..., K \tag{48}$$

Jedna z tych części obejmuje punkty $(u_1, u_2, ..., u_I)$ I - wymiarowej przestrzeni, które spełniają nierówność wektorową:

$$Wu \geq 0 \text{ lub } x \geq 0 \qquad (49)$$

Odpowiednim doborem wag można ukształtować dowolny wypukły podzbiór (rys. 11) S_1, spełniający (*49*):



Rys. 11 Wypukły liniowy podzbiór S_1

$$S_1 = \{(u_1, u_2, ..., u_I) \subseteq R^I: Wu \geq 0\} \qquad (50)$$

Zgodnie z tym, jeśli przy pojawieniu się wektora cech u wszystkie neurony sieci będą pobudzone, tzn. sygnały wyjściowe wszystkich neuronów będą dodatnie ($y^k > 0$; $k = 1, 2, ..., K$) – to odpowiadający temu wektorowi obiekt należy do podzbioru S_1. Jeśli chociaż jeden neuron będzie niepobudzony ($\exists k : y^k = 0$) – obiekt nie należy do podzbioru S_1.

Do wyróżnienia podzbioru liniowego ale niewypukłego potrzebna jest sieć składająca się co najmniej z dwóch równoległych warstw, do których doprowadzany jest ten sam sygnał u.

Niech wagi drugiej warstwy będą nastrojone tak, aby wyróżnić zbiór wypukły S_2:

$$S_2 = \{(u_1, u_2, ..., u_I) \subseteq R^I : W_2 u \geq 0\} \qquad (51)$$

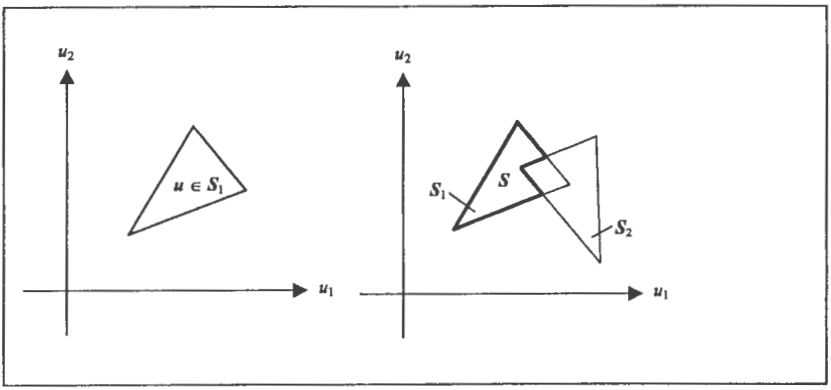
gdzie W_2 – macierz wag drugiej warstwy:

$$W_2 = \begin{bmatrix} w_{02}^1 & w_{12}^1 & w_{22}^1 & \dots & w_{K2}^1 \\ w_{02}^2 & w_{12}^2 & w_{22}^2 & \dots & w_{K2}^2 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ w_{02}^K & w_{12}^K & w_{21}^K & \dots & w_{K2}^K \end{bmatrix} \qquad (52)$$

Dwuwarstwowa sieć neuronowa może przez dobór W oraz W_2 kształtować i wyróżniać obiekty należące do **niewypukłych** zbiorów liniowych.

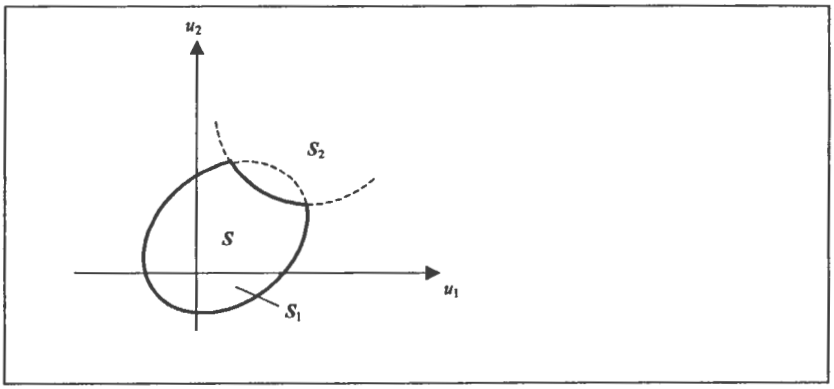
Takim zbiorem jest np. zbiór S (rys. 12), którego elementy należące do zbioru S_1 nie należą do zbioru S_2:

$$S = \{(u_1, u_2, ..., u_I) \subseteq R^I : [(u_1, u_2, ..., u_I \in S_1) \cup (u_1, u_2, ..., u_I) \notin S_2]\} \qquad (53)$$



