Sieci neuronowe: perspektywa systemowa

1. Co to są sieci neuronowe

Bibliografia

[0-1] James A. Anderson and Edward Rosenfeld (Eds.) *Neurocomputing. Foundations of Research*, MIT Press, Cambridge, Massachusetts 1988

[0-2] W.S. McCulloch and W.H. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophysics*, vol. **5**, 115–133, 1943

[0-3] J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. of the National Acad. of Sciences of the USA*, vol. **79**, 2554–2558, 1982

[0-4] J.J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. of the National Acad. of Sciences of the U.S.A.*, vol. **81**, 3088-3092, 1984

[0-5] M.L. Minsky and S.A. Papert, *Perceptrons*, MIT Press, Cambridge, Ma., 1969; Expanded Edition, MIT Press, Cambridge, Ma., 1988

[0-6] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain", *Psychological Review*, vol. 65, No. 6, pp. 386–408, 1958. Reprinted in [0-1], Ch. 8, pp. 89-114, 1988

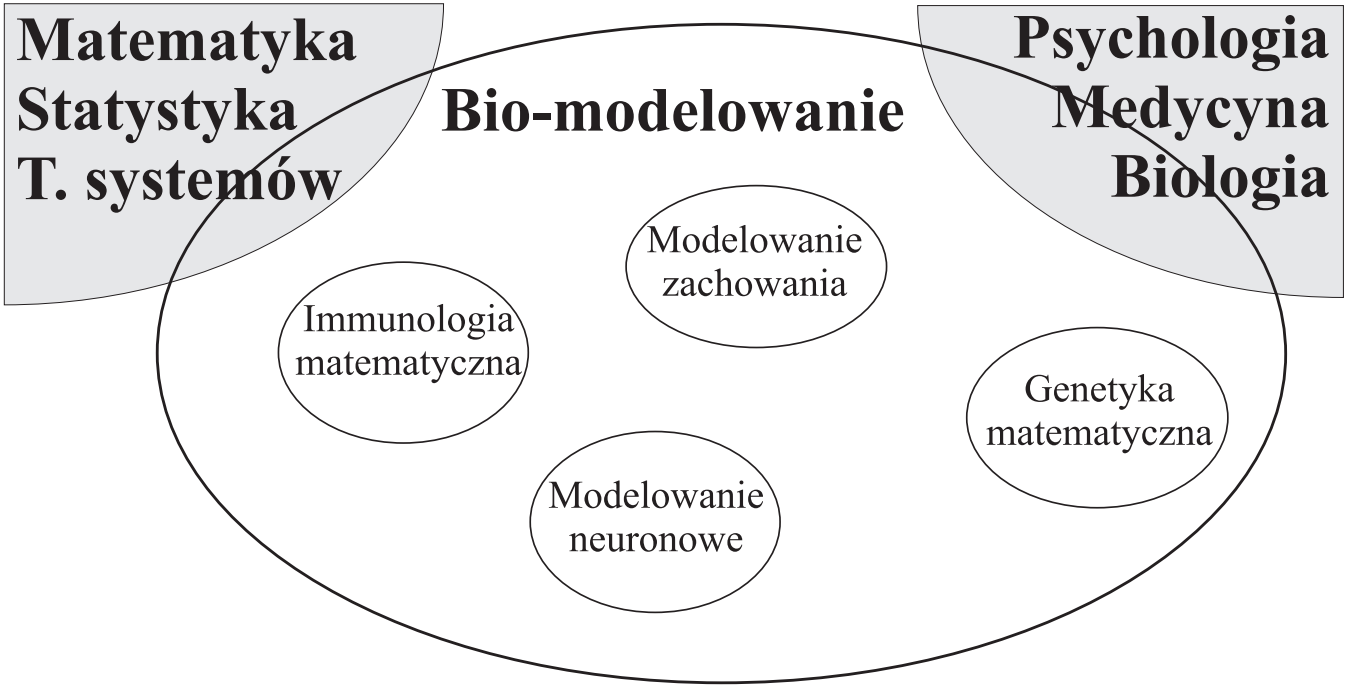
[0-7] F. Rosenblatt, *Principles of Neurodynamics*, Spartan Books, Washington, D.C., 1962

[0-8] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, D.E. Rumelhart and J.L. McClelland, Eds., vol. 1, Chap. 8, MIT Press, Cambridge, Ma., 1986

[0-9] P. Werbos, *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*, Wiley, New York 1994

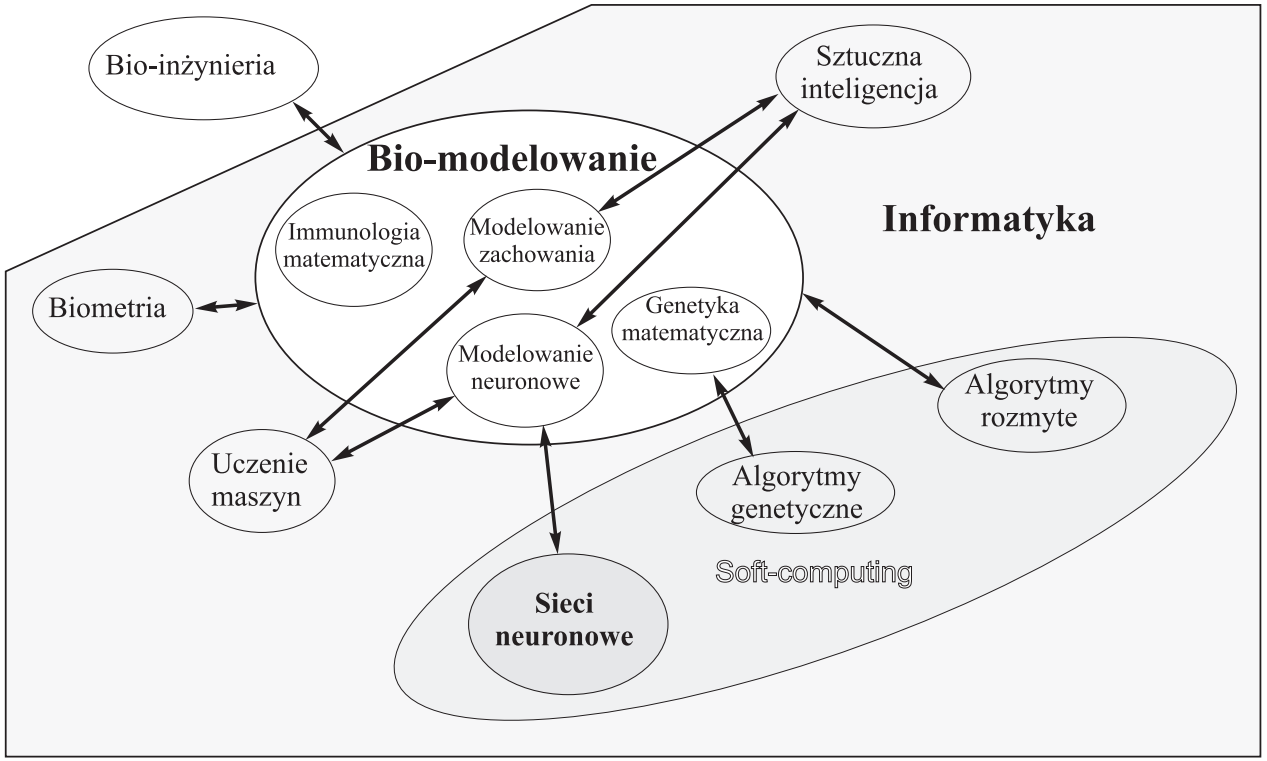
[0-10] B. Widrow and M.E. Hoff, "Adaptive switching circuits," *1960 IRE WESCON Convention Record*, vol. **4**, pp. 96–104, 1960

Bio-modelowanie



- Cele: *poznawcze*
- Zastosowania: biologia, medycyna, farmacja, psychologia

Badania inspirowane przez bio-modelowanie



- Cele: *specyficzne dla dziedziny zastosowań* (*nie-biologiczne*)
- Modyfikacja modeli biologicznych; *wyniki nie kopiują biologii*
- Potrzeba "ponawiania inspiracji" biologicznych

Sieci neuronowe - zgrubna charakterystyka

- Modularność
- Równoległość
- Nieliniowość
- Adaptacja do zmian środowiska
- Tolerancja na uszkodzenia (stopniowa utrata jakości)
- Nadmiar parametrów
- Własności aproksymacyjne

Kamienie milowe rozwoju sieci neuronowych

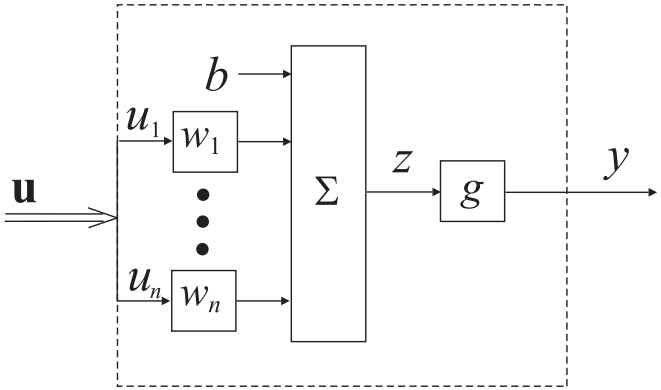
- **McCulloch i Pitts 1943** ▶*model neuronu* $y(t+1) = \mathbf{1}(\mathbf{w}^T\mathbf{u} - \vartheta)$
- **Rosenblatt 1958** (Cornell U., Ithaca, NY) ▶*perceptron*
- **Widrow 1960** (Stanford U., Stanford, CA) ▶*Adaline*
- **Minsky 1969** (MIT, Cambridge, MA) ▶*perceptron + sztuczna inteligencja*
- "dekada spokoju"
  - **Kohonen 1972** (Helsinki U. of Techn.) ▶*liniowa pamięć asocjacyjna*
  - **Werbos 1974** (Harvard U. Ph.D., Cambridge, MA) ▶*propagacja zwrotna*
  - **Grossberg 1983** (Boston U., Boston, MA) ▶*stabilność sieci dynamicznych*
- **Hopfield 1982** (Princeton U., Princeton, NJ) ▶*pamięć – stabilność sieci*
- **Rummelhart, Hinton, Williams 1986** (grupa PDP, MIT, Cambridge, MA) ▶*uczenie sieci poprzez propagację zwrotną*

Zastosowania sieci neuronowych

- Filtracja adaptacyjna, DSP
- Prognozowanie
- Klasyfikacja i rozpoznawanie (mowa, pismo)
- Biometria
- Wojskowość: rozpoznanie, identyfikacja celu
- Sterowanie, robotyka, systemy eksperckie, decyzyjne
- Ekonomia, Zarządzanie
- Fizyka, Astronomia
- Biologia (modelowanie DNA)
- Telekomunikacja
- Przemysł

2. Standardowe elementy sieci neuronowych

Neuron



afiniczna transformacja wejścia $\mathbf{u}$

$$z = b + \sum_{i=1}^{n} w_i\, u_i = b + \mathbf{w}^T \mathbf{u}$$

wagi $\mathbf{w} = [w_1 \ \cdots \ w_n]^T$

obciążenie b, *próg* $\vartheta = -b$

równanie neuronu

$$y = g\left(b + \sum_{i=1}^{n} w_i\, u_i\right) = g(b + \mathbf{w}^T \mathbf{u})$$

funkcja aktywacji $g : \mathbb{R} \mapsto \mathbb{R}$

obciążenie jako waga

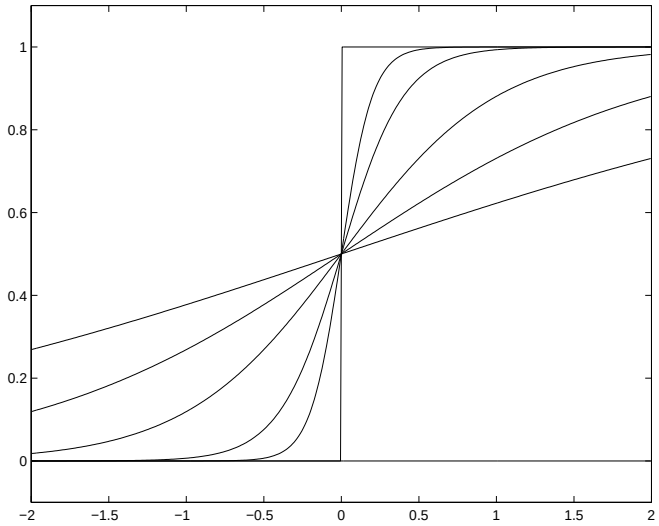
wagi rozszerzone o wagę $w_0 = b$

wejścia rozszerzone o wejście $u_0 = 1$

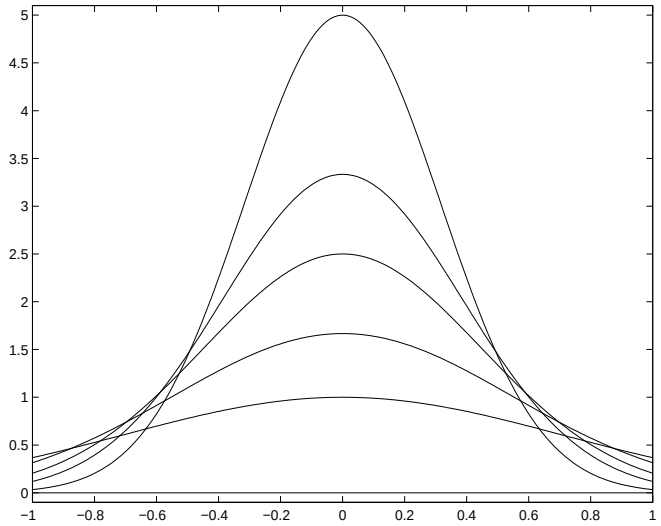
$$y = g\left(\sum_{i=0}^{n} w_i\, u_i\right) = g(\overline{\mathbf{w}}^T \overline{\mathbf{u}})$$

Funkcje aktywacji

- dyskretne f. aktywacji *(twarde nieliniowości)*: f. znaku, f. skokowa
- ciągłe f. aktywacji *(miękkie nieliniowości)*: f. sigmoidalne, sigmoid, f. gaussowska

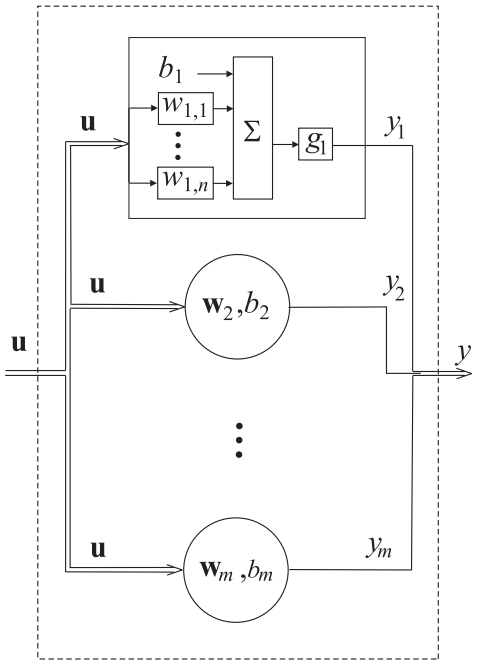


sigmoid



funkcje radialne gaussowskie

Warstwa



- struktura warstwy

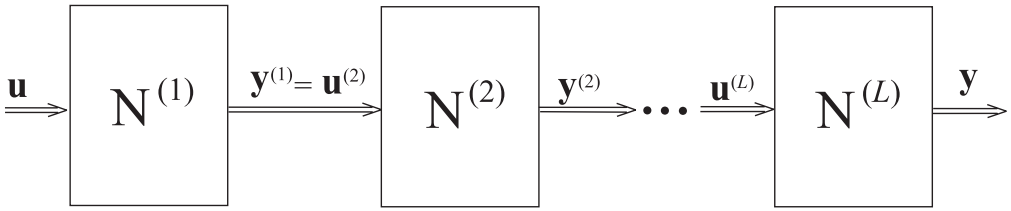
$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \mathbf{g}(\mathbf{b} + \mathbf{W}\mathbf{u}), \qquad \mathbf{g}(\mathbf{z}) = \begin{bmatrix} g_1(z_1) \\ \vdots \\ g_m(z_m) \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}, \qquad \mathbf{W} = \begin{bmatrix} \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_m^T \end{bmatrix} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \vdots \\ w_{m,1} & \cdots & w_{m,n} \end{bmatrix}$$

- zapis rozszerzony

$$\mathbf{y} = \mathbf{g}(\overline{\mathbf{W}}\,\overline{\mathbf{u}}), \qquad \overline{\mathbf{u}} = \begin{bmatrix} 1 \\ \mathbf{u} \end{bmatrix}, \qquad \overline{\mathbf{W}} = [\mathbf{b} \;\; \mathbf{W}] = \begin{bmatrix} b_1 & \mathbf{w}_1^T \\ \vdots & \\ b_m & \mathbf{w}_m^T \end{bmatrix}$$

