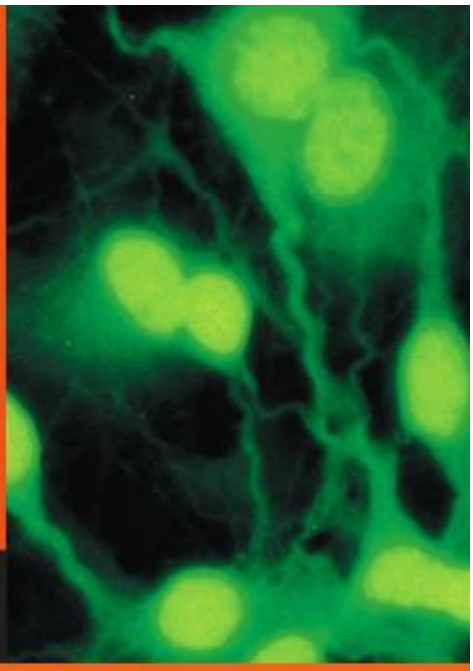


INFORMATYKA ■ ZASTOSOWANIA

Leszek Rutkowski

Metody i techniki sztucznej inteligencji

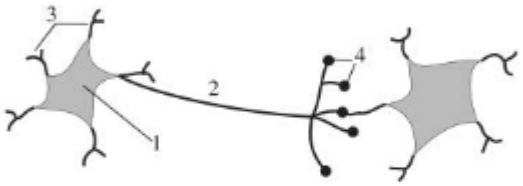
WYDAWNICTWO NAUKOWE PWN

6.2. Neuron i jego modele

6.2.1. Budowa i działanie pojedynczego neuronu

Podstawowym elementem systemu nerwowego jest komórka nerwowa nazywana *neuronem*. Na rysunku 6.1 pokazany jest jej uproszczony schemat. W neuronie możemy wyróżnić *ciało komórki* (nazywane somą) oraz otaczające je dwa rodzaje wypustek: wypustki wprowadzające informację do neuronu, tzw. *dendryty* i wypustkę wyprowadzającą informacje z neuronu, tzw. *akson*. Każdy neuron ma dokładnie jedną wypustkę

wyprowadzającą, poprzez którą może wysyłać impulsy do wielu innych neuronów. Pojedynczy neuron przyjmuje pobudzenie od ogromnej liczby neuronów dochodzącej do tysiąca. Jak wspomnieliśmy wcześniej, w mózgu człowieka jest ok. 100 miliardów neuronów, które oddziałują na siebie poprzez ogromną liczbę połączeń. Jeden neuron przekazuje pobudzenie innym neuronom przez złącza nerwowe nazywane *synapsami*, przy czym transmisja sygnałów odbywa się na drodze skomplikowanych procesów chemiczno-elektrycznych. Synapsy pełnią funkcję przekaźników informacji, a w wyniku ich działania pobudzenie może być wzmocnione lub osłabione. W rezultacie do neuronu dochodzą sygnały, z których część wywiera wpływ pobudzający, a część hamujący. Neuron sumuje impulsy pobudzające i hamujące. Jeżeli ich suma algebraiczna przekracza pewną wartość progową, to sygnał na wyjściu neuronu jest przesyłany — poprzez akson — do innych neuronów.



Rys. 6.1. Uproszczony schemat neuronu i jego połączenia z sąsiednim neuronem: 1 — ciało komórki, 2 — akson, 3 — dendryty, 4 — synapsy

Przedstawimy teraz model neuronu nawiązujący do pierwszych prób sformalizowania opisu działania komórki nerwowej. Wprowadźmy następujące oznaczenia: n — liczba wejść w neuronie, $x_1, \ldots, x_n$ — sygnały wejściowe, $\mathbf{x} = [x_1, \ldots, x_n]^T$, $w_0, \ldots, w_n$ — wagi synaptyczne, $\mathbf{w} = [w_0, \ldots, w_n]^T$, y — wartość wyjściowa neuronu, w_0 — wartość progowa, f — funkcja aktywacji.

Formuła opisująca działanie neuronu wyraża się zależnością

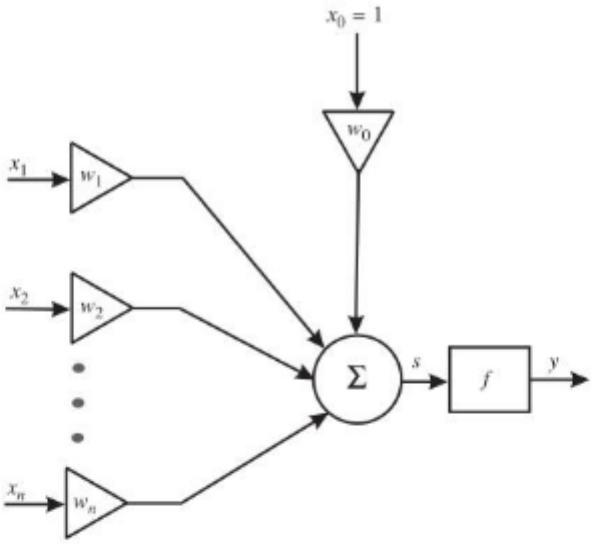
$$y = f(s), \tag{6.1}$$

w której

$$s = \sum_{i=0}^{n} x_i w_i. \tag{6.2}$$

Wzory (6.1) i (6.2) opisują neuron przedstawiony na rysunku 6.2. Funkcja aktywacji f może przybierać różną postać w zależności od konkretnego modelu neuronu.