Rys. 12 Wyróżnianie zbiorów liniowych niewypukłych

Zastosowanie nieliniowych ciągłych funkcji aktywacji, np. funkcji (7) lub (8), pozwala uzyskać krzywoliniowe hiperpowierzchnie rozgraniczające obiekty należące do różnych klas (rys. 13).



Rys. 13 Wyróżnianie zbiorów nieliniowych

6 Aproksymacja nieznanej funkcji

Sieci neuronowe mogą z powodzeniem rozwiązywać zadania aproksymacji nieznanej funkcji, przez liniową kombinację wybranych funkcji, zwanych funkcjami bazowymi. Znane są tylko zestawy N pomiarów, każdy z których obejmuje wartości I wielkości wejściowych:

$$\boldsymbol{u}(n) = [u_1(n), u_2(n), ..., u_I(n)]^T \qquad (54)$$

oraz odpowiadającą tym wartościom wejściowym – wartość wielkości wyjściowej $y(n)$, $n = 1,2, \ , N$.

Do celów aproksymacji może służyć sieć radialna o K neuronach I - wejściowych i K funkcjach bazowych w warstwie wyjściowej (rys.7). Model takiej sieci ma postać:

$$y = w_0 + \sum_{k=1}^{K} w_k \varphi_k(u) \tag{55}$$

lub w zapisie wektorowym:

$$y = w^T \varphi(u) \tag{56}$$

gdzie: $\varphi(u) = [1, \varphi_1(u), \varphi_2(u), ..., \varphi_K(u)]^T$

Jako funkcje bazowe stosowane są w wielu przypadkach funkcje radialne- (funkcje ciągłe, określone w kuli jednostkowej, symetryczne względem środka kuli). Funkcje takie zapewniają najlepszą aproksymację [Bishop, 1995], tzn. w określonym zbiorze funkcji aproksymujących, o dobieranych w procesie uczenia parametrach, dają minimum błędu aproksymacji.

Najczęściej stosowane są wielowymiarowe funkcje Gaussa o postaci:

$$\varphi(r_k) = \exp\left(-\frac{r_k^2}{2\sigma_k^2}\right); \; k = 1, 2, ..., K \tag{57}$$

$$r_k = \left(\frac{1}{I+1}\right)\sqrt{\sum_{i=0}^{I}(u_i - \mu_k)^2} \tag{58}$$

gdzie: $\sigma_k > 0$ – parametr określający „szerokość", μ_k – środek (centrum) k – tej funkcji bazowej. Parametr r_k charakteryzuje odległość (wg miary euklidesowej) wektora u od środka k-tej funkcji bazowej μ_k.

Wybór funkcji bazowych oraz ich liczba mają duży wpływ na jakość aproksymacji. Jednak brak jest teoretycznie uzasadnionych, efektywnych metod wyboru. Stosowane są na ogół metody heurystyczne.

Uczenie sieci neuronowej o określonej liczbie i rodzaju funkcji bazowych, polega na minimalizacji funkcji jakości aproksymacji $q(w, \mu)$ względem wag $w = (w_1, w_2, ..., w_K)$ oraz środków funkcji bazowych $\mu = (\mu_1, \mu_2, ..., \mu_K)$:

$$q(w, \mu) = \sum_{k=1}^{K} \sum_{n=1}^{N} \{y(n) - w_k \varphi_k[r_k(n)]\}^2 \qquad (59)$$

gdzie:

$$r_k(n) = \left(\frac{1}{I+1}\right) \sqrt{\sum_{i=0}^{I} [u_i(n) - \mu_k]^2} \qquad (60)$$

Warto wspomnieć, że w niektórych zastosowaniach stosuje się łączenie sieci neuronowych – z logiką rozmytą [Rutkowska, 1997]. W takich przypadkach współczynniki funkcji regresji są liczbami rozmytymi.