Perceptron wielowarstwowy



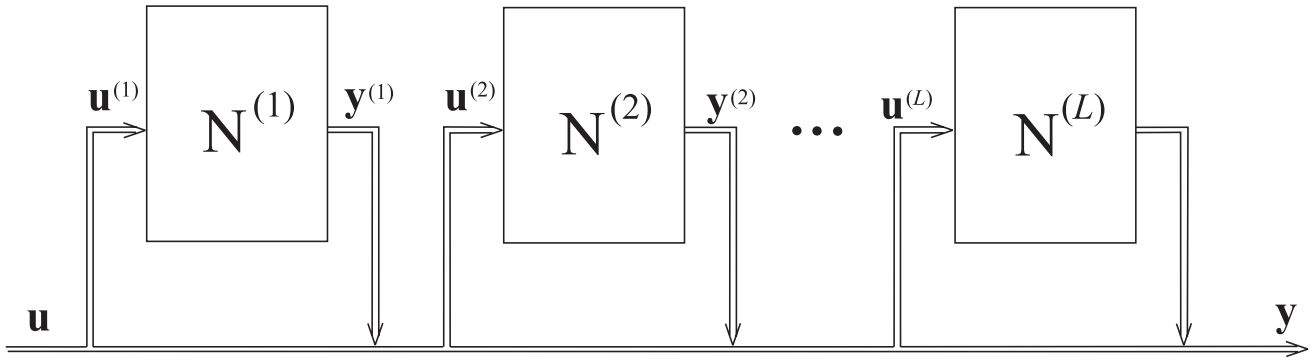
w a r s t w y u k r y t e warstwa wyjściowa

$$\mathbf{u}^{(1)} = \mathbf{u}, \qquad \mathbf{u}^{(k)} = \mathbf{y}^{(k-1)}, \quad k = 2, \ldots, L$$

$$\mathbf{y}^{(k)} = \mathbf{N}^{(k)}(\mathbf{u}^{(k)}), \quad k = 1, \ldots, L$$

$$\mathbf{y} = \mathbf{y}^{(L)}$$

Sieć bez sprzężeń

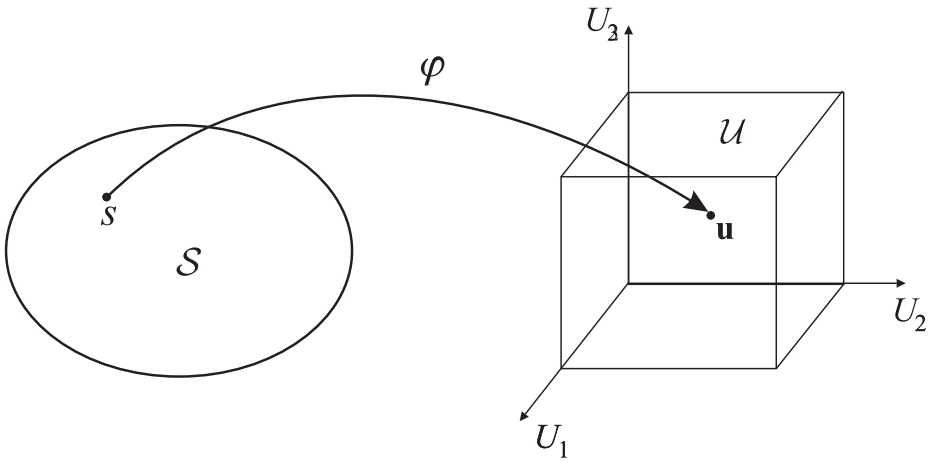


$$\mathbf{u}^{(1)} = \mathbf{u}, \qquad \mathbf{u}^{(k)} = \begin{bmatrix} \mathbf{u} \\ \mathbf{y}^{(1)} \\ \vdots \\ \mathbf{y}^{(k-1)} \end{bmatrix}, \quad k = 2, \ldots, L$$

$$\mathbf{y}^{(k)} = \mathbf{N}^{(k)}(\mathbf{u}^{(k)}), \quad k = 1, \ldots, L$$

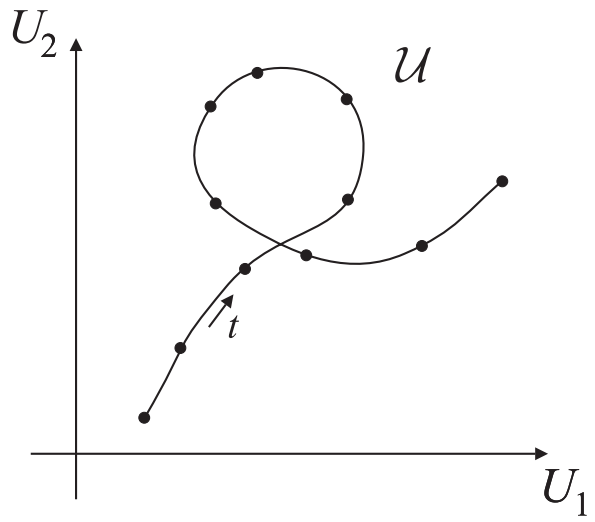
$$\mathbf{y} = \mathbf{y}^{(L)}$$

Sygnały



zbiór wejść $\mathcal{U} \subset \mathbb{R}^n$

- n – liczba punktów obrazu, liczba czujników
- punkty obrazu binarne: $\mathcal{U}$ = zbiór wierzchołków kostki
  - ▶ reprezentacja asymetryczna
  - ▶ reprezentacja symetryczna
- wejścia ograniczone: $\mathcal{U}$ = kostka w $\mathbb{R}^n$



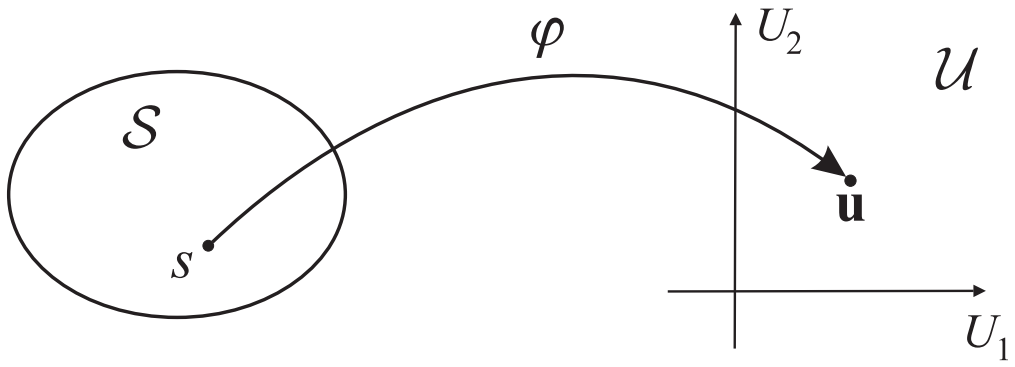
sygnał: funkcja $T \mapsto \mathcal{U}$

Adaptacja wag

- *sieć statyczna* $\mathbf{y}(t) = \mathbf{N}\big(\mathbf{u}(t); \mathbf{w}\big)$
- uczenie
  - przykład treningowy $\big(\mathbf{u}(k), \mathbf{y}^*(k)\big)$; $\mathbf{y}^*(k)$ – *wyjście pożądane*
  - ciąg trenujący (ciąg uczący) $\big\{\big(\mathbf{u}(k), \mathbf{y}^*(k)\big),\ i = 1, \ldots, N\big\}$
  - modyfikacja wag $\mathbf{w}(t+1) = h(\mathbf{w}(t), \mathbf{u}(t))$
- sieć z uczeniem: $\mathbf{y}(t) = \mathbf{N}\big(\mathbf{u}(t);\ h\big(\mathbf{w}(t-1), \mathbf{u}(t-1)\big)$
  układ dynamiczny $\mathbf{y}(t) = \mathbf{N}\{\mathbf{u}(s),\ s \le t\}$
- struktura neuronu/warstwy/sieci mogą być bardziej skomplikowane
- *sieć dynamiczna* $\mathbf{y}(t) = \mathbf{N}\{\mathbf{u}(s),\ s \le t\}$
- "szybka dynamika" (nie związana z uczeniem) i "wolna dynamika" (uczenie)

3. Zagadnienia klasyfikacji

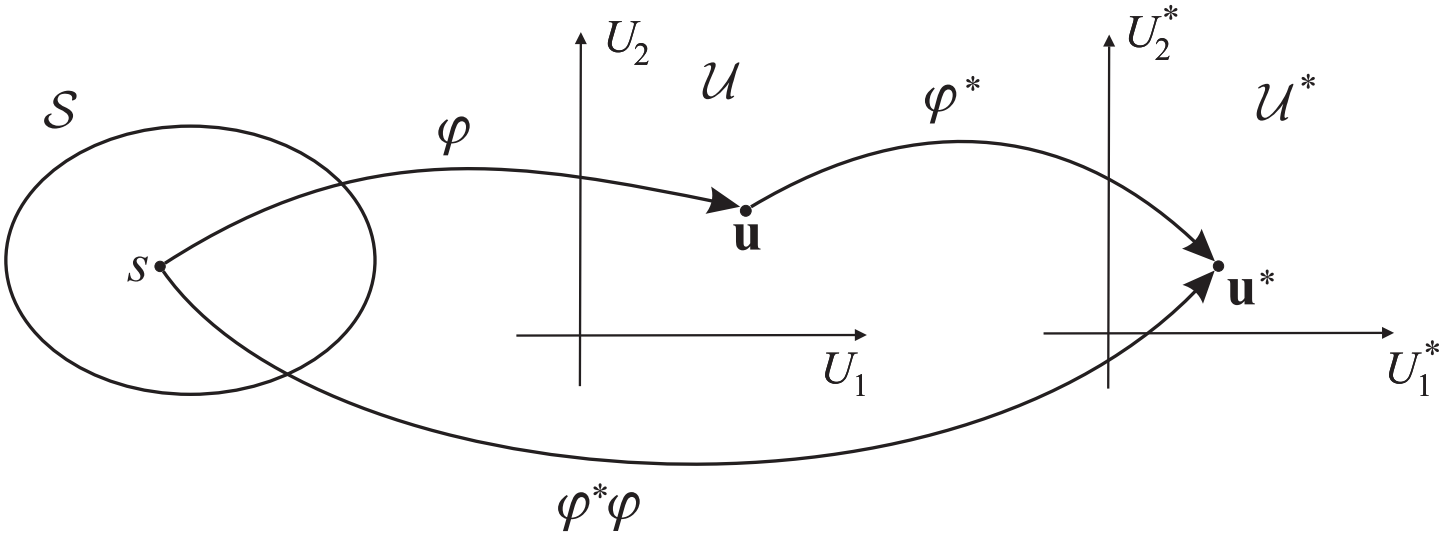
Obiekty i obrazy



- zbiór *obiektów* $\mathcal{S}$
- *funkcja obserwacji* $\varphi : \mathcal{S} \mapsto \mathcal{U}$

- zbiór *obrazów* $\mathcal{U}$
- zwykle $\mathcal{U} \subset \mathbb{R}^n$
  n – liczba czujników, liczba punktów obrazu
- *obraz* $\mathbf{u} = [u_1 \quad \cdots \quad u_n]^T \in \mathcal{U}$
- *obraz binarny*, *obraz czarno-biały*
  $u_i \in \{L, H\}$; (zwykle $\{0, 1\}$)

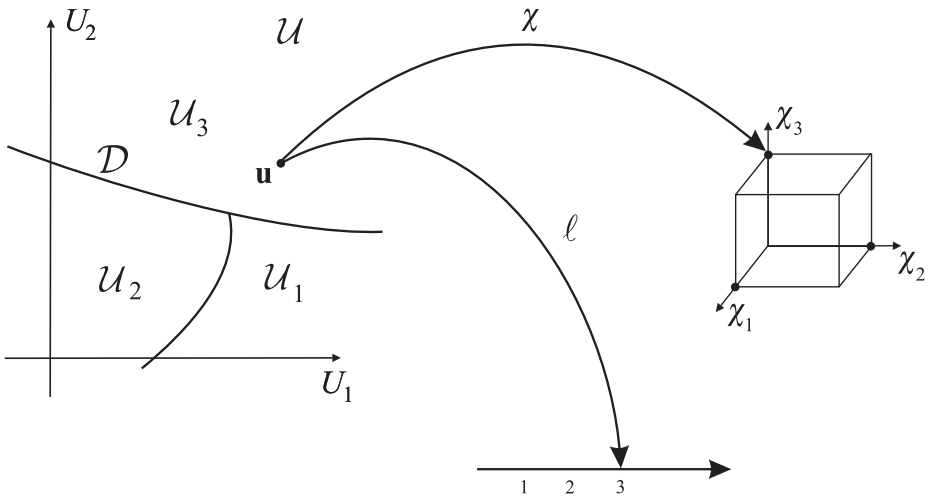
Cechy



- zbiór wektorów *cech* $\mathcal{U}^* \subset \mathbb{R}^{n*}$
- *funkcja cech* $\varphi^* : \mathcal{U} \mapsto \mathcal{U}^*$
- zmodyfikowana funkcja obserwacji $\varphi^*\varphi : \mathcal{S} \mapsto \mathcal{U}^*$

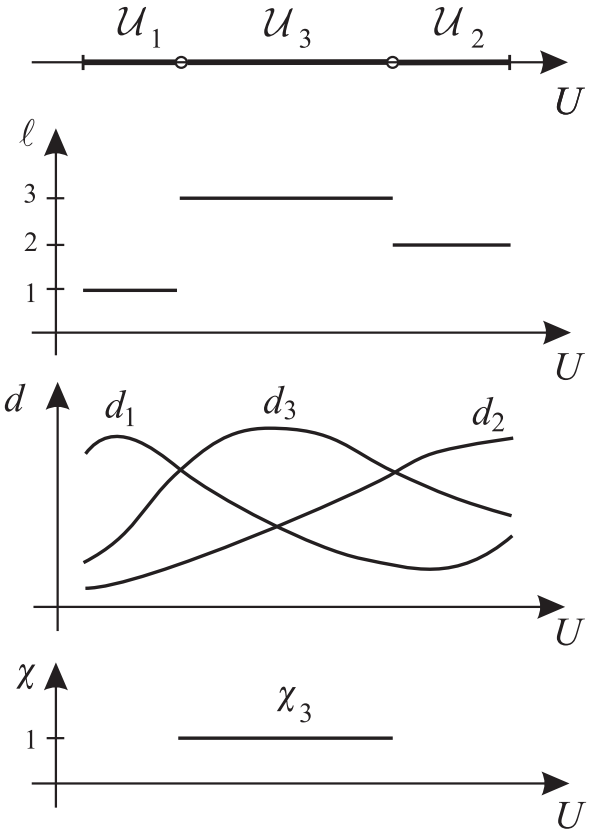
Klasyfikacja obrazów

- *funkcja klasyfikująca*
  $\ell(\mathbf{u}) = k \iff \mathbf{u} \in \mathcal{U}_k$
- *funkcje przynależności*
  do klasy $\mathcal{U}_k$, $k = 1, \ldots, c$
  $$\chi_k(\mathbf{u}) = 1 \iff \mathbf{u} \in \mathcal{U}_k$$
  $$\chi_k(\mathbf{u}) = 0 \iff \mathbf{u} \notin \mathcal{U}_k$$
  $$\boldsymbol{\chi}(\mathbf{u}) = \begin{bmatrix} \chi_1(\mathbf{u}) \\ \vdots \\ \chi_c(\mathbf{u}) \end{bmatrix}$$
- *zbiór decyzyjny*
  $\mathcal{D} = \sum_k \partial \mathcal{U}_k$

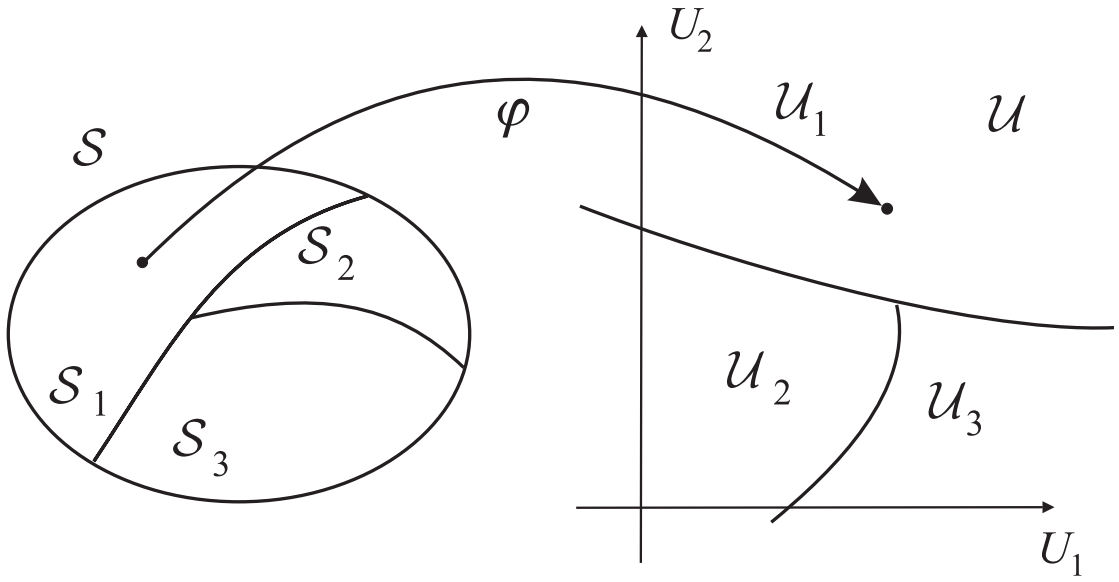


Funkcje decyzyjne

- funkcja klasyfikująca ℓ
- *funkcje decyzyjne*
  $\ell(\mathbf{u}) = k \iff d_k(\mathbf{u}) > d_j(\mathbf{u})$ dla $j \neq k$
- funkcje przynależności χ_k są funkcjami decyzyjnymi