Jak wynika z powyższych wzorów, działanie neuronu jest bardzo proste. Najpierw sygnały wejściowe $x_0, x_1, \ldots, x_n$ zostają pomnożone przez odpowiadające im wagi w_0, $w_1, \ldots, w_n$. Otrzymane w ten sposób wartości należy następnie zsumować. W wyniku powstaje sygnał s odzwierciedlający działanie części liniowej neuronu. Sygnał ten jest poddawany działaniu funkcji aktywacji, najczęściej nieliniowej. Zakładamy, że wartość sygnału x_0 jest równa 1, natomiast wagę w_0 nazywa się *progiem* (ang. *bias*). Gdzie zatem kryje się wiedza w tak opisanym neuronie? Otóż wiedza zapisana jest właśnie w wagach. Największym zaś fenomenem jest to, iż w łatwy sposób (za pomocą algorytmów opisanych w dalszej części rozdziału) można neurony uczyć, a więc odpowiednio dobierać wagi. Na rysunku 6.2 przedstawiliśmy ogólny schemat neuronu, jednakże w sieciach

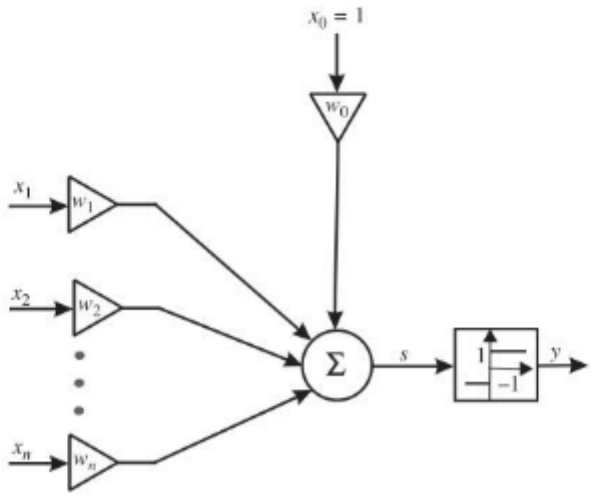


Rys. 6.2. Model neuronu

stosuje się różne jego modele. Niektóre z nich omówimy w następnych punktach. Należy jeszcze wspomnieć, iż podobnie jak w mózgu komórki nerwowe łączą się ze sobą, tak i w przypadku tworzenia modeli matematycznych sztuczne neurony przedstawione na rysunku 6.2 łączy się ze sobą, tworząc wielowarstwowe sieci neuronowe. Sposób łączenia neuronów, a także metody uczenia powstałych w ten sposób struktur poznamy w kolejnych punktach tego rozdziału.

6.2.2. Perceptron

Na rysunku 6.3 przedstawiono schemat perceptronu.



Rys. 6.3. Schemat perceptronu

Działanie perceptronu można opisać zależnością

$$y = f\Big(\sum_{i=1}^{n} w_i x_i + \theta\Big). \tag{6.3}$$

Zauważmy, że wzór (6.3) koresponduje z ogólnym zapisem (6.1), jeżeli $\theta = w_0$. Funkcja f może być nieciągłą funkcją skokową — bipolarną (przyjmuje wartości -1 lub 1) lub unipolarną (przyjmuje wartości 0 lub 1). Do dalszych rozważań przyjmiemy, iż funkcja aktywacji jest bipolarna

$$f(s) = \begin{cases} 1, & \text{gdy } s > 0, \\ -1, & \text{gdy } s \leq 0. \end{cases} \tag{6.4}$$

Perceptron ze względu na swą funkcję aktywacji przyjmuje tylko dwie różne wartości wyjściowe, może więc klasyfikować sygnały podane na jego wejście w postaci wektorów $\mathbf{x} = [x_1, \ldots, x_n]^T$ do jednej z dwóch klas. Na przykład perceptron z jednym wejściem może oceniać, czy sygnał wejściowy jest dodatni, czy ujemny. W przypadku dwóch wejść x_1 i x_2 perceptron dzieli płaszczyznę na dwie części. Podział ten wyznacza prosta o równaniu

$$w_1 x_1 + w_2 x_2 + \theta = 0. \tag{6.5}$$

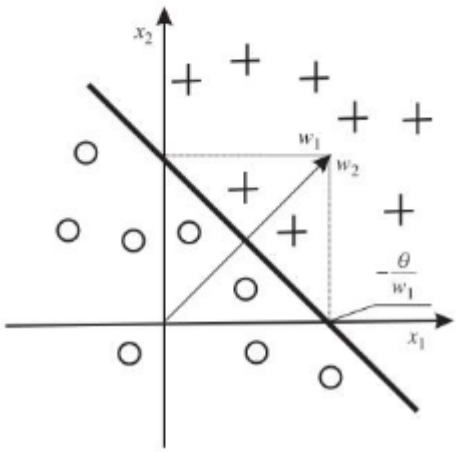
Zatem równanie (6.5) można napisać

$$x_2 = -\frac{w_1}{w_2} \cdot x_1 - \frac{\theta}{w_2}. \tag{6.6}$$

W ogólnym przypadku, gdy perceptron ma n wejść, wówczas dzieli n-wymiarową przestrzeń wektorów wejściowych $\mathbf{x}$ na dwie półprzestrzenie. Są one rozdzielone $n-1$-wymiarową hiperpłaszczyzną, nazywaną *granicą decyzyjną*, daną wzorem

$$\sum_{i=1}^{n} w_i x_i + \theta = 0. \tag{6.7}$$

Na rysunku 6.4 przedstawiono granicę decyzyjną dla $n = 2$. Należy zauważyć, że prosta wyznaczająca podział przestrzeni jest zawsze prostopadła do wektora wag $\mathbf{w} = [w_1, w_2]^T$.