7 Uczenie bez nauczyciela w sieciach neuronowych

Dotychczas ograniczyliśmy się do omówienia procesów uczenia z udziałem nauczyciela. Jego rola polega na przekazywaniu sieci w fazie uczenia pełnej informacji o obiekcie, w tym o klasie do której obiekt należy. Informację tą można porównać z decyzją sieci o klasie obiektu, co daje możliwość określenia błędu, który służy do odpowiedniej korekcji wag. W rezultacie sieć uczy się pożądanych zachowań. Mamy więc

układ strukturalnie podobny do układu regulacji z ujemnym sprzężeniem zwrotnym, który ma na celu dostosowanie wielkości wyjściowej do wielkości zadanej.

Jednakże w wielu przypadkach praktycznych użytkownik sieci nie dysponuje pełną wiedzą o obiekcie, którą może użytkować na etapie uczenia sieci. Również mózg wielu czynności uczy się, bez podpowiadania z zewnątrz jakie powinny być właściwe decyzje. Zakres stosowania sieci w takich warunkach jest znacznie ograniczony. Ponadto należy liczyć się z długim i kosztownym ciągiem błędnych decyzji na początku procesu uczenia. Nawet Mojżesz musiał się poparzyć, zanim zrozumiał, że ogień może być niebezpieczny!

Sposoby uczenia sieci neuronowych można podzielić na następujące podstawowe grupy:

uczenie nadzorowane (supervised learning)

- z nauczycielem
- z krytykiem

uczenie nienadzorowane (unsupervised learning)

- typu Hebba
- typu konkurencyjnego

O uczeniu z nauczycielem była już mowa. Pewną odmianą tej metody uczenia jest tzw. **uczenie z krytykiem.** W tym przypadku układ decyzyjny jest informowany jedynie o tym, czy dokonana zmiana daje wyniki pożądane z punktu widzenia podejmowania poprawnych decyzji. Jeśli odpowiedź jest pozytywna następuje działanie w tym samym kierunku przy następnych próbach. Jest to więc metoda prób i błędów, szeroko stosowana w najrozmaitszych dziedzinach.

Tryb uczenia **bez nauczyciela** stosuje się wtedy, gdy brak jest informacji o przynależności obiektu do określonej klasy. Taka sytuacja może np. wystąpić przy odczytywaniu tekstów pisanych w językach martwych. Zastosowanie sieci neuronowych pozwala w takich przypadkach grupować obiekty (np. heroglify) na klasy. Nie sposób jednak określić jakie dźwięki odpowiadają jakim znakom obrazkowym.

W omawianej sytuacji klasyfikacji obiektów stosowane jest nastrajanie wag zgodne z zasadę, że waga i-tej cechy k-tego neuronu w $(t+1)$- ej iteracji wzrasta proporcjonalnie do wpływu i-tej cechy na k-ty neuron:

$$w_i^k(t+1) = w_i^k(t) + \eta\, u_i(t)\, x^k(t);\ k = 1, 2, ..., K\ ,\ i = 0, 1, 2, ..., I \qquad (61)$$

gdzie: $$x^k(t) = [\boldsymbol{u}(t)]^T \boldsymbol{w}^k(t) \qquad (62)$$

$\boldsymbol{u}(t) = [1, u_1(t), u_2(t), ..., u_I]^T$ - wektor cech

Zgodnie z (62) układ korekcyjny wzmacnia te wagi, które odpowiadają cechom silnie skorelowanym z określoną klasą, co pozwala z powodzeniem nauczyć sieć rozdzielania obiektów na klasy. Np. mając do dyspozycji dane charakteryzujące stan zdrowotny grupy pacjentów, można nauczyć sieć neuronową zaliczania chorych do kategorii chorobowych, każda z których wymaga odrębnej terapii.