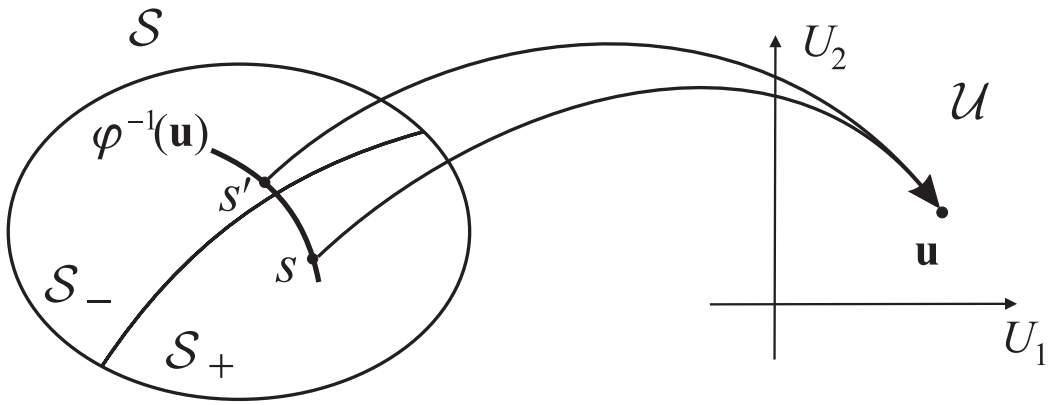


*Klasyfikacja obiektów a klasyfikacja obrazów



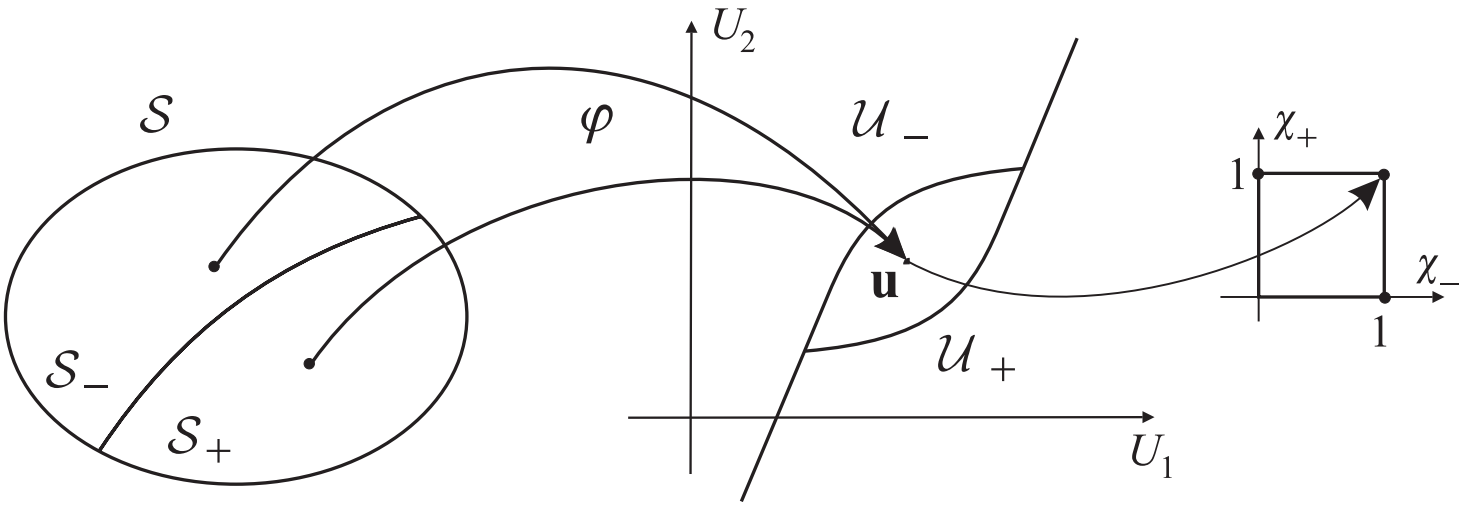
- klasy obiektów $\mathcal{S}_i$
- *dziedziczone* klasy obrazów $\mathcal{U}_i = \varphi(\mathcal{S}_i)$
- *dziedziczenie jednoznaczne* $\mathcal{U}_i \cap \mathcal{U}_j = \emptyset$ for $i \neq j$
- klasyfikacja obrazów a klasyfikacja obiektów: $\mathbf{u} \in \mathcal{U}_k \iff s \in \mathcal{S}_k$

*Problem niejednoznaczności obserwacji



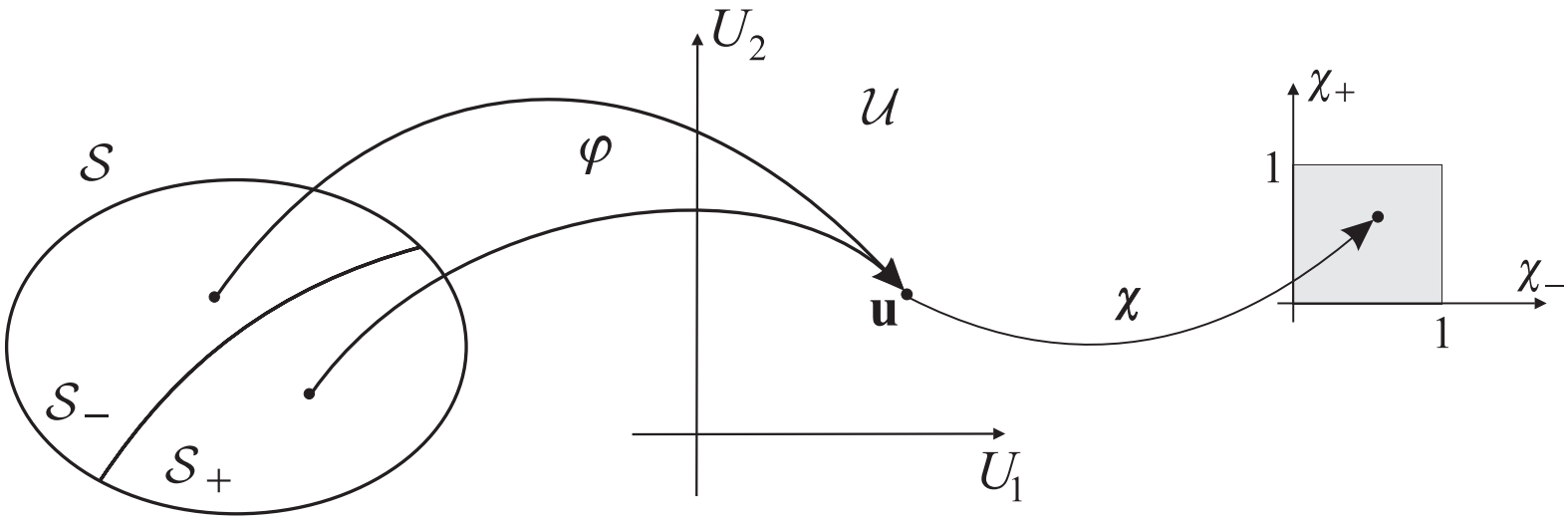
- obraz $\mathbf{u}$
- funkcja obserwacji **może nie być odwracalna**: *niejednoznaczności obserwacji*
- *przeciwobraz* $\varphi^{-1}(\mathbf{u})$ obrazu $\mathbf{u}$: zbiór obiektów mających **ten sam obraz** $\mathbf{u}$
  $s \in \varphi^{-1}(\mathbf{u}) \iff \varphi(s) = \mathbf{u}$
- klasy dziedziczone rozłączne: obiekty mające ten sam obraz należą do tej samej klasy
- klasy dziedziczone mogą nie być rozłączne

*Niejednoznaczne dziedziczenie klas



- niejednoznaczność, jeżeli $\mathcal{U}_i \cap \mathcal{U}_j \neq \emptyset$
- χ przyjmuje wartości z (niezerowych) wierzchołków kostki jednostkowej

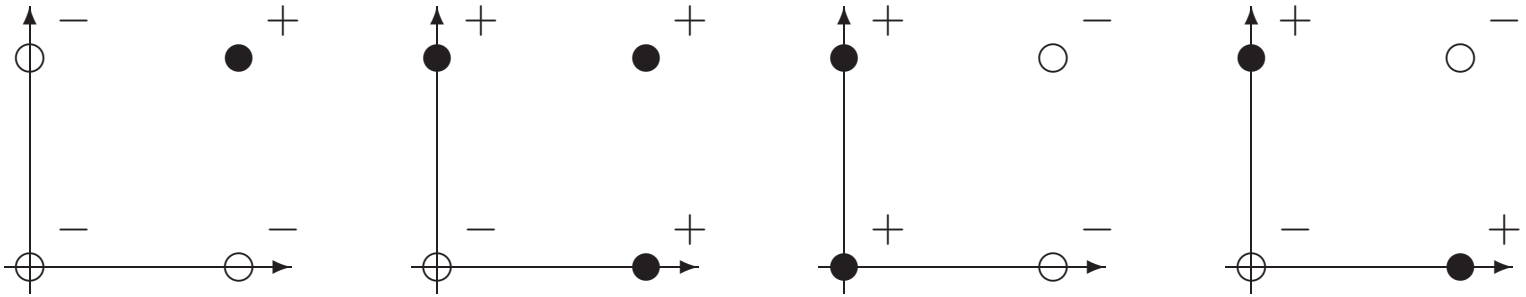
*Dziedziczenie probabilistyczne



- rozkład obiektów $\mathcal{P}$
- funkcja przynależności w $\mathcal{U}$: $\chi_k(\mathbf{u}) = \mathcal{P}\{s \in \mathcal{S}_k\}$
- χ przyjmuje wartości z kostki jednostkowej

Klasyfikacja binarna a funkcje logiczne

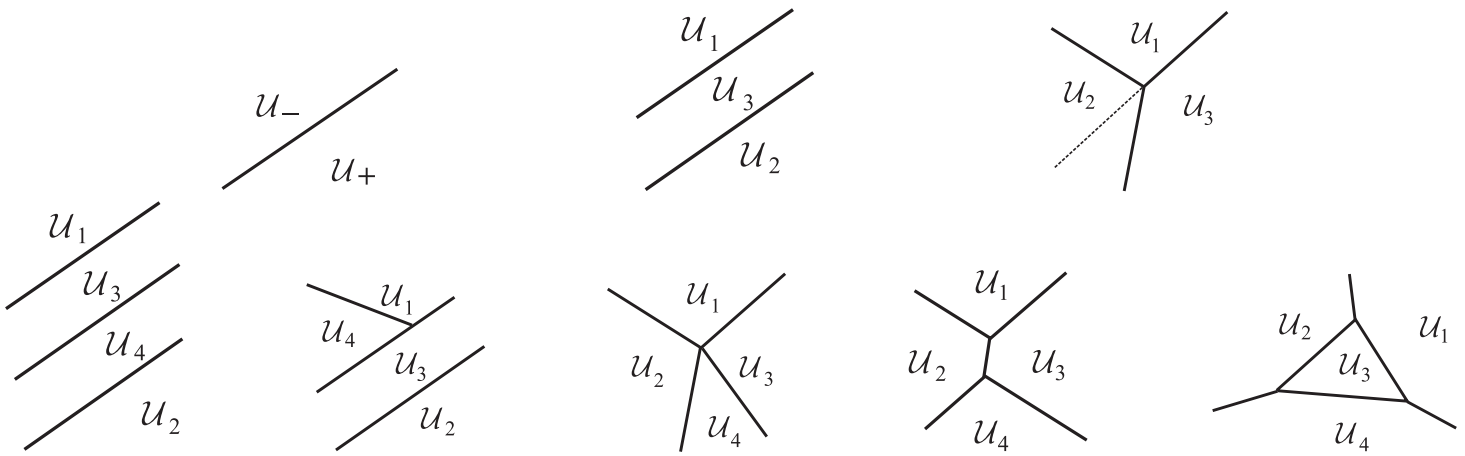
- $N = 4$ (4 obrazy)
- $\mathcal{U} \subset \mathbb{R}^2$ (obrazy złożone z 2 punktów)
- $c = 2$ (klasyfikacje binarne)



u_1 AND u_2 u_1 OR u_2 NOT u_1 u_1 XOR u_2

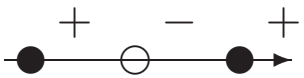
Klasyfikacja liniowa

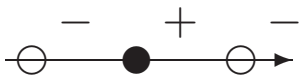
- *hiperpłaszczyzna* $\mathcal{H}(\mathbf{w}, b) = \{\mathbf{u} \in \mathbb{R}^n : \mathbf{w}^T\mathbf{u} + b = 0\}$
  $\mathbf{w}$ – *wektor normalny*, b — *przesunięcie*
- *podprzestrzeń dodatnia* $\mathcal{U}_+(\mathbf{w}, b) = \{\mathbf{u} \in \mathbb{R}^n : \mathbf{w}^T\mathbf{u} + b > 0\}$ względem hiperpłaszczyzny $\mathcal{H}(\mathbf{w}, b)$
  podprzestrzeń ujemna $\mathcal{U}_-(\mathbf{w}, b)$
- *klasyfikacja liniowa*: dowolne dwie klasy można rozdzielić hiperpłaszczyzną



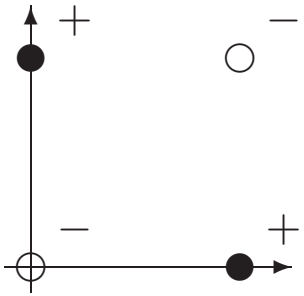
Najprostsze klasyfikacje nieliniowe

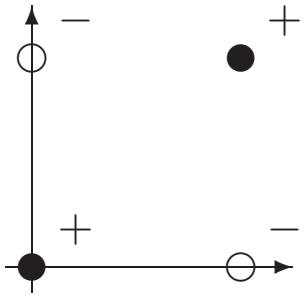
3 obrazy binarne w $\mathbb{R}$, 2 klasy





4 obrazy binarne w $\mathbb{R}^2$, 2 klasy





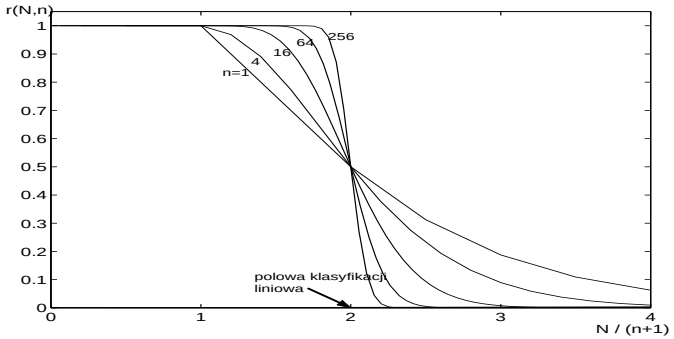
Liczba klasyfikacji liniowych N obrazów w $\mathbb{R}^n$

- liczba klasyfikacji liniowych $\leq L(N,n)$ ($<$ dla szczególnych usytuowań obrazów)

$$L(N,n) = \begin{cases} 2^N & \text{for } N \leq n+1 \quad (\text{“mało obrazów”}) \\ 2\sum_{i=0}^{n} \binom{N-1}{i} & \text{for } N \geq n+1 \end{cases}$$

- liczba klasyfikacji liniowych / liczba klasyfikacji binarnych $\leq$

$$r(N,n) = \frac{L(N,n)}{B(N)} = \begin{cases} 1 & \text{dla } \frac{N}{n+1} \leq 1 \\ 2^{1-N}\sum_{i=0}^{n} \binom{N-1}{i} & \text{dla } \frac{N}{n+1} \geq 1 \end{cases}$$



$n = 1, 4, 16, 64, 256$

- dla co najwyżej $N_0 = n+1$ obrazów w $\mathbb{R}^n$ wszystkie klasyfikacje mogą być liniowe
- dla $N_v = 2(n+1)$ obrazów w $\mathbb{R}^n$ co najwyżej połowa klasyfikacji jest liniowa

  N_v: *pojemność klasyfikacji liniowych* w $\mathbb{R}^n$

Liczba klasyfikacji liniowych - przykład

	$N=2$ ••	$N=3$ •••	$N=4$ ••••	$N=5$ •••• •	$N=6$ •••• ••	$N=7$ •••• •••	$N=8$ •••• ••••	wszystkie li-niowe	pojemność
$B(N)$	4	8	16	32	64	128	256		
$\mathbb{R}^1$	*4	6	•8	10	12	14	16	2	4
$\mathbb{R}^2$	4	*8	14	22	•32	44	58	3	6
$\mathbb{R}^3$	4	8	*16	30	52	84	•128	4	8
$\mathbb{R}^4$	4	8	16	*32	62	114	198	5	10
$\mathbb{R}^5$	4	8	16	32	*64	126	240	6	12
	lewa + górna lewa $L(N,n) = L(N-1,n) + L(N-1,n-1)$								

Liczba binarnych klasyfikacji liniowych

liczba punktów obrazu n	max. liczba obrazów binarnych $N = 2^n$	liczba klasyfikacji binarnych $B(N) = 2^N$	liczba klasyfikacji liniowych $L(N,n)$	ułamek klasyfikacji liniowych $r = L/B$
2	4	16	14	0.875
3	8	256	128	0.500
4	16	65536	3882	0.059
5	32	$4.3\ 10^9$	412736	$9.6\ 10^{-5}$
8	256	10^{77}	10^{15}	10^{-63}
16	65536	10^{19728}	10^{64}	10^{-19664}