Rys. 6.4. Granica decyzyjna dla $n = 2$

Zgodnie z tym, co napisaliśmy we wstępie, perceptron można uczyć. W czasie tego procesu jego wagi są modyfikowane. Metoda uczenia perceptronu należy do grupy algorytmów zwanych *uczeniem z nauczycielem* lub *uczeniem nadzorowanym*. Uczenie tego typu polega na tym, iż podaje się na wejście perceptronu sygnały $\mathbf{x}(t) = [x_0(t), x_1(t), \ldots, x_n(t)]^T$, $t = 1, 2, \ldots$, dla których znamy prawidłowe wartości sygnałów wyjściowych $d(t)$, $t = 1, 2, \ldots$, zwanych sygnałami wzorcowymi. Zbiór takich próbek wejściowych wraz z odpowiadającymi im wartościami sygnałów wzorcowych nazywamy *ciągiem uczącym*. W metodach tych po podaniu wartości wejściowych oblicza się sygnał wyjściowy neuronu. Następnie modyfikuje się wagi w ten sposób, aby minimalizować błąd między sygnałem wzorcowym a wyjściem perceptronu. Stąd właśnie nazwa „uczenie z nauczycielem", ponieważ nauczyciel określa, jaka powinna być wartość wzorcowa. Oczywiście można się domyślać, iż istnieją algorytmy uczące sieci bez nauczyciela, ale o nich napiszemy w następnych punktach tego rozdziału. Algorytm uczenia perceptronu przedstawia się następująco:

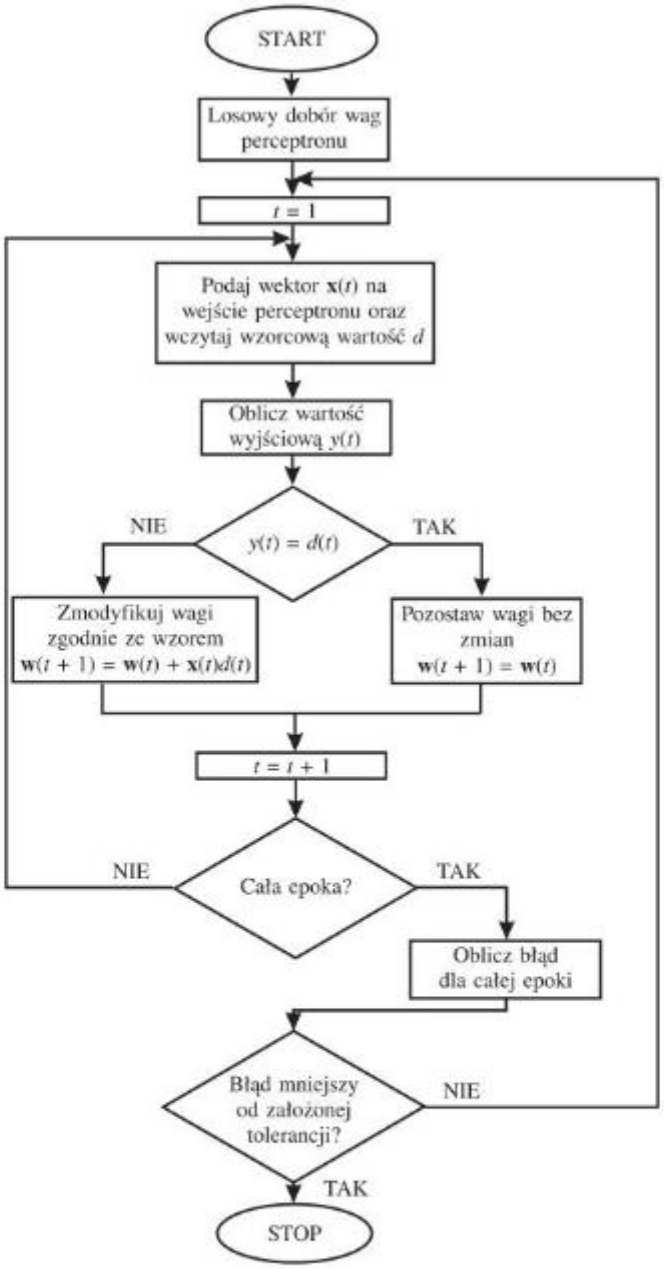
1. Wybieramy w sposób losowy wagi początkowe perceptronu.
2. Na wejścia neuronu podajemy wektor uczący $\mathbf{x}$, przy czym $\mathbf{x} = \mathbf{x}(t) = [x_0(t), x_1(t), \ldots, x_n(t)]^T$, $t = 1, 2, \ldots$
3. Obliczamy wartość wyjściową perceptronu y, zgodnie ze wzorem (6.3).
4. Porównujemy wartość wyjściową $y(t)$ z wartością wzorcową $d = d\big(\mathbf{x}(t)\big)$ znajdującą się w ciągu uczącym.
5. Dokonujemy modyfikacji wag według zależności:
   a) jeżeli $y\big(\mathbf{x}(t)\big) \neq d\big(\mathbf{x}(t)\big)$, to $w_i(t+1) = w_i(t) + d\big(\mathbf{x}(t)\big)x_i(t)$;
   b) jeżeli $y\big(\mathbf{x}(t)\big) = d\big(\mathbf{x}(t)\big)$, to $w_i(t+1) = w_i(t)$, czyli wagi pozostają bez zmian.
6. Wracamy do punktu 2.

Algorytm powtarza się tak długo, aż dla wszystkich wektorów wejściowych wchodzących w skład ciągu uczącego błąd na wyjściu będzie mniejszy od założonej tolerancji. Na rysunku 6.5 przedstawiono schemat blokowy uczenia perceptronu. Działanie pętli wewnętrznej na tym rysunku dotyczy tzw. jednej *epoki*, na którą składają się dane tworzące ciąg uczący. Działanie pętli zewnętrznej odzwierciedla możliwość wielokrotnego stosowania tego samego ciągu uczącego, aż zostanie spełniony warunek zatrzymania algorytmu.