Zasada nastrajania wag (62), zwana też regułą Hebba, jest oparta na obserwacjach neurobiologicznych, z których wynika, że siła sprzężenia pomiędzy dwoma biologicznymi neuronami ulega wzmocnieniu, jeśli oba są pobudzone, natomiast słabnie – gdy oba są nieaktywne.

Stosowane są liczne modyfikacje reguły Hebba. Jedną z nich jest tzw. uczenie Kohonena (uczenie z rywalizacją, uczenie konkurencyjne, uczenie „zwycięzca bierze wszystko"). Polega ono na tym, że, neurony „rywalizują" między sobą, w tym sensie, że zmiany wag w kolejnych iteracjach, dotyczą tylko tego neuronu na którego wyjściu pojawi się największy sygnał (zwycięzca bierze wszystko). Inną modyfikacją tej reguły jest objęcie procedurą nastrajania, również wag neuronów sąsiednich, według odległości ich wektorów wag od wektorów wag zwycięzcy (zwycięzca bierze najwięcej).

Stosowanie sieci neuronowej uczonej bez nadzoru nauczyciela jest poważnie ograniczone, co wynika z faktu, że podczas uczenia, sieć nie otrzymuje wzorców do poprawnego działania. Można wprawdzie dokonać grupowania obiektów podobnych. Nie można jednak stwierdzić, czy dokonany podział jest istotny z punktu widzenia celów grupowania. W sieciach uczonych bez nadzoru nie wiadomo czy liczba klas, na które sieć dzieli rozpoznawane obiekty jest właściwa. Jeśli liczba klas jest większa, od liczby znaków, to ten sam znak może być zaliczany do kilku klas jednocześnie. Jeśli liczba klas jest mniejsza od liczby znaków – różne w istocie znaki mogą być zaliczane do tej samej klasy. Nie wiadomo też, który neuron reprezentuje którą klasę. Występuje tu również duża zależność procesu klasyfikacji od warunków początkowych.

W sieciach neuronowych, w stosunku do sieci biologicznych, występuje istotna ograniczająca różnica, która obniża sens analogii bio - technicznych. Polega ona na subiektywnym charakterze w sieciach neuronowych pojęcia „sąsiedztwa", które w sieciach nerwowych ma klarowną interpretację topologiczną. W sieciach neuronowych pojęcie odległości między neuronami ma charakter subiektywny, zależy od arbitralnego sposobu numeracji.

Sieci neuronowe dorobiły się nadzwyczaj obszernej literatury. Tym nie mniej wiele zagadnień, dotyczących zarówno ich teorii jak i zastosowań, nie znalazło jeszcze zadawalającego rozwiązania. Dotyczy to w szczególności struktury sieci (liczba warstw, liczba neuronów, topologia), kategorii funkcji aktywacji, metod uczenia. Stosowane są w szeroko metody heurystyczne i intuicja. Dlatego też należy spodziewać się w najbliższych latach znacznego postępu i ugruntowania teoretycznych podstaw sieci neuronowych oraz efektywnych zastosowań tej dziedziny modelowania.

Literatura

Davidor Y. [1991] *Genetic Algorithms; A Heuristic Strategy for Optimization*, World Scientific, Singapore

Goldberg D.E. [1998]: *Algorytmy genetyczne i ich zastosowanie*, WNT

Głuszek A. [2000]: *Neuronowo-rozmyte systemy syntezy wiedzy z danych*, Rozprawa doktorska, IBS PAN

Haykin S. [1999]: *Neural Networks; A comprehensive Foundation*, Macmillan College Publishing Company< New York

Mańdziuk J.[2000]: *Sieci neuronowe typu Hopfielda; teoria i przykłady zastosowań*, Akademicka Oficyna Wydawnicza Exit, Warszawa

McCulloch W.S., Pitts W. [1943]: *A Logical Calculus of the Ideas Imminent in Nervous Activity*, Bulletin of Mathematical Biophysics, 5, str. 115-133

Osowski S. [1996]: *Sieci neuronowe*, Oficyna Wydawnicza Politechniki Warszawskiej

Rosenblatt F. [1962] *The Principles of Neurodynamics*, Spartan Books, Washington

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

Witkowska D.[2000]: *Sztuczne sieci neuronowe w analizach ekonomicznych*, Menadżer, Łódź