Klasyfikatory liniowe

- *afiniczne funkcje decyzyjne* $d_k(\mathbf{u}) = b_k + \mathbf{w}_k^T\mathbf{u}, \;\; k = 1, \ldots, c$

$$\mathbf{d}(\mathbf{u}) = \begin{bmatrix} b_1 + \mathbf{w}_1^T\mathbf{u} \\ \vdots \\ b_c + \mathbf{w}_c^T\mathbf{u} \end{bmatrix} = \mathbf{b} + \mathbf{W}\,\mathbf{u} = \overline{\mathbf{W}}\overline{\mathbf{u}}$$

- afiniczne funkcje decyzyjne $\Rightarrow$ klasy liniowo rozdzielne
  klasy liniowo rozdzielne $\Rightarrow$ istnieją afiniczne funkcje decyzyjne

- funkcje decyzyjne są porównywane parami (liczbę wierszy $\overline{\mathbf{W}}$ można zmniejszyć o jeden)

$$b_i + \mathbf{w}_i^T\mathbf{u} > b_j + \mathbf{w}_j^T\mathbf{u} \quad \Leftrightarrow \quad b + \mathbf{w}^T\mathbf{u} > 0$$

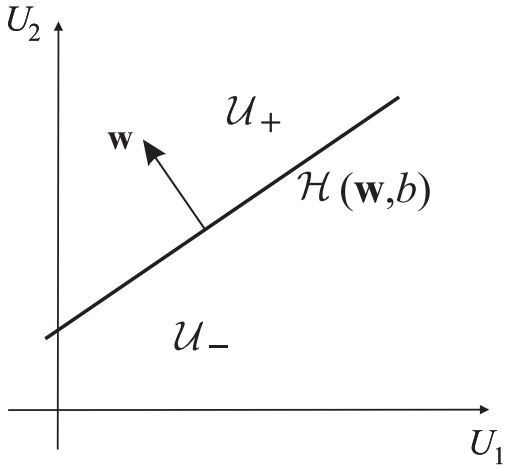
  gdzie $\mathbf{w} = \mathbf{w}_i - \mathbf{w}_j$, $b = b_i - b_j$

Liniowe klasyfikatory binarne

- binarny klasyfikator liniowy $\mathbf{d}(\mathbf{u}) = \begin{bmatrix} b_+ + \mathbf{w}_+^T \mathbf{u} \\ b_- + \mathbf{w}_-^T \mathbf{u} \end{bmatrix}$

$$\ell(\mathbf{u}) = \begin{cases} -1 & \text{jeśli } \mathbf{w}^T\mathbf{u} + b < 0 \\ 1 & \text{jeśli } \mathbf{w}^T\mathbf{u} + b > 0 \end{cases}$$

gdzie $\mathbf{w} = \mathbf{w}_+ - \mathbf{w}_-$, $b = b_+ - b_-$



- zmodyfikowana funkcja decyzyjna – porównywana z zerem

$$d(\mathbf{u}) = b + \mathbf{w}^T\mathbf{u}$$

- hiperpłaszczyzna klasyfikująca $\mathcal{H}(\mathbf{w}, b) = \{\mathbf{u} : d(\mathbf{u}) = 0\}$

*Kontekst probabilistyczny

- jeśli $\varphi(\mathcal{S}_i) \cap \varphi(\mathcal{S}_j) \neq \emptyset$ to klasyfikacja nie jest jednoznaczna.
- dodatkowo znana miara probabilistyczne $\mathcal{P}$ na $\mathcal{S}$
  - rozkład obrazów w ramach każdej klasy (ciągły)

$$f_{\mathbf{u}\,|\,k}(\mathbf{z})\,d\mathbf{z} = \mathcal{P}(\mathbf{z} \le \mathbf{u} \le \mathbf{z} + d\mathbf{z}\,|\,s \in \mathcal{S}_k)$$

  - *rozkład a priori klas* $\{\pi_k,\ k = 1, \ldots, c\}$, gdzie $\pi_k = \mathcal{P}\{s \in \mathcal{S}_k\}$
- znany obraz $\mathbf{u}$; *rozkład a posteriori klas*

$$\pi_{k\,|\,\mathbf{u}} = \mathcal{P}\{s \in \mathcal{S}_k\,|\,\mathbf{u}\}$$

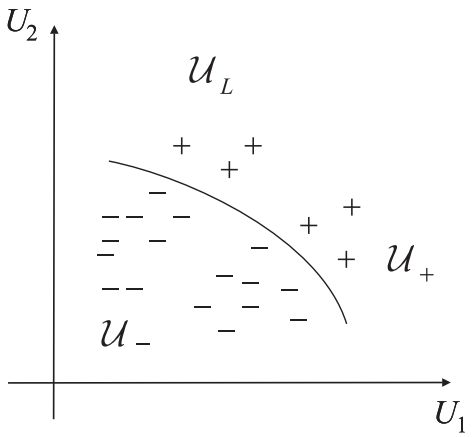
$$(\textit{reguła Bayesa}) = \frac{\pi_k\, f_{\mathbf{u}\,|\,k}(\mathbf{u})}{f_{\mathbf{u}}(\mathbf{u})} = \frac{\pi_k\, f_{\mathbf{u}\,|\,k}(\mathbf{u})}{\sum_{i=1}^{c} \pi_i\, f_{\mathbf{u}\,|\,i}(\mathbf{u})}$$

- klasyfikacja w zbiorze obrazów
  - funkcja decyzyjna: $d_k(\mathbf{u}) = \pi_{k\,|\,\mathbf{u}}$ *klasyfikator bayesowski*
  - równoważna f.d: $d_k(\mathbf{u}) = \pi_k f_{\mathbf{u}\,|\,k}(\mathbf{u})$
- dla dyskretnego rozkładu obrazów: $p_{j\,|\,k} = \mathcal{P}(\mathbf{u} = j\,|\,s \in \mathcal{S}_k)$

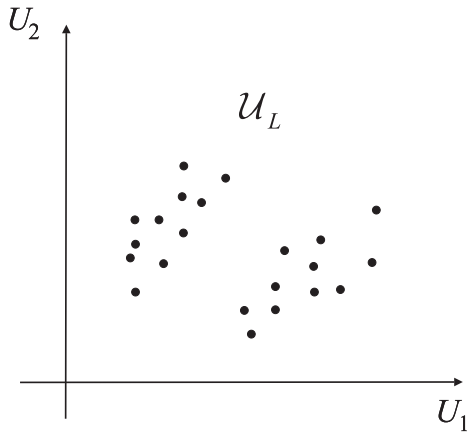
$$\pi_{k\,|\,j} = \frac{\pi_k\, p_{j\,|\,k}}{p_j} = \frac{\pi_k\, p_{j\,|\,k}}{\sum_{i=1}^{c} \pi_i\, p_{j\,|\,i}}$$

Budowa klasyfikatorów — dostępna informacja

- liczba klas c znana
- *przykład uczący* $(\mathbf{u}_i, \ell(\mathbf{u}_i))$
- *zbiór uczący* (*zbiór trenujący*)
  $\mathcal{U}_L = \{(\mathbf{u}_i, \ell(\mathbf{u}_i)),\ i = 1, \ldots, N\}$



- liczba klas c znana lub nie
- przykład uczący $\mathbf{u}_i$ (klasa nieznana)
- zbiór uczący
  $\mathcal{U}_L = \{\mathbf{u}_i,\ i = 1, \ldots, N\}$



4. Perceptron Rosenblatta

Krótka historia perceptronu Rosenblatta

- Frank Rosenblatt (Cornell U.) 1958
  "uniwersalna maszyna do rozpoznawania i klasyfikacji wzorców"

  "sztuczny mózg", histeryczna reakcja mediów

- hardware: "Mark I" (1960)

  matryca 20×20 fotodiod, każda losowo łączona z 40 neuronami warstwy ukrytej

  512 neuronów warstwy ukrytej

  8 neuronów wyjściowych, silniki sterujące potencjometrami wag

- Rosenblatt "Principles of Neurodynamics", 1962

- Minsky i Papert (MIT) "Perceptrons", 1969

  "... większość publikacji jest bez wartości naukowej"

  perceptron jednowarstwowy – tylko klasyfikacja liniowa

- pozostało: 'connectionism', architektury warstwowe, obliczenia równoległe, uczenie wag

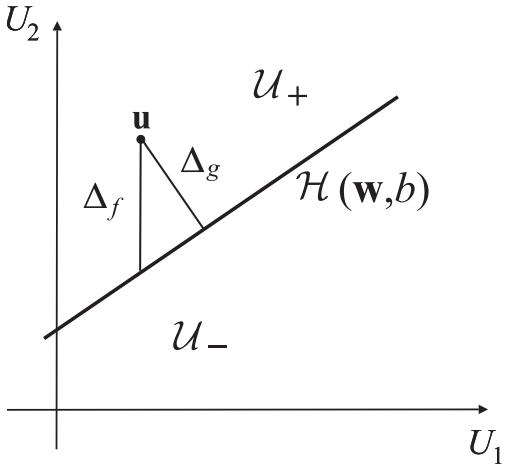
Binarne klasyfikatory liniowe

- funkcja decyzyjna $d(\mathbf{u}) = \mathbf{w}^T\mathbf{u} + b$
- funkcja klasyfikująca

$$\ell(\mathbf{u}) = \begin{cases} -1 & \text{jeśli } \mathbf{w}^T\mathbf{u} + b < 0 \\ 1 & \text{jeśli } \mathbf{w}^T\mathbf{u} + b > 0 \end{cases}$$

- *margines funkcyjny obrazu* $\mathbf{u}$ względem hiperpłaszczyzny $\mathcal{H}(\mathbf{w}, b)$

$$\Delta_f(\mathbf{u}) = (\mathbf{w}^T\mathbf{u}+b)\,\ell(\mathbf{u}) \begin{cases} > 0 & \text{poprawna klasyfikacja } \mathbf{u} \\ < 0 & \text{błędna klasyfikacja } \mathbf{u} \end{cases}$$

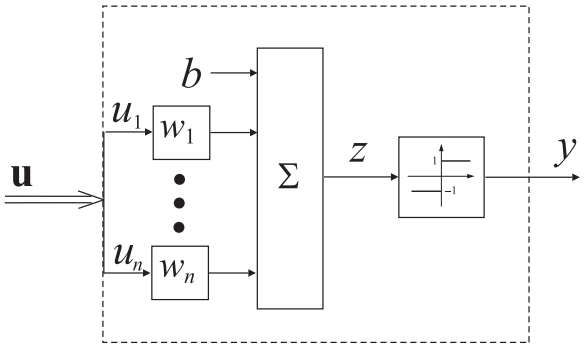


(przyjęto $w_2 = -1$)

- *margines geometryczny obrazu* $\mathbf{u}$ względem hiperpłaszczyzny $\mathcal{H}(\mathbf{w}, b)$ odległość opatrzona znakiem "$-$" przy błędnej klasyfikacji

$$\Delta_g(\mathbf{u}) = \frac{\Delta_f(\mathbf{u})}{\|\mathbf{w}\|} = \begin{cases} d\big(\mathbf{u}, \mathcal{H}(\mathbf{w}, b)\big) & \text{poprawna klasyfikacja } \mathbf{u} \\ -d\big(\mathbf{u}, \mathcal{H}(\mathbf{w}, b)\big) & \text{błędna klasyfikacja } \mathbf{u} \end{cases}$$

Struktura perceptronu Rosenblatta

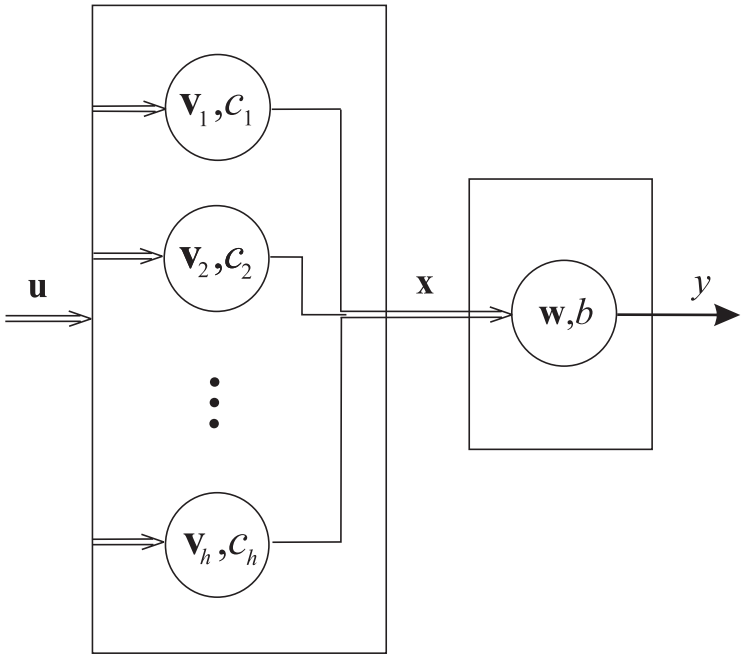


jednowarstwowy perceptron Rosenblatta

neuron dwuwartościowy

(*aktywny - nieaktywny*)

typowa funkcja aktywacji: sign



dwuwarstwowy perceptron Rosenblatta

wejścia: obrazy $\mathbf{u} \in \mathbb{R}^n$ (*warstwa n sensorów*)

warstwa ukryta (*asocjacyjna*): **wagi stałe**

warstwa wyjściowa (*odpowiedzi*): **wagi adaptowane**

Perceptron Rosenblatta a klasyfikacja

- perceptron jako funkcja klasyfikująca (wejście: obrazy $\in \mathbb{R}^n$, wyjście: $\{-1, 1\}$)
- *problem reprezentacji*: czy rodzina funkcji wytwarzanych przez perceptron Rosenblatta jest wystarczająco bogata, by rozwiązać dowolny problem klasyfikacji ?
- *perceptron jednowarstwowy*: dla dowolnego zbioru obrazów jednowarstwowy perceptron Rosenblatta może realizować **dowolne klasyfikacje liniowe** binarne tych obrazów
- *perceptron dwuwarstwowy*: dla **zadanego skończonego zbioru** obrazów istnieje dwuwarstwowy perceptron Rosenblatta, którego wagi warstwy wyjściowej dobierać tak, by realizować **dowolne klasyfikacje** binarne tych obrazów

Perceptron jednowarstwowy: uczenie

- prezentacja obrazów ze *zbioru uczącego* $\mathcal{U}_L$:

  ciąg uczący $\big\{\big(\mathbf{u}(t), y^*(t)\big),\ t = 1, \ldots, N\big\}$

  gdzie $y^*(t) = \ell(\mathbf{u}(t))$ — *wyjście pożądane*

- *epoka* — jednokrotna prezentacja wszystkich N obrazów ze zbioru treningowego w losowej kolejności

- *modyfikacja wag* $\quad \mathbf{w}(t+1) = \mathbf{w}(t) + \Delta\mathbf{w}(t)$

  dowolne wagi początkowe $\mathbf{w}(0)$, *korekcja* wag w przypadku błędnej klasyfikacji

$$\Delta\mathbf{w} = \begin{cases} 0 & \text{jeżeli } (\mathbf{w}^T\mathbf{u} + b)\, y^* > 0 \ \text{ (klasyfikacja poprawna)} \\ y^*\mathbf{u} & \text{jeżeli } (\mathbf{w}^T\mathbf{u} + b)\, y^* \le 0 \ \text{ (klasyfikacja błędna)} \end{cases}$$
$$= \mathbf{u}\ \mathrm{sign}\big(y^* - y\big)$$

- po *skończonej liczbie korekcji* sieć klasyfikuje prawidłowo

- wagi prawidłowo klasyfikujące *nie są jednoznaczne*

Modyfikacje algorytmu uczenia

- zróżnicowanie *współczynników szybkości uczenia* μ:
  korekcja w przypadku błędnej klasyfikacji $\left(y^*(\mathbf{w}^T\mathbf{u}+b)<0\right)$

$$\Delta\mathbf{w} = \mu\,\mathbf{u}\,\ell(\mathbf{u})$$

$$\Delta b = \mu\,R^2\,\ell(\mathbf{u})$$

  gdzie $R=\max_{\mathcal{U}_L}\|\mathbf{u}\|$

- tw. [Novikoff] dla liniowo rozdzielnego zbioru uczącego o hiperpłaszczyźnie decyzyjnej $\mathcal{H}(\mathbf{w}^*,b^*)$ *liczba korekcji* jest nie większa niż $\frac{4R^2}{\Delta_g^2}$ gdzie Δ_g oznacza *margines geometryczny hiperpłaszczyzny decyzyjnej*

$$\Delta_g = \inf_{\mathbf{u}\in\mathcal{U}_L} d\big(\mathbf{u},\mathcal{H}(\mathbf{w}^*,b^*)\big)$$

- współczynnik uczenia może zmieniać się w czasie $0<\mu_{\min}<\mu(t)<\mu_{\max}$
- *korekcja bezwzględna* — powtarzanie korekcji dla każdego $\mathbf{u}_i$ aż do poprawnej klasyfikacji $\mathbf{u}_i$

Funkcja kosztu dla perceptronu Rosenblatta

- obrazy źle klasyfikowane przez hiperpłaszczyznę decyzyjną $\mathcal{H}(\mathbf{w}, b)$

$$\overline{\mathcal{U}}_L(\mathbf{w}, b) = \{\mathbf{u} \in \mathcal{U}_L(\mathbf{w}, b) : \ell(\mathbf{u})(b + \mathbf{w}^T\mathbf{u}) < 0\}$$