Wykażemy, iż algorytm uczenia perceptronu jest zbieżny. Twierdzenie o zbieżności algorytmu uczenia perceptronu formułuje się następująco:

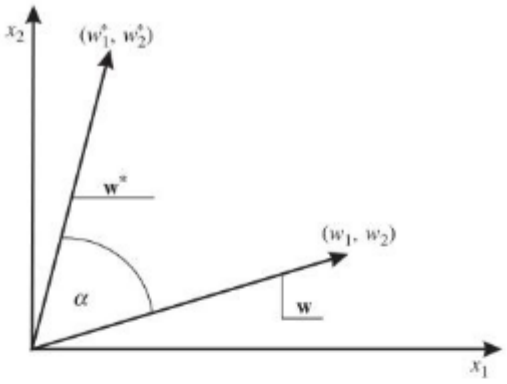
Jeżeli istnieje zestaw wag $\mathbf{w}^* = [w_1^*, \ldots, w_n^*]^T$ klasyfikujący wzorce uczące $\mathbf{x} = [x_1, \ldots, x_n]^T$ poprawnie, tzn. wyznaczający odwzorowanie $y = d(\mathbf{x})$, to algorytm uczący znajdzie rozwiązanie w skończonej liczbie iteracji dla dowolnych wartości początkowych wektora wag $\mathbf{w}$.

Zakładamy, że dane uczące reprezentują klasy liniowo separowane, gdyż tylko wtedy można nauczyć perceptron. Pokażemy, że istnieje skończona liczba kroków modyfikacji wag, po których perceptron będzie realizował odwzorowanie $y = d(\mathbf{x})$. Ze względu na to, że funkcja aktywacji w perceptronie jest typu sgn, długość wektora $\mathbf{w}^*$ możemy przyjąć dowolną, np. równą 1, tzn. $\|\mathbf{w}^*\| = 1$. Zatem w czasie uczenia wektor $\mathbf{w}$



Rys. 6.5. Schemat blokowy algorytmu uczenia perceptronu

wystarczy modyfikować tak, aby pokazany na rysunku 6.6 kąt α był równy 0. Oczywiście wtedy $\cos(\alpha) = 1$. Z faktu, że $|\mathbf{w}^* \circ \mathbf{x}| > 0$ (znak $\circ$ w tym przypadku oznacza iloczyn skalarny wektorów) i $\mathbf{w}^*$ jest rozwiązaniem, wynika istnienie takiej stałej $\delta > 0$, dla której $|\mathbf{w}^* \circ \mathbf{x}| > \delta$ dla wszystkich wektorów $\mathbf{x}$ z ciągu uczącego. Z definicji iloczynu



Rys. 6.6. Ilustracja działania algorytmu uczenia perceptronu dla $n = 2$

skalarnego wynika, że

$$\cos(\alpha) = \frac{\mathbf{w}^* \circ \mathbf{w}}{\sqrt{\|\mathbf{w}^*\|^2 \|\mathbf{w}\|^2}}. \tag{6.8}$$

Ponieważ

$$\sqrt{\|\mathbf{w}^*\|^2} = \|\mathbf{w}^*\| = 1, \tag{6.9}$$

więc

$$\cos(\alpha) = \frac{\mathbf{w}^* \circ \mathbf{w}}{\|\mathbf{w}\|}. \tag{6.10}$$

Zgodnie z algorytmem uczenia perceptronu, wagi modyfikowane są dla podanego wektora wejściowego $\mathbf{x}$ według następującej zależności: $\mathbf{w}' = \mathbf{w} + \Delta\mathbf{w}$, gdzie $\Delta\mathbf{w} = d(\mathbf{x})\mathbf{x}$. Oczywiście zakładamy, że na wyjściu sieci pojawił się błąd i korekcja wag jest niezbędna. Zauważmy, że

$$\mathbf{w}' \circ \mathbf{w}^* = \mathbf{w} \circ \mathbf{w}^* + d(\mathbf{x})\mathbf{w}^* \circ \mathbf{x}, \tag{6.11}$$

a zatem

$$\mathbf{w}' \circ \mathbf{w}^* = \mathbf{w} \circ \mathbf{w}^* + \mathrm{sgn}(\mathbf{w}^* \circ \mathbf{x})\mathbf{w}^* \circ \mathbf{x}. \tag{6.12}$$

Zachodzą następujące fakty:

(i) Jeżeli $\mathbf{w}^* \circ \mathbf{x} < 0$, to $\mathrm{sgn}(\mathbf{w}^* \circ \mathbf{x}) = -1$, więc $\mathrm{sgn}(\mathbf{w}^* \circ \mathbf{x})\mathbf{w}^* \circ \mathbf{x} = -1(\mathbf{w}^* \circ \mathbf{x}) > 0$,
(ii) Jeżeli $\mathbf{w}^* \circ \mathbf{x} > 0$, to $\mathrm{sgn}(\mathbf{w}^* \circ \mathbf{x}) = 1$, więc $\mathrm{sgn}(\mathbf{w}^* \circ \mathbf{x})\mathbf{w}^* \circ \mathbf{x} = 1(\mathbf{w}^* \circ \mathbf{x}) > 0$.