- koszt błędnej klasyfikacji

$$Q(\mathbf{w}, b) = \sum_{\mathbf{u}_i \in \overline{\mathcal{U}}_L(\mathbf{w},b)} \Delta_f(\mathbf{w}, b) = -\sum_{\mathbf{u}_i \in \mathcal{U}_L(\mathbf{w},b)} (b + \mathbf{w}^T\mathbf{u}_i)\ \mathrm{sign}(y_i^* - y_i)$$

- $Q(\overline{\mathbf{w}})$ różniczkowalna z wyjątkiem N hiperpłaszczyzn $\{(\mathbf{w}, b) :\ \mathbf{u}_i^T\mathbf{w} + b = 0\}$ w $\mathbb{R}^{n+1}$
- gradient $\dfrac{dQ(\overline{\mathbf{w}})}{d\overline{\mathbf{w}}} = -\sum_{\mathbf{u}_i \in \mathcal{U}_L} \overline{\mathbf{u}}_i\ \mathrm{sign}(y_i^* - y_i)$,
- metoda największego spadku z *obliczeniem gradientu raz na epokę* T_k

$$\overline{\mathbf{w}}(t + N) = \overline{\mathbf{w}}(t) + \mu \frac{dQ(\overline{\mathbf{w}})}{d\overline{\mathbf{w}}} = \overline{\mathbf{w}}(t) + \mu \sum_{t \in T_k} \overline{\mathbf{u}}(t)\ \mathrm{sign}\big(y^*(t) - y(t)\big)$$

— *identyczna z algorytmem Rosenblatta z korekcją wag raz na epokę*

- metoda największego spadku z *chwilową aproksymacją gradientu*

$$\overline{\mathbf{w}}(t + 1) = \overline{\mathbf{w}}(t) + \mu \frac{d\widehat{Q}(\overline{\mathbf{w}})}{d\overline{\mathbf{w}}} = \overline{\mathbf{w}}(t) + \mu\overline{\mathbf{u}}(t)\ \mathrm{sign}\big(y^*(t) - y(t)\big)$$

— *identyczna z algorytmem Rosenblatta*

Algorytm 'kieszeniowy' Gallanta

- waga 'w kieszeni' $\overline{\mathbf{w}}^G(t)$

  odpowiada najdłuższej w chwili t serii identycznych wag

- waga optymalna $\mathbf{w}^*$

  minimalizuje prawdopodobieństwo błędnej klasyfikacji

- jeżeli

  $\mathcal{U}_L$ jest skończony

  wagi $\overline{\mathbf{w}}(t)$ są losowe niezależne o identycznych rozkładach,

  to

  - $\operatorname{Plim} \overline{\mathbf{w}}^G(t) = \overline{\mathbf{w}}^*$
  - $\mathcal{P}\{\overline{\mathbf{w}}^G(t) = \overline{\mathbf{w}}^*\} \underset{t\to\infty}{\longrightarrow} 1$

Istnienie uniwersalnego dwuwarstwowego perceptronu Rosenblatta

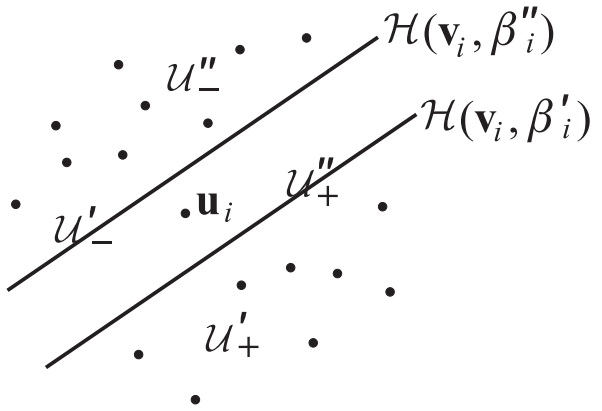
zbiór uczący $\mathcal{U}_L = \{\mathbf{u}_1, \ldots \mathbf{u}_N\}$, $\mathbf{u}_i \in \mathbb{R}^n$

warstwa ukryta: $2N$ *neuronów*

- dla każdego obrazu $\mathbf{u}_i$ określić hiperpłaszczyzny $\mathcal{H}(\mathbf{v}_i, \beta'_i)$, $\mathcal{H}(\mathbf{v}_i, \beta''_i)$ i neurony i', i'', aby
$$\mathbf{u}_i \in \mathcal{U}_-(\mathbf{v}_i, \beta'_i) \cap \mathcal{U}_+(\mathbf{v}_i, \beta''_i)$$
$$\mathbf{u}_j \in \mathcal{U}_+(\mathbf{v}_i, \beta'_i) \cup \mathcal{U}_-(\mathbf{v}_i, \beta''_i),\ j \neq i$$
- wyjścia neuronów ukrytych dla wejścia $\mathbf{u}_i$
$$x_{i'} = 0, \quad x_{i''} = 1$$
$$x_{j'} = x_{j''} \text{ dla } j \neq i$$

neuron wyjściowy

$$w_{i'} = 1,\ w_{i''} = -1 \text{ jeżeli } \mathbf{u}_i \in \mathcal{U}_-$$
$$w_{i'} = -1,\ w_{i''} = 1 \text{ jeżeli } \mathbf{u}_i \in \mathcal{U}_+$$
$$(w_{j'} x_{j'} + w_{j''} x_{j''} = 0 \text{ dla } j \neq i)$$
$$b = 0$$



- **jak wyznaczyć** $\mathbf{v}_i$, $\beta_{i'}$, $\beta_{i''}$
- konstrukcja **nieefektywna**: więcej neuronów ukrytych niż obrazów treningowych
- **zła generalizacja** dla praktycznych problemów

Warstwa ukryta: kodowanie

- **kodowanie wybranych obrazów** — N wybranych obrazów, $h = 2N$ **wymaga znajomości obrazów — a zakłócenia ?**
- **kodowanie zupełne** — wszystkie obrazy binarne, $h = N \leq 2^n$ **nierealizowalnie duża warstwa ukryta** dla rzeczywistych zadań
- **kodowanie losowe** — warstwa ukryta o zadanej wielkości h wybrana losowo

  tylko $N \leq h + 1$ obrazów **klasyfikowanych poprawnie**

  tylko $N = 2(h + 1)$ obrazów (pojemność liniowa) klasyfikowanych **poprawnie z prawdopodobieństwem 0.5**
- **kodowanie adaptacyjne** — najlepsze, ale wymaga **adaptacji warstwy ukrytej**

Maszyny liniowe

- warstwa ukryta $\mathbf{x} = \mathbf{1}(\mathbf{V}\mathbf{u})$
- warstwa wyjściowa: $\mathbf{y} = \boldsymbol{\chi}(\mathbf{W}\mathbf{x}) \in \mathbb{R}^c$

$$y_k = \begin{cases} 1 & \text{jeżeli } \mathbf{w}_k^T\mathbf{x} > \mathbf{w}_j^T\mathbf{x} \text{ dla wszystkich } j \neq k \\ 0 & \text{w przeciwnym przypadku} \end{cases}$$

"zwycięzca bierze wszystko" (WTA *ang. winner takes all*)

Maszyna liniowa a perceptron

- jednowarstwowa maszyna liniowa z dwoma wyjściami

$$y_1 = \begin{cases} 1 & \text{jeżeli } \overline{\mathbf{w}}_1 \overline{\mathbf{x}} > \overline{\mathbf{w}}_2 \overline{\mathbf{x}} \\ 0 & \text{w przeciwnym przypadku} \end{cases} = \mathbf{1}(\overline{\mathbf{w}}^T \overline{\mathbf{x}})$$

$$y_2 = 1 - y_1$$

  jest równoważna perceptronowi o wektorze wag wyjściowych $\overline{\mathbf{w}} = \overline{\mathbf{w}}_1 - \overline{\mathbf{w}}_2$ a zatem może realizować dowolną *binarną* klasyfikację liniową

- maszyna liniowa z c wyjściami jest równoważna perceptronowi utworzonemu dla $n\,c$ wymiarowej przestrzeni obrazów zawierającej Nc razy więcej obrazów.

- maszyna liniowa o c wyjściach może realizować dowolną liniową klasyfikację na c klas

Uczenie jednowarstwowej maszyny liniowej

- klasyfikacja liniowa skończonego zbioru obrazów na c klas
- każdy obraz pojawia wielokrotnie, w skończonych odstępach
- wagi początkowe dowolne
- algorytm korekcji wag $\Delta\overline{\mathbf{W}}(t) = \operatorname{sign}\big(\mathbf{y}^*(t) - \mathbf{y}(t)\big)\,\mathbf{u}^T(t)$

  dla obrazów klasyfikowanych poprawnie $\Delta\overline{\mathbf{W}}(t) = 0$

  dla obrazów klasyfikowanych błędnie

$$\Delta\overline{\mathbf{w}}_k(t) = \begin{cases} \overline{\mathbf{u}}(t) & \text{dla } k = \ell(\mathbf{u}(t)) \quad \textbf{(klasa właściwa)} \\ -\overline{\mathbf{u}}(t) & \text{dla } k = \text{WTA}\left(\overline{\mathbf{W}}\overline{\mathbf{u}}(t)\right) \quad \textbf{(klasa wygrywająca)} \\ 0 & \text{dla pozostałych } k \end{cases}$$

- wagi są prawidłowe po skończonej liczbie korekcji
- dla $c = 2$ algorytm identyczny z algorytmem Rosenblatta

5. Maszyny wektorów podpierających

Bibliografia

[0-1] N. Cristiannini, J Shave-Taylor, *An Introduction to Support Vector Machines* Cambridge University Press, Cambridge, UK 2000

[0-2] http://www.support-vector.net/references.html

Postać dualna perceptronu Rosenblatta

- optymalne wagi są liniową kombinacją obrazów treningowych

$$\mathbf{w} = \sum_t \mathbf{u}(t)\ \mathrm{sign}\big(y^*(t) - y(t)\big)$$

$$= \sum_{i=1}^{N} \alpha_i\, \mathbf{u}_i\, \ell(\mathbf{u}_i)$$

  α_i — liczba błędnych klasyfikacji i-tego obrazu (*siła obrazu*)

- reprezentacja ta nie jest jednoznaczna

- perceptron — *postać dualna*

$$\mathbf{N}(\mathbf{u}) = \mathrm{sign}\big(b + \mathbf{u}^T \sum_{i=1}^{N} \alpha_i\, \mathbf{u}_i\, \ell(\mathbf{u}_i)\big)$$

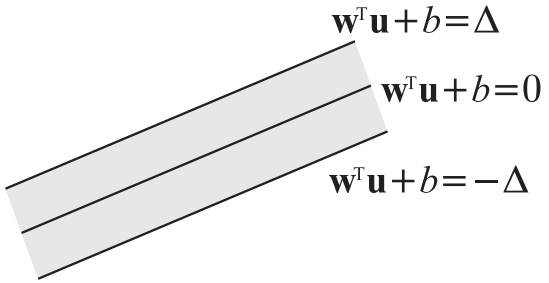
$$= \mathrm{sign}\big(b + \sum_{i=1}^{N} \alpha_i\, (\mathbf{u}_i^T \mathbf{u})\, \ell(\mathbf{u}_i)\big)$$

Region separujący

- $\mathcal{H}(\mathbf{w}, b)$ klasyfikuje poprawnie, jeśli dla pewnego $\Delta > 0$

$$\mathbf{u}_i \in \mathcal{U}_{L+} \Rightarrow b + \mathbf{w}^T \mathbf{u}_i \geq \Delta$$

$$\mathbf{u}_i \in \mathcal{U}_{L-} \Rightarrow b + \mathbf{w}^T \mathbf{u}_i \leq -\Delta$$



czyli $\big(b + \mathbf{w}^T \mathbf{u}_i\big)\, \ell(\mathbf{u}_i) \geq \Delta$ dla $\mathbf{u}_i \in \mathcal{U}_L$

- *margines funkcyjny* Δ_f hiperpłaszczyzny: największe możliwe Δ, tzn. dla co najmniej jednego $\mathbf{u}_i \in \mathcal{U}_L$

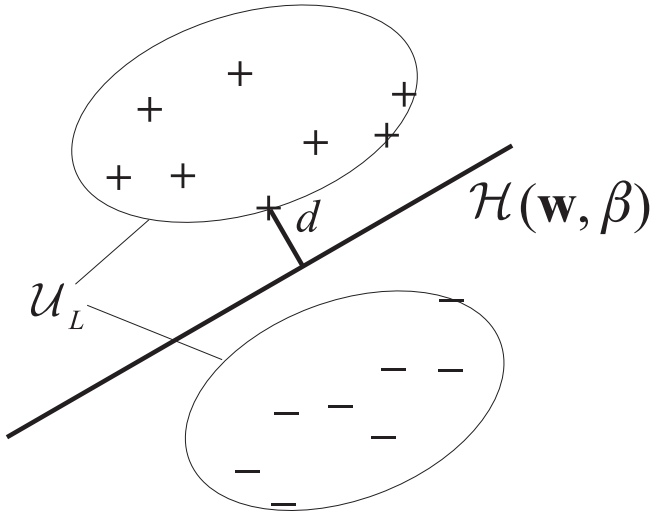
$$\big(b + \mathbf{w}^T \mathbf{u}_i\big)\, \ell(\mathbf{u}_i) = \Delta_f \quad (\textit{obrazy podpierające } \mathcal{U}_S)$$

- *region separujący* $\{\mathbf{u} \in \mathbb{R}^n : \ |b + \mathbf{w}^T \mathbf{u}| < \Delta_f\}$
- przypomnienie: odległość punktu $\mathbf{u}_0$ od hiperpłaszczyzny $\mathcal{H}(\mathbf{w}, b)$ jest równa $\frac{|\mathbf{w}^T \mathbf{u}_0 + b|}{\|\mathbf{w}\|}$, odległość hiperpłaszczyzn $\mathcal{H}(\mathbf{w}, b_1)$, $\mathcal{H}(\mathbf{w}, b_2)$ jest równa $\frac{|b_1 - b_2|}{\|\mathbf{w}\|}$
- szerokość regionu separującego $\frac{2\Delta_f}{\|\mathbf{w}\|}$
- *postać kanoniczna* $\Delta_f = 1$ (usunięta niejednoznaczność)

Optymalna hiperpłaszczyzna decyzyjna

- optymalna hiperpłaszczyzna decyzyjna $\mathcal{H}(\mathbf{w}^*, b^*)$:
  - **a**. bezbłędnie klasyfikuje obrazy uczące $\mathbf{u}_i \in \mathcal{U}_L$
    $\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) \ell(\mathbf{u}_i) \geq 1 \quad$ dla $\mathbf{u}_i \in \mathcal{U}_L$
  - **b**. maksymalizuje *margines geometryczny* tzn. odległość $d(\mathcal{U}_L, \mathcal{H}(\mathbf{w}, b))$
    $\Delta_g = d(\mathcal{U}_L, \mathcal{H}(\mathbf{w}^*, b^*)) = \frac{1}{\|\mathbf{w}^*\|}$
- klasyfikator maksymalizujący margines geometryczny

$$\mathrm{N}(\mathbf{u}) = \mathrm{sign}(\mathbf{w}^{*T}\mathbf{u} + b^*)$$



Przypomnienie: Minimalizacja z ograniczeniami (1)

- **problem pierwotny** znaleźć $\boldsymbol{\vartheta}^*$ minimalizujące funkcję $f(\boldsymbol{\vartheta})$, $\boldsymbol{\vartheta} \in \Theta \subseteq \mathbb{R}^n$, przy ograniczeniach

$$g_i(\boldsymbol{\vartheta}) \le 0, \qquad i = 1, \ldots, k \qquad (\mathbf{g}(\boldsymbol{\vartheta}) \le \mathbf{0})$$
$$h_j(\boldsymbol{\vartheta}) = 0, \qquad j = 1, \ldots, m \qquad (\mathbf{h}(\boldsymbol{\vartheta}) = \mathbf{0})$$

- *obszar dopuszczalny* $\mathcal{D} = \{\boldsymbol{\vartheta} \in \Theta : g(\boldsymbol{\vartheta}) \le 0,\ h(\boldsymbol{\vartheta}) = \mathbf{0}\}$
- rozwiązanie (globalne) problemu pierwotnego $\boldsymbol{\vartheta}^*$

$$f(\boldsymbol{\vartheta}^*) < f(\boldsymbol{\vartheta}) \quad \text{dla każdego} \quad \boldsymbol{\vartheta} \in \mathcal{D},\ \boldsymbol{\vartheta} \neq \boldsymbol{\vartheta}^*$$

- *ograniczenie aktywne*: $g_i(\boldsymbol{\vartheta}^*) = 0$, *ograniczenie nieaktywne*: $g_i(\boldsymbol{\vartheta}^*) < 0$
- *funkcja Lagrange'a*, mnożniki Lagrange'a

$$L(\boldsymbol{\vartheta}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\boldsymbol{\vartheta}) + \sum_{i=1}^{k} \alpha_i\, g_i(\boldsymbol{\vartheta}) + \sum_{j=1}^{m} \beta_j\, h_j(\boldsymbol{\vartheta})$$