Zatem

$$\mathrm{sgn}(\mathbf{w}^* \circ \mathbf{x})\mathbf{w}^* \circ \mathbf{x} = |\mathbf{w}^* \circ \mathbf{x}|. \tag{6.13}$$

Zgodnie ze wzorami (6.12) i (6.13) możemy napisać

$$\mathbf{w}' \circ \mathbf{w}^* = \mathbf{w} \circ \mathbf{w}^* + |\mathbf{w}^* \circ \mathbf{x}|. \tag{6.14}$$

Wiemy też, że $|\mathbf{w}^* \circ \mathbf{x}| > \delta$, stąd

$$\mathbf{w}' \circ \mathbf{w}^* > \mathbf{w} \circ \mathbf{w}^* + \delta. \tag{6.15}$$

Oszacujmy teraz wartość $\|\mathbf{w}'\|^2$, pamiętając jednocześnie, iż rozpatrujemy przypadek, kiedy po podaniu na wejście wektora uczącego $\mathbf{x}$ na wyjściu sieci pojawia się błąd, tzn.

$$d(\mathbf{x}) = -\mathrm{sgn}(\mathbf{w} \circ \mathbf{x}). \tag{6.16}$$

Oczywiście

$$\|\mathbf{w}'\|^2 = \|\mathbf{w} + d(\mathbf{x})\mathbf{x}\|^2 = \|\mathbf{w}\|^2 + 2d(\mathbf{x})\mathbf{w} \circ \mathbf{x} + \|\mathbf{x}\|^2. \tag{6.17}$$

Wykorzystując zależności (6.16) i (6.17) oraz zakładając ograniczoność sygnałów wejściowych, mamy

$$\|\mathbf{w}'\|^2 < \|\mathbf{w}\|^2 + \|\mathbf{x}\|^2 = \|\mathbf{w}\|^2 + C. \tag{6.18}$$

Po t krokach modyfikacji wag sieci zależności (6.15) oraz (6.18) przybierają postać

$$\mathbf{w}(t) \circ \mathbf{w}^* > \mathbf{w} \circ \mathbf{w}^* + t\delta \tag{6.19}$$

oraz

$$\|\mathbf{w}(t)\|^2 < \|\mathbf{w}\|^2 + tC. \tag{6.20}$$

Korzystając ze wzorów (6.10), (6.19) i (6.20), otrzymujemy

$$\cos\alpha(t) = \frac{\mathbf{w}^* \circ \mathbf{w}(t)}{\|\mathbf{w}(t)\|} > \frac{\mathbf{w}^* \circ \mathbf{w} + t\delta}{\sqrt{\|\mathbf{w}\|^2 + tC}}. \tag{6.21}$$

Dlatego też musi istnieć takie $t = t_{\max}$, dla którego $\cos(\alpha) = 1$. A więc istnieje skończona liczba kroków modyfikacji wag, po których wektor wag początkowych będzie realizował odwzorowanie $y = d(\mathbf{x})$. Jeżeli przyjmiemy, że wartości startowe wag są równe 0, to

$$t_{\max} = \frac{C}{\delta^2}. \tag{6.22}$$

Przykład 6.1

Przedstawimy teraz przykład uczenia perceptronu. Omawiając jego działanie, stwierdziliśmy, iż ten model neuronu o dwóch wejściach dzieli płaszczyznę na dwie części (por. (6.5)). Wobec tego, jeżeli na płaszczyźnie umieścimy dwie klasy próbek, które będzie można rozdzielić za pomocą prostej, to perceptron w procesie uczenia będzie w stanie znaleźć tę linię podziału. W naszym doświadczeniu wykreślimy prostą wzorcową, oznaczoną literą L na rysunku 6.7. Przyjmiemy, że wszystkie punkty płaszczyzny leżące nad tą prostą reprezentują próbki z klasy 1, natomiast punkty leżące pod prostą L reprezentują klasę 2. Takich punktów na obu półpłaszczyznach znajduje się nieskończenie wiele i dlatego musimy wybrać po kilka próbek z każdej klasy. Chcemy, aby perceptron po nauczeniu dla próbek z klasy pierwszej na wyjściu podawał sygnał równy 1, natomiast dla próbek z klasy drugiej sygnał równy -1. W ten sposób zbudowaliśmy ciąg uczący przedstawiony w tabeli 6.1.

Tabela 6.1. Ciąg uczący w przykładzie 6.1

x_1	x_2	$d(\mathbf{x})$
2	1	1
2	2	1
0	6	1
−2	8	−1
−2	0	−1
0	0	−1
4	−20	−1

Oczywiście

$$\|\mathbf{w}'\|^2 = \|\mathbf{w} + d(\mathbf{x})\mathbf{x}\|^2 = \|\mathbf{w}\|^2 + 2d(\mathbf{x})\mathbf{w} \circ \mathbf{x} + \|\mathbf{x}\|^2. \tag{6.17}$$

Wykorzystując zależności (6.16) i (6.17) oraz zakładając ograniczoność sygnałów wejściowych, mamy

$$\|\mathbf{w}'\|^2 < \|\mathbf{w}\|^2 + \|\mathbf{x}\|^2 = \|\mathbf{w}\|^2 + C. \tag{6.18}$$

Po t krokach modyfikacji wag sieci zależności (6.15) oraz (6.18) przybierają postać

$$\mathbf{w}(t) \circ \mathbf{w}^* > \mathbf{w} \circ \mathbf{w}^* + t\delta \tag{6.19}$$

oraz

$$\|\mathbf{w}(t)\|^2 < \|\mathbf{w}\|^2 + tC. \tag{6.20}$$

Korzystając ze wzorów (6.10), (6.19) i (6.20), otrzymujemy

$$\cos\alpha(t) = \frac{\mathbf{w}^* \circ \mathbf{w}(t)}{\|\mathbf{w}(t)\|} > \frac{\mathbf{w}^* \circ \mathbf{w} + t\delta}{\sqrt{\|\mathbf{w}\|^2 + tC}}. \tag{6.21}$$