Minimalizacja z ograniczeniami (2): Problem dualny

- **problem dualny**: znaleźć $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*$ maksymalizujące funkcję

$$L^*(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \inf_{\boldsymbol{\vartheta} \in \Theta} L(\boldsymbol{\vartheta}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

  przy ograniczeniach $\boldsymbol{\alpha} \geq \mathbf{0}$

- $f(\boldsymbol{\vartheta}) \geq L^*(\boldsymbol{\alpha}, \boldsymbol{\beta})$ dla każdego $\boldsymbol{\vartheta} \in \mathcal{D}$, $\boldsymbol{\alpha} \geq 0$, $\boldsymbol{\beta}$

- jeśli $f(\boldsymbol{\vartheta}^*) = L^*(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ gdzie $\boldsymbol{\alpha}^* \geq 0$, $\boldsymbol{\vartheta}^* \in \mathcal{D}$,
  to $\boldsymbol{\vartheta}^*$, $\left(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*\right)$ są rozwiązaniami problemu pierwotnego i dualnego,
  a ponadto $\boldsymbol{\alpha}^* \odot \mathbf{g}^* = \mathbf{0}$

- *luka dualności*: $\overline{L} = \inf_{\boldsymbol{\vartheta} \in \mathcal{D}} f(\boldsymbol{\vartheta}) - \sup_{\boldsymbol{\alpha} \geq 0,\, \boldsymbol{\beta}} L^*(\boldsymbol{\alpha}, \boldsymbol{\beta}) \geq 0$

- *punkt siodłowy*: luka dualności jest zerowa *wtedy i tylko wtedy* gdy $(\boldsymbol{\vartheta}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$
  jest punktem siodłowym funkcji Lagrange'a tzn. dla każdego $\boldsymbol{\vartheta} \in \Theta$, $\boldsymbol{\alpha} \geq 0$, $\boldsymbol{\beta}$

$$L(\boldsymbol{\vartheta}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \leq L(\boldsymbol{\vartheta}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \leq L(\boldsymbol{\vartheta}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$$

Minimalizacja z ograniczeniami (3): Warunki Kuhna-Tuckera

- jeżeli Θ jest zbiorem wypukłym i ograniczenia są afiniczne, to $\overline{L} = 0$
- *warunki Kuhna-Tuckera* dla funkcji f wypukłej i ciągłej wraz z pochodnymi na zbiorze wypukłym Θ i afinicznych funkcji g_i, h_j:
  $\boldsymbol{\vartheta}^*$ jest rozwiązaniem problemu pierwotnego wtedy i tylko wtedy, gdy istnieją $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*$ takie, że

$$\frac{\partial L}{\partial \boldsymbol{\vartheta}}\Big|_{\boldsymbol{\vartheta}^*,\boldsymbol{\alpha}^*,\boldsymbol{\beta}^*} = \mathbf{0} \qquad \text{(K-T 1)}$$

$$\frac{\partial L}{\partial \boldsymbol{\beta}}\Big|_{\boldsymbol{\vartheta}^*,\boldsymbol{\alpha}^*,\boldsymbol{\beta}^*} = \mathbf{0} \qquad \text{(K-T 2)}$$

$$\alpha_i^* \, g_i(\boldsymbol{\vartheta}^*) = 0, \qquad i = 1, \ldots, k \qquad \text{(K-T 3a)}$$

$$g_i(\boldsymbol{\vartheta}^*) \leq 0, \qquad i = 1, \ldots, k \qquad \text{(K-T 3b)}$$

$$\alpha_i^* \geq 0, \qquad i = 1, \ldots, k \qquad \text{(K-T 3c)}$$

- dla ograniczeń aktywnych $\alpha_i^* \geq 0$, dla ograniczeń nieaktywnych $\alpha_i^* = 0$

Maksymalizacja marginesu hiperpłaszczyzny decyzyjnej

- problem optymalizacji

  minimalizować $Q(\mathbf{w},\, b) = \frac{1}{2}\|\mathbf{w}\|^2$ *funkcja wypukła na zbiorze* $\mathbb{R}^{n+1}$

  przy ograniczeniach $\ell(\mathbf{u}_i)\left(\mathbf{w}^T\mathbf{u}_i + b\right) \geq 1, \quad i = 1, \ldots, N$ *afiniczne*

- funkcja Lagrange'a

$$L(\mathbf{w},\, b\,;\, \boldsymbol{\alpha}) = \tfrac{1}{2}\,\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{N} \alpha_i \left(1 - \ell(\mathbf{u}_i)\left(\mathbf{w}^T\mathbf{u}_i + b\right)\right)$$

- zadanie dualne: programowanie kwadratowe, rozwiązanie numeryczne $\Longrightarrow$
- warunki Kuhna-Tuckera nie dają rozwiązania w postaci analitycznej, ale pozwalają na określenie jego właściwości $\Longrightarrow$

Zadanie dualne

- funkcja dualna: $\dfrac{\partial L}{\partial \mathbf{w}} = \dfrac{\partial L}{\partial b} = 0$ (K-T 1) czyli

$$\mathbf{w}^* = \sum_{i=1}^{N} \alpha_i \, \mathbf{u}_i^* \, \ell(\mathbf{u}_i) \qquad \text{podstawiamy do funkcji Lagrange'a}$$

$$\sum_{i=1}^{N} \alpha_i \, \ell(\mathbf{u}_i) = 0 \qquad \text{ograniczenie w problemie dualnym}$$

czyli $L^*(\boldsymbol{\alpha}) = -\frac{1}{2}\, \mathbf{w}^{*\,T} \mathbf{w}^* + \sum_{i=1}^{N} \alpha_i = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \ell(\mathbf{u}_i) \, \ell(\mathbf{u}_j) \, \mathbf{u}_i^T \mathbf{u}_j$

zadanie dualne: maksymalizować

$$L^*(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \ell(\mathbf{u}_i) \, \ell(\mathbf{u}_j) \, \mathbf{u}_i^T \mathbf{u}_j$$

przy ograniczeniach $\sum_{i=1}^{N} \alpha_i \, \ell(\mathbf{u}_i) = 0 \qquad \alpha_i \geq 0, \qquad i = 1, \ldots, N$

- $\boldsymbol{\alpha}^*$ **zależy od** $\mathbf{u}_i$ **tylko poprzez iloczyny skalarne** $\mathbf{u}_i^T \mathbf{u}_j$
- numeryczne rozwiązanie problemu dualnego: programowanie kwadratowe

Własności rozwiązania

- trzeci warunek Kuhna-Tuckera

$$\alpha_i \left(1 - \ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right)\right) = 0, \qquad i = 1, \ldots, N \qquad \text{K-T 3a}$$

$$\ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) \geq 1, \qquad i = 1, \ldots, N \qquad \text{K-T 3b}$$

$$\alpha_i^* \geq 0, \qquad i = 1, \ldots, N \qquad \text{K-T 3c}$$

$\alpha_i = 0$	$\ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) < 1$	$\mathbf{u}_i$ wewnątrz zbioru dopuszczalnego
$\alpha_i \geq 0$	$\ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) = 1$	$\mathbf{u}_i$ — obraz podpierający

- dla obrazów *innych niż podpierające* $\alpha_i^* = 0$ (K-T 3a) czyli

$$\mathbf{w}^* = \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^* \, \mathbf{u}_i \, \ell(\mathbf{u}_i)$$

- optymalny margines geometryczny
  $\|\mathbf{w}^*\|^2 = \mathbf{w}^{*T} \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^* \ell(\mathbf{u}_i)\, \mathbf{u}_i = \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^* \left(1 - \ell(\mathbf{u}_i)\, b\right) = \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^*$ czyli
  $\Delta_g = \frac{1}{\sqrt{\sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^*}}$

Wyznaczenie obciążenia

- b^* można wyznaczyć z ograniczeń pierwotnych

$$\begin{array}{ll} 1 - \mathbf{w}^{*T}\mathbf{u}_i \le b^* & \text{dla } \ell(\mathbf{u}_i) = 1 \\ b^* \le -1 - \mathbf{w}^{*T}\mathbf{u}_i & \text{dla } \ell(\mathbf{u}_i) = -1 \end{array} \text{, czyli } \begin{array}{l} b^* \ge 1 - \min_{\mathbf{u}_i:\ell(\mathbf{u}_i)=1} \mathbf{w}^{*T}\mathbf{u}_i \\ b^* \le -1 - \max_{\mathbf{u}_i:\ell(\mathbf{u}_i)=-1} \mathbf{w}^{*T}\mathbf{u}_i \end{array}$$

ostatecznie

$$b^* = \frac{-1}{2}\Big(\min_{\mathbf{u}_i \in \mathcal{U}_{L+}} \mathbf{w}^{*T}\mathbf{u}_i + \max_{\mathbf{u}_i \in \mathcal{U}_{L-}} \mathbf{w}^{*T}\mathbf{u}_i\Big)$$

- dla dowolnego **obrazu podpierającego** $\ell(\mathbf{u}_j)\left(\mathbf{w}^{*T}\mathbf{u}_j + b^*\right) = 1$ tzn.

$$b^* = \ell(\mathbf{u}_j) - \mathbf{w}^{*T}\mathbf{u}_j, \quad \mathbf{u}_j \in \mathcal{U}_S$$

Iloczyn skalarny w przestrzeni cech

- reguła klasyfikacji dla sieci jednowarstwowej

$$\mathrm{N}(\mathbf{u}) = \mathrm{sign}\left(b^* + \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^* \, \ell(\mathbf{u}_i) \, \mathbf{u}_i^T \mathbf{u}\right)$$

  $\mathcal{U}_S$, α_i^*, b^* znalezione przez rozwiązanie zadania minimalizacji

- reguła klasyfikacji dla sieci dwuwarstwowej
  pierwsza warstwa: obliczanie cech $\mathbf{x} = \boldsymbol{\varphi}(\mathbf{u})$
  druga warstwa:

$$\mathrm{N}(\mathbf{u}) = \mathrm{sign}\left(b^* + \sum_{\boldsymbol{\varphi}(\mathbf{x}_i) \in \mathcal{X}_S} \alpha_i^* \, \ell(\mathbf{u}_i) \, \mathbf{x}_i^T \mathbf{x}\right)$$

- iloczyn skalarny w przestrzeni cech

$$\mathbf{x}_i^T \mathbf{x} = \boldsymbol{\varphi}(\mathbf{u}_i)^T \boldsymbol{\varphi}(\mathbf{u}) = k(\mathbf{u}_i, \mathbf{u})$$

- maszynę dwuwarstwową można zmodyfikować, gdyż **nie trzeba obliczać cech** — rozwiązanie zależy tylko od iloczynu skalarnego k

- przypomnienie: iloczyn skalarny
  symetria: $k(\mathbf{u}', \mathbf{u}'') = k(\mathbf{u}'', \mathbf{u}')$, warunek Cauchy-Schwarza: $k(\mathbf{u}', \mathbf{u}'')^2 \le k(\mathbf{u}', \mathbf{u}') \, k(\mathbf{u}'', \mathbf{u}'')$

SVM maksymalizujące margines

- reguła klasyfikacji

$$\mathrm{N}(\mathbf{u}) = \operatorname{sign} d(\mathbf{u}_i) = \operatorname{sign}\Big(b^* + \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^* \, \ell(\mathbf{u}_i)\, k(\mathbf{u}_i, \mathbf{u})\Big)$$

gdzie

$\boldsymbol{\alpha}^*$ jest jednoznacznym rozwiązaniem problemu maksymalizacji funkcji

$$L^*(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \ell(\mathbf{u}_i)\, \ell(\mathbf{u}_j)\, k(\mathbf{u}_i, \mathbf{u}_j)$$

przy ograniczeniach $\displaystyle\sum_{i=1}^{N} \alpha_i \, \ell(\mathbf{u}_i) = 0 \qquad \alpha_i \geq 0, \qquad i = 1, \ldots, N$

$b^* = \ell(\mathbf{u}_j) - \sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i^* \, \ell(\mathbf{u}_i)\, k(\mathbf{u}_i, \mathbf{u}_j)\Big)$ dla dowolnego $\mathbf{u}_i \in \mathcal{U}_S$

- $\alpha_i > 0$ $\mathbf{u}_i$ — obrazy podpierające, $\mathbf{u}_i \in \mathcal{U}_S$
- margines geometryczny $\dfrac{1}{\sqrt{\sum_{\mathbf{u}_i \in \mathcal{U}_S} \alpha_i}}$

Warunek Mercera

- **nie każdy iloczyn skalarny jest realizowany przez pewną przestrzeń cech**
- *warunek Mercera* gwarantuje istnienie funkcji φ:

  dla każdego skończonego podzbioru $\{\mathbf{u}_1, \ldots, \mathbf{u}_N\} \subset \mathcal{U}$ macierz $\{k(\mathbf{u}_i, \mathbf{u}_j)\}$ jest dodatnio półokreślona
- przykłady: $k(\mathbf{u}', \mathbf{u}'') = (\mathbf{u}'^T \mathbf{u}'' + 1)^p$, $\quad k(\mathbf{u}', \mathbf{u}'') = \exp\left(\frac{-1}{2\sigma^2}\|\mathbf{u}' - \mathbf{u}''\|^2\right)$
- jeżeli funkcje k spełniają warunek Mercera, to warunek ten spełnia

  każda kombinacja liniowa tych funkcji

  iloczyn tych funkcji

  $k\big(f(\mathbf{u}'), f(\mathbf{u}'')\big)$ dla dowolnej funkcji f

  $\pi(k(\mathbf{u}', \mathbf{u}''))$; π - wielomian o nieujemnych współczynnikach

  $\exp(k(\mathbf{u}', \mathbf{u}''))$

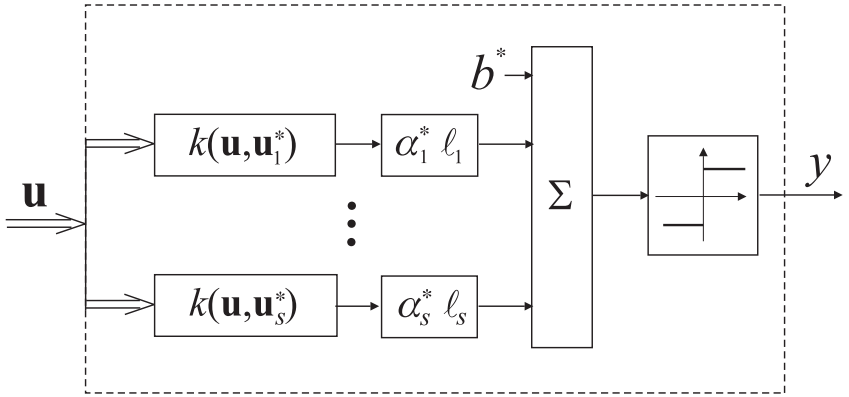
Przykłady SVM

- *maszyna wielomianowa*

$$k(\mathbf{u}', \mathbf{u}'') = (\mathbf{u}'^T \mathbf{u}'' + 1)^p$$

- *maszyna radialna*

$$k(\mathbf{u}', \mathbf{u}'') = \exp\Big(\frac{-1}{2\sigma^2} \|\mathbf{u}' - \mathbf{u}''\|^2\Big)$$



- uogólnienie maszyny radialnej

$$k(\mathbf{u}', \mathbf{u}'') = \exp\Big(\frac{-1}{2\sigma^2} d(\mathbf{u}', \mathbf{u}'')\Big)$$

gdzie d jest miarą odległości obrazów, np.

$$d(\mathbf{u}', \mathbf{u}'') = \chi^2(\mathbf{u}', \mathbf{u}'') \approx \sum_{k=1}^{n} \frac{|u'_k - u''_k|^2}{u'_k + u''_k}$$

W.M. ?

$$d(\mathbf{u}', \mathbf{u}'') = \|\mathbf{u}' - \mathbf{u}''\|_p = \sqrt[p]{\sum_{k=1}^{n} |u'_k - u''_k|^p}$$

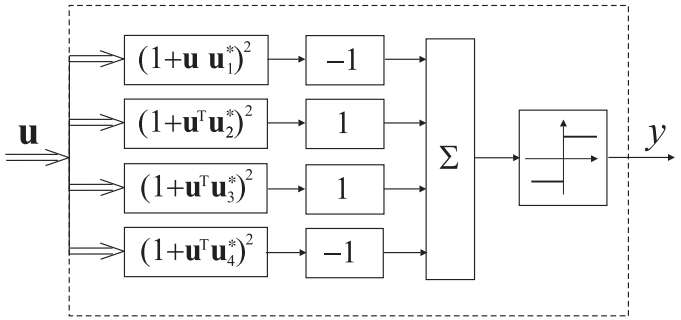
W.M. spełniony dla $p = 1, 2$

Przykład: Maszyna wielomianowa dla problemu XOR

- $\mathbf{u}_i$ $\begin{bmatrix}-1 & -1\end{bmatrix}^T$, $\begin{bmatrix}-1 & 1\end{bmatrix}^T$, $\begin{bmatrix}1 & -1\end{bmatrix}^T$, $\begin{bmatrix}1 & 1\end{bmatrix}^T$
  $\ell(\mathbf{u}_i)$ -1, 1, 1, -1
- $k(\mathbf{u}', \mathbf{u}'') = (1 + \mathbf{u}'^T \mathbf{u}'')^2$
- problem dualny: minimalizować $L^*(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \boldsymbol{i} - \frac{1}{2}\,\boldsymbol{\alpha}^T \big(\boldsymbol{\ell}\,\boldsymbol{\ell}^T \odot K\big)\boldsymbol{\alpha}$ ($\odot$ - produkt Schura)
  przy ograniczeniach $\boldsymbol{\alpha} \geq 0$, $\boldsymbol{\ell}^T \boldsymbol{\alpha} \geq 0$