Dlatego też musi istnieć takie $t = t_{\max}$, dla którego $\cos(\alpha) = 1$. A więc istnieje skończona liczba kroków modyfikacji wag, po których wektor wag początkowych będzie realizował odwzorowanie $y = d(\mathbf{x})$. Jeżeli przyjmiemy, że wartości startowe wag są równe 0, to

$$t_{\max} = \frac{C}{\delta^2}. \tag{6.22}$$

Przykład 6.1

Przedstawimy teraz przykład uczenia perceptronu. Omawiając jego działanie, stwierdziliśmy, iż ten model neuronu o dwóch wejściach dzieli płaszczyznę na dwie części (por. (6.5)). Wobec tego, jeżeli na płaszczyźnie umieścimy dwie klasy próbek, które będzie można rozdzielić za pomocą prostej, to perceptron w procesie uczenia będzie w stanie znaleźć tę linię podziału. W naszym doświadczeniu wykreślimy prostą wzorcową, oznaczoną literą L na rysunku 6.7. Przyjmiemy, że wszystkie punkty płaszczyzny leżące nad tą prostą reprezentują próbki z klasy 1, natomiast punkty leżące pod prostą L reprezentują klasę 2. Takich punktów na obu półpłaszczyznach znajduje się nieskończenie wiele i dlatego musimy wybrać po kilka próbek z każdej klasy. Chcemy, aby perceptron po nauczeniu dla próbek z klasy pierwszej na wyjściu podawał sygnał równy 1, natomiast dla próbek z klasy drugiej sygnał równy -1. W ten sposób zbudowaliśmy ciąg uczący przedstawiony w tabeli 6.1.

Tabela 6.1. Ciąg uczący w przykładzie 6.1

x_1	x_2	$d(\mathbf{x})$
2	1	1
2	2	1
0	6	1
−2	8	−1
−2	0	−1
0	0	−1
4	−20	−1

6.2.3. Model Adaline

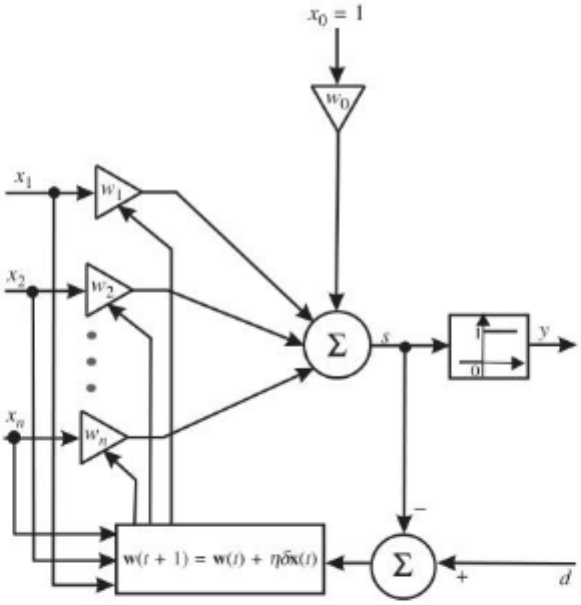
Schemat neuronu Adaline (ang. *Adaptive Linear Neuron*) przedstawiono na rysunku 6.8 Budowa tego neuronu jest bardzo podobna do modelu perceptronu, a jedyna różnica dotyczy algorytmu uczenia. Sposób wyznaczania sygnału wyjściowego jest identyczny z przedstawionym w poprzednim punkcie dotyczącym perceptronu. Jednak w przypadku neuronu typu Adaline porównuje się sygnał wzorcowy d z sygnałem s na wyjściu części liniowej neuronu (sumator). Stąd pochodzi nazwa tego typu neuronów. W ten sposób otrzymujemy błąd dany wzorem

$$\varepsilon = d - s. \tag{6.23}$$

Uczenie neuronu, czyli dobór wag, sprowadza się do minimalizacji funkcji określonej w sposób następujący:

$$Q(\mathbf{w}) = \frac{1}{2}\varepsilon^2 = \frac{1}{2}\left[d - \left(\sum_{i=0}^{n} w_i x_i\right)\right]^2. \tag{6.24}$$

Miarę błędu (6.24) określa się mianem *błędu średniego kwadratowego*. Uwzględniając tylko część liniową neuronu, możemy do modyfikacji wag użyć algorytmów gra-



Rys. 6.8. Schemat neuronu Adaline

dientowych, gdyż funkcja celu zdefiniowana zależnością (6.24) jest różniczkowalna. Do minimalizacji tejże funkcji użyjemy metody największego spadku. Metoda ta zostanie dokładniej omówiona przy okazji opisywania algorytmu wstecznej propagacji błędów. Wagi w neuronie typu Adaline modyfikuje się zgodnie ze wzorem

$$w_i(t+1) = w_i(t) - \eta \frac{\partial Q(w_i)}{\partial w_i}, \tag{6.25}$$

w którym η jest współczynnikiem uczenia. Zauważmy, że

$$\frac{\partial Q(w_i)}{\partial w_i} = \frac{\partial Q(w_i)}{\partial s} \cdot \frac{\partial s}{\partial w_i}. \tag{6.26}$$

Ponieważ s jest funkcją liniową względem wektora wag, więc możemy napisać

$$\frac{\partial s}{\partial w_i} = x_i. \tag{6.27}$$

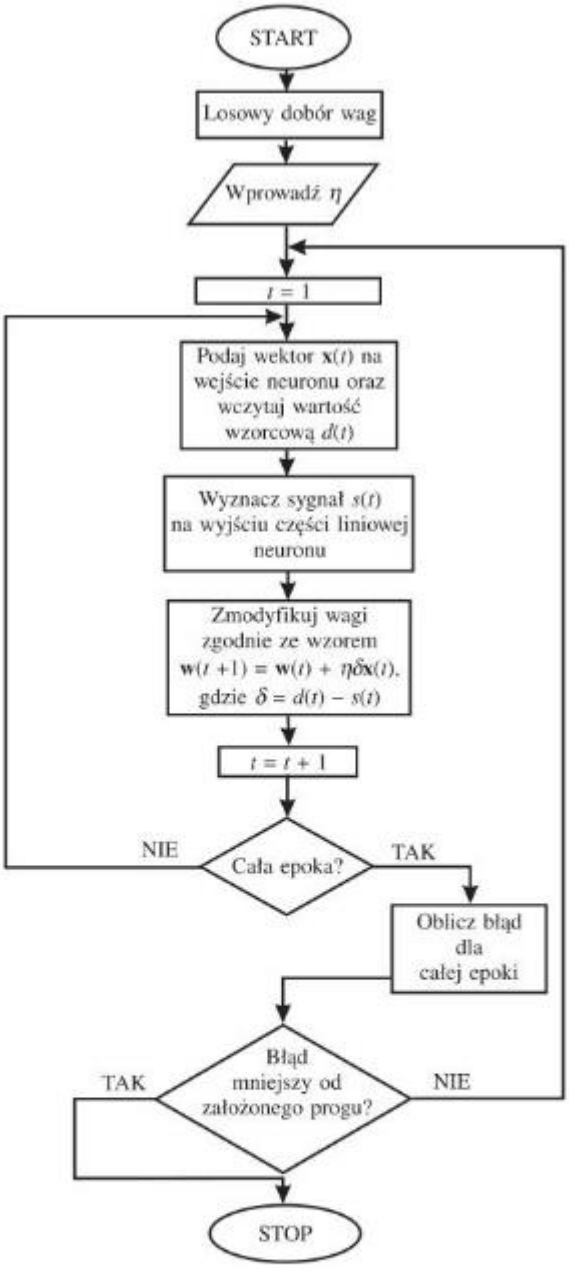
Ponadto

$$\frac{\partial Q(w_i)}{\partial s} = -(d - s). \tag{6.28}$$

Zatem zależność (6.25) przybiera postać

$$w_i(t+1) = w_i(t) + \eta \delta x_i, \tag{6.29}$$

gdzie $\delta = d - s$. Powyższa reguła nosi nazwę *reguły delta* (jest to szczególna postać tej reguły, ponieważ nie uwzględnia ona funkcji aktywacji neuronu). Na rysunku 6.9 przedstawiono schemat blokowy algorytmu uczenia neuronu typu Adaline za pomocą tej reguły.



Rys. 6.9. Schemat blokowy algorytmu uczenia neuronu typu Adaline

Neurony typu Adaline można również uczyć za pomocą rekurencyjnej metody najmniejszych kwadratów (ang. *Recursive Least Squares* — RLS). Jako miarę błędu przyjmuje się następujące wyrażenie:

$$Q(t) = \sum_{k=1}^{t} \lambda^{t-k} \varepsilon^2(k) = \sum_{k=1}^{t} \lambda^{t-k} |d(k) - \mathbf{x}^T(k)\mathbf{w}(t)|^2, \tag{6.30}$$

w którym λ jest współczynnikiem zapominania (ang. *forgetting factor*) wybieranym z przedziału $[0, 1]$. Zauważmy, że poprzednie błędy mają mniejszy wpływ na wartość wyrażenia (6.30). Obliczając gradient miary błędu, otrzymujemy następującą zależność:

$$\begin{aligned}\frac{\partial Q(t)}{\partial \mathbf{w}(t)} &= \frac{\partial \sum_{k=1}^{t} \lambda^{t-k} \varepsilon^2}{\partial \mathbf{w}(t)} \\ &= \frac{\partial \sum_{k=1}^{t} \lambda^{t-k} |d(k) - \mathbf{x}^T(k)\mathbf{w}(t)|^2}{\partial \mathbf{w}(t)} \\ &= -2 \sum_{k=1}^{t} \lambda^{t-k} |d(k) - \mathbf{x}^T(k)\mathbf{w}(t)| \mathbf{x}(k).\end{aligned} \tag{6.31}$$

Optymalne wartości wag powinny spełniać tzw. *równanie normalne*

$$\sum_{k=1}^{t} \lambda^{t-k} |d(k) - \mathbf{x}^T(k)\mathbf{w}(t)| \mathbf{x}(k) = \mathbf{0}. \tag{6.32}$$

Równanie (6.32) można przedstawić w postaci

$$\mathbf{r}(t) = \mathbf{R}(t)\mathbf{w}(t), \tag{6.33}$$

gdzie

$$\mathbf{R}(t) = \sum_{k=1}^{t} \lambda^{t-k} \mathbf{x}(k)\mathbf{x}^T(k) \tag{6.34}$$

jest $n \times n$-wymiarową macierzą autokorelacji, oraz

$$\mathbf{r}(t) = \sum_{k=1}^{t} \lambda^{t-k} d(k)\mathbf{x}(k) \tag{6.35}$$

jest $n \times 1$-wymiarowym wektorem korelacji wzajemnej sygnału wejściowego i sygnału wzorcowego. Zakładamy, że sygnały te są realizacjami stacjonarnych procesów stochastycznych. Rozwiązanie równania normalnego (6.33) przybiera postać

$$\mathbf{w}(t) = \mathbf{R}^{-1}(t)\mathbf{r}(t), \tag{6.36}$$

jeżeli $\det \mathbf{R}(t) \neq 0$. Zastosujemy teraz algorytm RLS w celu uniknięcia operacji odwracania macierzy w równaniu (6.36) i rozwiążemy równanie normalne (6.33) w sposób rekurencyjny.