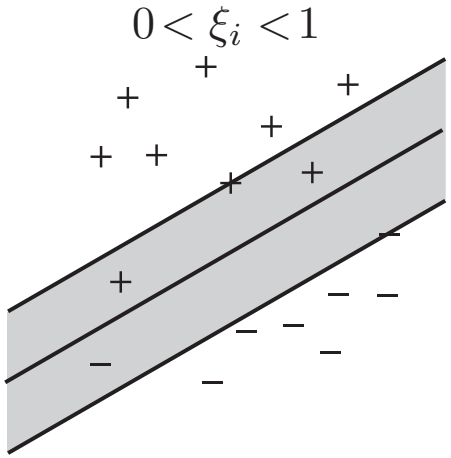
  gdzie $\boldsymbol{\ell} = \begin{bmatrix}-1\\1\\1\\-1\end{bmatrix}$, $\boldsymbol{\ell}\,\boldsymbol{\ell}^T = \begin{bmatrix}1 & -1 & -1 & 1\\-1 & 1 & 1 & -1\\-1 & 1 & 1 & -1\\1 & -1 & -1 & 1\end{bmatrix}$, $\mathbf{K} = \{k(\mathbf{u}_i, \mathbf{u}_j)\} = \begin{bmatrix}9 & 1 & 1 & 1\\1 & 9 & 1 & 1\\1 & 1 & 9 & 1\\1 & 1 & 1 & 9\end{bmatrix}$
- rozwiązanie dualne bez ograniczeń: $\boldsymbol{\alpha}^* = \big(\boldsymbol{\ell}\,\boldsymbol{\ell}^T \odot K\big)^{-1}\,\boldsymbol{i} = \frac{1}{96}\begin{bmatrix}11 & 1 & 1 & -1\\1 & 11 & -1 & 1\\1 & -1 & 11 & 1\\-1 & 1 & 1 & 11\end{bmatrix}\boldsymbol{i} = \frac{1}{8}\,\boldsymbol{i}$,

  ograniczenia dualne spełnione, wszystkie $\mathbf{u}_i$ są podpierające
- reguła klasyfikacji $y(\mathbf{u}) = \operatorname{sign}\Big(b^* + \sum_{\mathbf{u}_i \in \mathcal{U}} \alpha_i^*\,\ell(\mathbf{u}_i)\,k\big(\mathbf{u}, \mathbf{u}_i\big)\Big)$
- $b^* = \ell(\mathbf{u}_j) - \sum_{\mathbf{u}_i \in \mathcal{U}_s} \alpha_i^*\,\ell(\mathbf{u}_i)\,k\big(\mathbf{u}_i, \mathbf{u}_j\big)$ dla $\mathbf{u}_j \in \mathcal{U}_S$
  czyli $b^* = 0$
- przestrzeń cech: ponieważ
  $k(\mathbf{u}, \mathbf{u}') = (1 + \mathbf{u}^T\mathbf{u}')^2 = 1 + 2\,\mathbf{u}^T\mathbf{u}' + (\mathbf{u}^T\mathbf{u}')^2$
  $= 3 + 2u_1u_1' + 2u_2u_2' + 2u_1u_2u_1'u_2'$
  a zatem $\boldsymbol{\varphi}(\mathbf{u}) = \begin{bmatrix}\sqrt{3} & \sqrt{2}\,u_1 & \sqrt{2}\,u_2 & \sqrt{2}\,u_1u_2\end{bmatrix}^T$

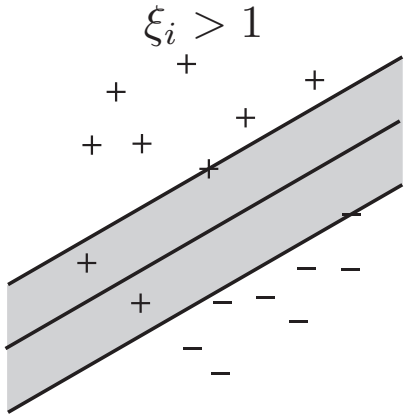


Klasy liniowo nierozdzielne

- przypadek liniowo rozdzielny
$$\left(\mathbf{w}^T\mathbf{u}_i + b\right)\ell(\mathbf{u}_i) \geq 1 \quad \text{dla } \mathbf{u}_i \in \mathcal{U}_L$$
- *miękki margines*: możliwość naruszenia warunków poprawnej klasyfikacji dla i-tego obrazu
$$\left(\mathbf{w}^T\mathbf{u}_i + b\right)\ell(\mathbf{u}_i) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \text{dla } \mathbf{u}_i \in \mathcal{U}_L,$$
- ξ_i - *zmienne dopełniające* *(slack variables)*



naruszony region rozdzielający



brak separacji liniowej

Liniowa kara za naruszenie ograniczeń

- problem pierwotny (c kontroluje liczbę błędnych klasyfikacji)

  minimalizować funkcję $\quad Q(\mathbf{w},\, b,\, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + c\,\sum_{i=1}^{N}\xi_i$

  przy ograniczeniach $\quad \xi_i \geq 0, \quad \ell(\mathbf{u}_i)\left(\mathbf{w}^T\mathbf{u}_i + b\right) \geq 1 - \xi_i, \quad i = 1,\ldots,N$

- funkcja Lagrange'a $L(\mathbf{w},\, b,\, \boldsymbol{\xi};\, \boldsymbol{\alpha}, \boldsymbol{\lambda})$

$$= \frac{1}{2}\,\mathbf{w}^T\mathbf{w} + c\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i\left(1 - \xi_i - \ell(\mathbf{u}_i)\left(\mathbf{w}^T\mathbf{u}_i + b\right)\right) - \sum_{i=1}^{N}\lambda_i\,\xi_i$$

- warunki Kuhna-Tuckera

$$\mathbf{w}^* = \sum_{i=1}^{N}\alpha_i\,\mathbf{u}_i\,\ell(\mathbf{u}_i), \qquad \sum_{i=1}^{N}\alpha_i\,\ell(\mathbf{u}_i) = 0$$

$$\alpha_i + \lambda_i = c, \qquad i = 1,\ldots,N \qquad \text{(K-T 1)}$$

$$\alpha_i\left(1 - \xi_i - \ell(\mathbf{u}_i)\left(\mathbf{w}^T\mathbf{u}_i + b\right)\right) = 0, \qquad \lambda_i\,\xi_i = 0, \qquad i = 1,\ldots,N \qquad \text{(K-T 3a)}$$

$$\ell(\mathbf{u}_i)\left(\mathbf{w}^T\mathbf{u}_i + b\right) \geq 1 - \xi_i, \qquad \xi_i \geq 0, \qquad i = 1,\ldots,N \qquad \text{(K-T 3b)}$$

$$\alpha_i \geq 0, \qquad \lambda_i \geq 0, \qquad i = 1,\ldots,N \qquad \text{(K-T 3c)}$$

- b^* trzeba wyznaczyć z ograniczeń pierwotnych

SVM dla liniowej kary

- funkcja dualna - identyczna jak przy "ostrym marginesie"

$$L^*(\boldsymbol{\alpha},\boldsymbol{\lambda}) = -\tfrac{1}{2}\,\mathbf{w}^{*T}\mathbf{w}^* + \sum_{i=1}^{N}\alpha_i = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\ell(\mathbf{u}_i)\,\ell(\mathbf{u}_j)\,k(\mathbf{u}_i,\mathbf{u}_j)$$

- problem dualny: maksymalizować funkcję

$$L^*(\boldsymbol{\alpha}) = \textstyle\sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\ell(\mathbf{u}_i)\,\ell(\mathbf{u}_j)\,k(\mathbf{u}_i,\mathbf{u}_j)$$

przy ograniczeniach $\sum_{i=1}^{N}\alpha_i\,\ell(\mathbf{u}_i) = 0, \qquad 0 \le \alpha_i \le c, \qquad i = 1,\ldots,N$

$\alpha_i = 0$	$\ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) < 1$	$\mathbf{u}_i$ wewnątrz zbioru dopuszczalnego
$0 \le \alpha_i \le c$	$\ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) = 1$	$\mathbf{u}_i$ — obraz podpierający
$\alpha_i = c$	$\ell(\mathbf{u}_i)\left(\mathbf{w}^{*T}\mathbf{u}_i + b^*\right) = 1$	$\mathbf{u}_i$ narusza ograniczenia $(\xi_i > 0)$

- *reguła klasyfikacji w przestrzeni obrazów*

$$y(\mathbf{u}) = \operatorname{sign}\Big(b^* + \sum_{\mathbf{u}_i\in\mathcal{U}_s}\alpha_i^*\,\ell(\mathbf{u}_i)\,k\big(\mathbf{u},\mathbf{u}_i\big)\Big)$$

- b^* dobrane tak, by spełniać warunek $\ell(\mathbf{u}_i)\,y(\mathbf{u}_i) = 1$ dla każdego $i : 0 < \alpha_i^* < c$
- margines geometryczny $\Delta_g = \dfrac{1}{\sqrt{\sum_{\mathbf{u}_i\in\mathcal{U}_S}\sum_{\mathbf{u}_j\in\mathcal{U}_S}\alpha_i^*\,\alpha_j^*\ell(\mathbf{u}_i)\,\ell(\mathbf{u}_j)\,k(\mathbf{u}_i,\mathbf{u}_j)}}$

SVM dla liniowej kary – problem skalowania

- zmodyfikowany problem dualny: $0 \le c_0 \le 1$

  maksymalizować funkcję $L^*(\boldsymbol{\alpha}) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \, \alpha_j \ell(\mathbf{u}_i) \, \ell(\mathbf{u}_j) \, k(\mathbf{u}_i, \mathbf{u}_j)$

  przy ograniczeniach $\sum_{i=1}^N \alpha_i \, \ell(\mathbf{u}_i) = 0, \qquad \sum_{i=1}^N \alpha_i \ge c_0, \qquad 0 \le \alpha_i \le 1/N,$ $i = 1, \ldots, N$

- $0 < \alpha_i < \frac{1}{N}$ $\mathbf{u}_i$ - wektor podpierający
  $\alpha_i = \frac{1}{N}$ $\mathbf{u}_i$ - wektor naruszający ograniczenia

- b^* dobrane tak, by spełniać warunek

  $\ell(\mathbf{u}_i) \, y(\mathbf{u}_i) = 1$ dla każdego wektora podpierającego $\mathbf{u}_i$ (tzn jeżeli $0 < \alpha_i^* < \dfrac{1}{N}$)

Rozwiązania numeryczne

- metoda największego spadku

$$\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k) + \eta \, \frac{dL^*(\boldsymbol{\alpha})}{d\boldsymbol{\alpha}}$$

- *wersja sekwencyjna*, j wybierane losowo

$$\alpha_j(k+1) = \alpha_j(k) + \eta_j \, \frac{\partial L^*(\boldsymbol{\alpha})}{\partial \alpha_j} = \alpha_j(k) + \eta_j \big(1 - \ell(u_i)\, K(\mathbf{u}_i, \mathbf{u}_j)\big)$$

$$\alpha_j(k+1) =: \max(0, \alpha_j(k+1))$$

$$\alpha_j(k+1) =: \min(c, \alpha_j(k+1)) \quad \text{(dla problemu z karą liniową)}$$

warunek liniowy związany z b jest niemożliwe do spełniania — założyć b

- ograniczenia liniowe zastąpić przez

$$\Big|\sum_{\mathbf{u}\in\mathcal{U}} \alpha_i \, \ell(u_i)\Big| \le z_i(k)$$

gdzie ciągi $\{z_i(k)\}$ są malejące

- *procedury dwustopniowe* (nawet duże zbiory obrazów, rzędu 10^6)
  - i dla bieżącego $\boldsymbol{\alpha}$, znaleźć obrazy podpierające $\mathcal{U}_{\boldsymbol{\alpha}} = \{\mathbf{u}_i \in \mathcal{U} : \alpha_i > 0\}$
  - ii przeprowadzić optymalizację dla $\mathbf{u} \in \mathcal{U}_{\boldsymbol{\alpha}}$
  - iii wykonać iteracje dla wszystkich $\mathbf{u}_i \notin \mathcal{U}_{\alpha_i}$
  - iv utworzyć $\boldsymbol{\alpha}$, przejść do [i]

Porcjowanie i dekompozycja

- *porcjowanie*
  - wydzielić *zbiór roboczy* obrazów,
  - powtarzać do spełnienia warunku stopu
    - przeprowadzić optymalizację na zbiorze roboczym,
    - utworzyć nowy zbiór roboczy spośród obrazów naruszających warunki KT
- *dekompozycja*: porcjowanie z zachowaniem stałej objętości zbioru roboczego
- dekompozycja dla 2-elementowych zbiorów roboczych: *sekwencyjna optymalizacja minimalna*
  - ograniczenia $0 \le \alpha', \alpha'' \le c$ prowadzą do $U \le \alpha'' \le V$ gdzie

$$U = \max(0, \alpha'' - \alpha'), \qquad V = \min(c, c - \alpha' + \alpha'') \qquad \text{jeśli } \mathbf{u}', \mathbf{u}'' \text{ w różnych klasach}$$

$$U = \max(0, \alpha' + \alpha'' - c), \qquad V = \min(c, \alpha' + \alpha'') \qquad \text{jeśli } \mathbf{u}', \mathbf{u}'' \text{ w tej samej klasie}$$

  - wyznaczenie α', α''

$$e' = d(\mathbf{u}') - \ell(\mathbf{u}') = \Big(\sum_{\mathbf{u}_j \in \mathcal{U}} \alpha_j \, \ell(\mathbf{u}_j) \, k(\mathbf{u}_j, \mathbf{u}') + b \Big) - \ell(\mathbf{u}'), \quad e'' d(\mathbf{u}'') - \ell(\mathbf{u}'')$$

$$\kappa = k(\mathbf{u}', \mathbf{u}') + k(\mathbf{u}'', \mathbf{u}'') - 2\, k(\mathbf{u}', \mathbf{u}'')$$

$$\alpha''(k) = \alpha''(k-1) + (e' - e'')\, \ell(\mathbf{u}'')/\kappa \quad \text{warunek liniowy}, \quad \alpha''(k) =: \max(U, \alpha''(k))$$

$$\alpha'(k) = \alpha'(k-1) + \ell(\mathbf{u}')\ell(\mathbf{u}'')\big(\alpha''(k-1) - \alpha''(k)\big)$$

6. Adaline

Bibliografia

[0-1] B. Widrow and M.A. Lehr, "30 years of adaptive neural networks: perceptron, madaline, and backpropagation,", Proceedings of IEEE, vol. 78, No. 9, Sept. 1990

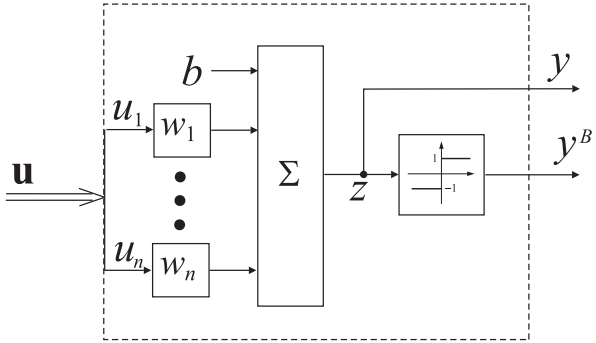
Adaline - Liniowy neuron

- **ADA**ptive **LI**near **NE**uron
- **ADA**ptive **LIN**ear **E**lement

- współczesne perceptrony wielowarstwowe

  perceptron Rosenblatta $\longrightarrow$ struktura sieci

  Adaline $\longrightarrow$ algorytmy uczenia



liniowe wyjście sieci (trening wag) $\quad y(t) = \mathbf{w}(t)^T \mathbf{u}(t)$

binarne wyjście sieci (klasyfikacja) $\quad y^B(t) = \mathrm{sign}\big(\mathbf{w}(t)^T \mathbf{u}(t)\big)$

odchyłka wyjścia liniowego $\quad \varepsilon(t) = y^*(t) - y(t) = y^*(t) - \mathbf{w}(t)^T \mathbf{u}(t)$

Minimalizacja błędu kwadratowego

- *błąd kwadratowy* wyjścia sieci (dla jednowymiarowego wyjścia)

$$\widehat{Q}_N = \widehat{Q}_N(\mathbf{w}) = \frac{1}{2}\sum_{t=1}^{N} |y^*(t) - \mathbf{w}^T\mathbf{u}(t)|^2 = \frac{1}{2}\sum_{t=1}^{N} |\varepsilon(t)|^2$$

- gradient i hessian

$$\frac{d\widehat{Q}_N(\mathbf{w})}{d\mathbf{w}} = -\sum_{t=1}^{N} \mathbf{u}(t)\,\varepsilon(t) = -\sum_{t=1}^{N} \mathbf{u}(t)\,y^*(t) + \sum_{t=1}^{N} \mathbf{u}(t)\,\mathbf{u}(t)^T\,\mathbf{w}$$

$$\frac{d^2\widehat{Q}_N(\mathbf{w})}{d\mathbf{w}\,d\mathbf{w}^T} = \sum_{t=1}^{N} \mathbf{u}(t)\,\mathbf{u}(t)^T > 0 \quad \textit{warunek jednoznaczności}$$

- *wagi optymalne* minimalizują uśredniony błąd kwadratowy odchyłki

$$\mathbf{w}^* = \Big(\sum_{t=1}^{N} \mathbf{u}(t)\,\mathbf{u}(t)^T\Big)^{-1} \sum_{t=1}^{N} \mathbf{u}(t)\,y^*(t) = \widehat{R}_{\mathbf{u}}(t)^{-1}\,\widehat{R}_{\mathbf{u},y^*}(t)$$