Zauważmy, że macierz $\mathbf{R}(t)$ oraz wektor $\mathbf{r}(t)$ można przedstawić w postaci

$$\mathbf{R}(t) = \lambda \mathbf{R}(t-1) + \mathbf{x}(t)\mathbf{x}^T(t) \tag{6.37}$$

oraz

$$\mathbf{r}(t) = \lambda \mathbf{r}(t-1) + \mathbf{x}(t)d(t). \tag{6.38}$$

Zastosujemy teraz lemat o odwrotności macierzy. Niech $\mathbf{A}$ i $\mathbf{B}$ będą dodatnio określonymi $n \times n$-wymiarowymi macierzami takimi, że

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}^{-1}\mathbf{C}^T, \tag{6.39}$$

gdzie $\mathbf{D}$ jest dodatnio określoną $m \times m$-wymiarową macierzą, natomiast $\mathbf{C}$ jest $n \times m$-wymiarową macierzą. Wówczas

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{BC}(\mathbf{D} + \mathbf{C}^T\mathbf{BC})^{-1}\mathbf{C}^T\mathbf{B}. \tag{6.40}$$

Porównując wzory (6.40) i (6.37), otrzymujemy

$$\begin{aligned} \mathbf{A} &= \mathbf{R}(t), \\ \mathbf{B}^{-1} &= \lambda\mathbf{R}(t-1), \\ \mathbf{C} &= \mathbf{x}(t), \\ \mathbf{D} &= 1. \end{aligned} \tag{6.41}$$

Zatem

$$\mathbf{P}(t) = \lambda^{-1}[\mathbf{I} - \mathbf{g}(t)\mathbf{x}^T(t)]\mathbf{P}(t-1), \tag{6.42}$$

gdzie

$$\mathbf{P}(t) = \mathbf{R}^{-1}(t) \tag{6.43}$$

oraz

$$\mathbf{g}(t) = \frac{\mathbf{P}(t-1)\mathbf{x}(t)}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)}. \tag{6.44}$$

Wykażemy prawdziwość następującego równania:

$$\mathbf{g}(t) = \mathbf{P}(t)\mathbf{x}(t). \tag{6.45}$$

W wyniku prostych operacji algebraicznych otrzymujemy

$$\begin{aligned} \mathbf{g}(t) &= \frac{\mathbf{P}(t-1)\mathbf{x}(t)}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)} \\ &= \frac{\lambda^{-1}[\lambda\mathbf{P}(t-1)\mathbf{x}(t) + \mathbf{P}(t-1)\mathbf{x}(t)\mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)]}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)} \\ &\quad - \frac{\lambda^{-1}[\mathbf{P}(t-1)\mathbf{x}(t)\mathbf{x}^T(t) + \mathbf{P}(t-1)\mathbf{x}(t)]}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)} \\ &= \frac{\lambda^{-1}\left[\left(\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)\right)\mathbf{I}\right]\mathbf{P}(t-1)\mathbf{x}(t)}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)} \\ &\quad - \frac{\lambda^{-1}[\mathbf{P}(t-1)\mathbf{x}(t)\mathbf{x}^T(t)]\mathbf{P}(t-1)\mathbf{x}(t)}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)} \\ &= \lambda^{-1}\left[\mathbf{I} - \frac{\mathbf{P}(t-1)\mathbf{x}(t)\mathbf{x}^T(t)}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)}\right]\mathbf{P}(t-1)\mathbf{x}(t) \\ &= \lambda^{-1}[\mathbf{I} - \mathbf{g}(t)\mathbf{x}^T(t)]\mathbf{P}(t-1)\mathbf{x}(t) = \mathbf{P}(t)\mathbf{x}(t). \end{aligned} \tag{6.46}$$

Z zależności (6.38) oraz (6.36) wynika, że

$$\mathbf{w}(t) = \mathbf{R}^{-1}(t)\mathbf{r}(t) = \lambda\mathbf{P}(t)\mathbf{r}(t-1) + \mathbf{P}(t)\mathbf{x}(t)d(t). \tag{6.47}$$

Z równania (6.42) oraz (6.47) otrzymujemy

$$\mathbf{w}(t) = [\mathbf{I} - \mathbf{g}(t)\mathbf{x}^T(t)]\mathbf{P}(t-1)\mathbf{r}(t-1) + \mathbf{P}(t)\mathbf{x}(t)d(t). \tag{6.48}$$

Konsekwencją zależności (6.38) i (6.36) jest następujący związek:

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \mathbf{g}(t)\mathbf{x}^T(t)\mathbf{w}(t-1) + \mathbf{P}(t)\mathbf{x}(t)d(t). \tag{6.49}$$

Uwzględniając związek (6.45) w zależności (6.49), otrzymujemy następującą rekursję:

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \mathbf{g}(t)[d(t) - \mathbf{x}^T(t)\mathbf{w}(t-1)]. \tag{6.50}$$

W konsekwencji algorytm RLS zastosowany do uczenia neuronu typu Adaline przybiera następującą postać:

$$\varepsilon(t) = d(t) - \mathbf{x}^T(t)\mathbf{w}(t-1) = d(t) - y(t), \tag{6.51}$$

$$\mathbf{g}(t) = \frac{\mathbf{P}(t-1)\mathbf{x}(t)}{\lambda + \mathbf{x}^T(t)\mathbf{P}(t-1)\mathbf{x}(t)}, \tag{6.52}$$

$$\mathbf{P}(t) = \lambda^{-1}[\mathbf{I} - \mathbf{g}(t)\mathbf{x}^T(t)]\mathbf{P}(t-1), \tag{6.53}$$

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \mathbf{g}(t)\varepsilon(t). \tag{6.54}$$

Jako wartości początkowe zazwyczaj przyjmuje się

$$\mathbf{P}(0) = \gamma\mathbf{I}, \qquad \gamma > 0, \tag{6.55}$$

gdzie γ jest stałą, natomiast $\mathbf{I}$ jest macierzą jednostkową.