- uwzględnienie informacji a priori:

$$\widehat{Q}'_N = \widehat{Q} + \frac{1}{2}(\mathbf{w}-\mathbf{w}_0)^T\mathbf{P}_0^{-1}(\mathbf{w}-\mathbf{w}_0)$$

- wagi optymalne

$$\mathbf{w}^* = \Big(\mathbf{P}_0^{-1} + \sum_{t=1}^{N} \mathbf{u}(t)\,\mathbf{u}(t)^T\Big)^{-1} \Big(\mathbf{P}_0^{-1}\mathbf{w}_0 + \sum_{t=1}^{N} \mathbf{u}(t)\,y^*(t)\Big)$$

Rekurencyjne najmniejsze kwadraty

- realizacja rekurencyjna z uwzględnieniem informacji a priori: szczególny przypadek *Filtru Kalmana*

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mathbf{P}(t)\,\mathbf{u}(t)\,\varepsilon(t) \qquad \textit{aktualizacja wag}$$

$$\mathbf{P}(t+1) = \mathbf{P}(t) - \frac{\mathbf{P}(t)\,\mathbf{u}(t)\,\mathbf{u}^T(t)\,\mathbf{P}(t)}{1+\mathbf{u}^T(t)\,\mathbf{P}(t)\,\mathbf{u}(t)} \qquad \textit{aktualizacja wzmocnień}$$

$$\mathbf{w}(0) = \mathbf{w}_0, \quad \mathbf{P}(0) = \mathbf{P}_0 > 0 \qquad \textit{warunki początkowe}$$

- równoważna postać aktualizacji wzmocnień dla $\mathbf{P}(t) > 0$

$$\mathbf{P}(t)^{-1} = \mathbf{u}(t)\mathbf{u}(t)^T + \mathbf{P}(t-1)^{-1} = \mathbf{P}_0^{-1} + \sum_{s=1}^{t} \mathbf{u}(s)\,\mathbf{u}(s)^T$$

  stosowanie jej do aktualizacji wag wymaga jednak odwracania macierzy

- błąd średniokwadratowy wag

$$\|\mathbf{w}(t) - \mathbf{w}(t-1)\| \downarrow 0 \quad \textbf{słabszy niż warunek Cauchy'ego !}$$

$$\text{jeżeli } \min \operatorname{eig}\left(\mathbf{P}(t)^{-1}\right) \to \infty \quad \text{to} \quad \mathbf{w}(t) \to \mathbf{w}^*$$

- własności odchyłek wyjść

$$\frac{\varepsilon(t)}{\sqrt{1+\kappa\|\mathbf{u}(t)\|^2}} \to 0 \quad \text{gdzie} \quad \kappa = \max \operatorname{eig}(\mathbf{P}_0)$$

Algorytm najmniejszych średnich kwadratów LMS

- metoda największego spadku

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \mu \frac{d\widehat{Q}_N(\mathbf{w})}{d\mathbf{w}}$$
$$= \mathbf{w}(t) - \mu \sum_{t=1}^{N} \mathbf{u}(t)\,\varepsilon(t)$$

- *algorytm najmniejszych średnich kwadratów μ-LMS*:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu\,\mathbf{u}(t)\,\varepsilon(t)$$

- jeśli obrazy są losowane niezależnie z rozkładu o macierzy kowariancji $\mathbf{R_u}$ to

  średnie wagi $\mathcal{E}\mathbf{w}(t)$ są zbieżne jeżeli

$$0 < \mu \le \frac{2}{\max \mathrm{eig}(\mathbf{R_u})}$$

  błąd średniokwadratowy $\mathcal{E}|\varepsilon(t)|^2$ jest zbieżny jeśli

$$0 < \mu \le \frac{2}{\mathrm{tr}\,\mathbf{R_u}} \qquad (\mathrm{tr}\,\mathbf{R_u} = \sum \mathrm{eig}(\mathbf{R_u}))$$

Zasada najmniejszej zmiany

- minimalizować $Q(\mathbf{w}^+) = \frac{1}{2}\|\mathbf{w}^+ - \mathbf{w}\|^2$ przy ograniczeniach $y^* = \mathbf{w}^{+T}\mathbf{u}$
- rozwiązanie $\mathbf{w}^+$ jest rzutem $\mathbf{w}$ na hiperpłaszczyznę $y^* = \mathbf{w}^{+T}\mathbf{u}$
- *funkcja Lagrange'a*

$$L(\mathbf{w}^+, \beta) = Q(\mathbf{w}^+) + \beta\,(y^* - \mathbf{w}^{+T}\mathbf{u})$$

warunki Kuhna-Tuckera (funkcja Q jest wypukła i ciągła wraz z pochodnymi na $\mathbb{R}^d$, ograniczenia są afiniczne)

$$\begin{aligned} \frac{dL}{d\mathbf{w}^+} &= \mathbf{w}^+ - \mathbf{w} - \beta\,\mathbf{u} = 0 \\ \frac{dL}{d\beta} &= y^* - \mathbf{w}^{+T}\mathbf{u} = 0 \end{aligned} \qquad \Longrightarrow \qquad \beta\,\|\mathbf{u}\|^2 = y^* - \mathbf{w}^T\mathbf{u} = \varepsilon$$

- optymalne wagi

$$\mathbf{w}^+ = \mathbf{w} + \frac{1}{\|\mathbf{u}(t)\|^2}\,\mathbf{u}(t)\,\varepsilon(t) \qquad (\text{dla } \|\mathbf{u}(t)\| > 0)$$

Algorytmy LMS z normalizacją

- *algorytm Kaczmarza*

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \frac{1}{\|\mathbf{u}(t)\|^2}\,\mathbf{u}(t)\,\varepsilon(t) \qquad (\text{dla } \|\mathbf{u}(t)\| > 0)$$

- *algorytm normalizowanych najmniejszych średnich kwadratów (α-LMS)*

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \frac{\alpha}{\|\mathbf{u}(t)\|^2}\,\mathbf{u}(t)\,\varepsilon(t) \qquad (\text{dla } \|\mathbf{u}(t)\| > 0)$$

- zmiany wyjścia liniowego i odchyłki wyjścia są proporcjonalne do odchyłki ($y(t+) =$ wyjście obliczone dla zmodyfikowanej wagi $= \mathbf{w}(t+1)^T\mathbf{u}(t)$)

$$y(t+1) - y(t) = \big(\mathbf{w}(t+) - \mathbf{w}(t)\big)^T\mathbf{u}(t) = \alpha\,\varepsilon(t)$$
$$\varepsilon(t+) - \varepsilon(t) = -\alpha\,\varepsilon(t)$$

- skalowanie $\mathbf{u}' = \frac{\mathbf{u}}{\|\mathbf{u}\|}$, $y' = \frac{y}{\|\mathbf{u}\|}$, $\mathbf{y}^{*\prime} = \frac{\mathbf{y}^*}{\|\mathbf{u}\|}$ prowadzi do μ-LMS

- *zmodyfikowany algorytm α-LMS*

$$\mathbf{w}(t+) = \mathbf{w}(t) + \frac{\alpha}{c + \|\mathbf{u}(t)\|^2}\,\mathbf{u}(t)\,\varepsilon(t)$$

$c > 0$ (człon regularyzujący), $0 < \alpha < 2$

Własności algorytmów LMS z normalizacją

- własności ciągu wag
  - $\|\mathbf{w}(t+1) - \mathbf{w}(t)\| \downarrow 0$
  - jeżeli $\sum_{i=t}^{s+k} \frac{\mathbf{u}(i)\mathbf{u}(i)^T}{\|\mathbf{u}(i)\|^2} > cI$ dla każdego t i pewnego k, to

$$\mathbf{w}(t) \to \mathbf{w}^*$$

- ważone odchyłki wyjścia sieci maleją do zera

$$\frac{\varepsilon(t)}{\sqrt{c + \|\mathbf{u}(t)\|^2}} \to 0$$

7. Zagadnienia aproksymacji funkcji

Bibliografia

[0-1] A.N. Kolmogorov, "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition," *Dokl. Akad. Nauk USSR*, vol. 114, pp. 953–956, 1957

[0-2] R. Hecht-Nielsen, "Kolmogorov's mapping neural network existence theorem", *Proc. Int. Conf. on Neural Networks*, vol. III, pp. 11–13, 1987

[0-3] K. Hornik, M. Stinchcombe, and H. White "Multilayer feedforward networks are univeral approximators," *Neural Networks*, vol 2, No. 5, pp. 359–366, 1989

[0-4] K. Hornik, M. Stinchcombe, and H. White "Univeral approximation of an unknown mapping and its derivatives using multilayer neural networks," *Neural Networks*, vol 3, No. 5, pp. 551–560, 1990

[0-5] M. Leshno, V. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation funtion can approximate any function", *Neural Networks*, vol. 6, pp. 861–867, 1993

Problem aproksymacji

- aproksymacja nieznanej funkcji $f \in \mathcal{F}$
  wybrać $\widehat{f}$ z rodziny funkcji $\widehat{\mathcal{F}}$ tak aby minimalizować $d(f, \widehat{f})$
- rodzina $\mathcal{F}$ wynika z problemu, np. $\mathcal{C}(\mathcal{U})$, $\mathcal{C}_k(\mathcal{U})$, $\mathcal{L}_p(\mathcal{U})$
- d – metryka właściwa dla $\mathcal{F}$
- nieznana funkcja poznawana poprzez eksperyment

$$\big(\mathbf{u}(1), \mathbf{y}(1)\big),\ \big(\mathbf{u}(2), \mathbf{y}(2)\big),\ \ldots,\ \big(\mathbf{u}(N), \mathbf{y}(N)\big)$$

- czy $\widehat{\mathcal{F}}$ jest wystarczająco bogata?
- uniwersalny aproksymator
  dla dowolnych $\epsilon > 0$, $f \in \mathcal{F}$ istnieje $\widehat{f} \in \widehat{\mathcal{F}}$ dla której

$$d(f, \widehat{f}) < \epsilon$$

- **czy sieci neuronowe są uniwersalnymi aproksymatorami ?**

Reprezentacja Kołmogorowa

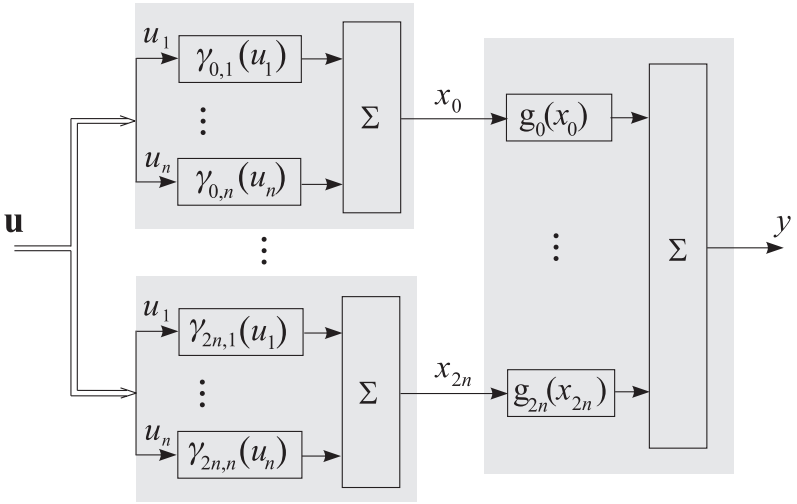
- 13 Hipoteza Hilberta istnieje analityczna funkcja 3 zmiennych która *nie jest* skończoną superpozycją funkcji ciągłych 2 zmiennych

- Kołmogorow każda funkcja ciągła n zmiennych $f : \mathcal{I}^n \mapsto \mathbb{R}$, $n \geq 2$, ma reprezentację

$$f(\mathbf{u}) = \sum_{i=0}^{2n} g_i \left(\sum_{j=1}^{n} \gamma_{i,j}(u_j) \right)$$

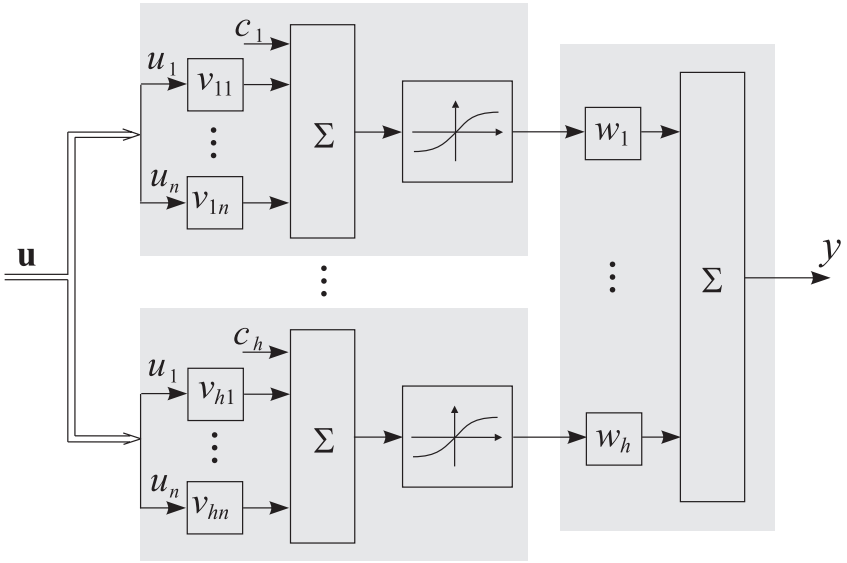
  $\{g_0, \ldots g_{2n}\}$ — ciągłe funkcje 1 zmiennej, zależą od f

  rodzina funkcji monotonicznych ciągłych $\gamma_{i,j} : \mathcal{I} \mapsto \mathbb{R}$ jest uniwersalna



- twierdzenie Kołmogorowa określa *reprezentację* funkcji
  - wymaga znajomości funkcji
  - błąd reprezentacji jest zerowy

Typowa sieć aproksymacyjna



- aproksymacja funkcji $f : \mathcal{U} \subset \mathbb{R}^n \mapsto \mathbb{R}$
- rodzina sieci dwuwarstwowych $\mathcal{N}[g]$
  - warstwa wyjściowa liniowa o zerowym obciążeniu
  - warstwa ukryta o identycznych funkcjach aktywacji g
- "wolne" parametry:
  - liczba neuronów warstwy ukrytej h
  - wagi (nh wag) i obciążenia (h obciążeń) warstwy ukrytej
  - wagi warstwy wyjściowej (h wag)

Aproksymacja funkcji ciągłych

- aproksymowane funkcje są *ciągłe na zbiorze domkniętym ograniczonym* $\mathcal{U} \subset \mathbb{R}^n$
- odległość funkcji rozumiana w sensie supremum tzn.

$$d(f, \widehat{f}) = \sup_{\mathbf{u}\in\mathcal{U}} |f(\mathbf{u}) - \widehat{f}(\mathbf{u})|$$

- [Hornik, 1991] jeżeli funkcja aktywacji g jest

  ciągła, *ograniczona* i *różna od stałej*

  to sieć typu $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie (w sensie supremum) aproksymować dowolną funkcję ciągłą na zbiorze domkniętym i ograniczonym
- tzn. dla każdej funkcji f ciągłej na zbiorze domkniętym i ograniczonym $\mathcal{U}$ i dowolnego $\epsilon > 0$, istnieje sieć typu $\mathcal{N}[g]$ taka, że

$$\sup_{\mathbf{u}\in\mathcal{U}} |f(\mathbf{u}) - \widehat{f}(\mathbf{u})| < \epsilon$$

Aproksymacja funkcji ciągłych - warunek konieczny i wystarczający

- funkcję $g : \mathbb{R} \mapsto \mathbb{R}$ nazywamy *lokalnie istotnie ograniczoną*, jeżeli jest *ograniczona prawie wszędzie* (ograniczona za wyjątkiem zbioru zerowej miary Lebesgue'a μ_L) na każdym podzbiorze domkniętym i ograniczonym
- istotne supremum funkcji

$$\operatorname{ess\,sup}_{\mathcal{U}} |f(u| = \inf\{\delta : \mu_L\{x : |g(x)| > \delta\} = 0\}$$

- [Leshno, 1993] jeżeli
  - funkcja aktywacji g jest lokalnie istotnie ograniczona
  - domknięcie zbioru punktów nieciągłości funkcji aktywacji g ma miarę zero

  to sieć typu $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie (w sensie istotnego supremum) aproksymować dowolną funkcję ciągłą na zbiorze domkniętym i ograniczonym **wtedy i tylko wtedy** gdy g *nie jest wielomianem prawie wszędzie*

Równoczesna aproksymacja funkcji ciągłych i ich pochodnych

- funkcje aproksymowane: ciągłe wraz z pochodnymi do rzędu d na zbiorze domkniętym i ograniczonym $\mathcal{U} \in \mathbb{R}^n$
- [Hornik, Stinchcombe, White, 1990] jeżeli funkcja aktywacji g jest

  d-krotnie różniczkowalna w sposób ciągły, ograniczona i różna od stałej

  to sieć $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie aproksymować (w sensie supremum) dowolną funkcję ciągłą oraz jej pochodne do rzędu d włącznie
- tzn. dla każdej funkcji ciągłej wraz z pochodnymi do rzędu d na zbiorze domkniętym i ograniczonym $\mathcal{U} \in \mathbb{R}^n$ i każdego $\epsilon > 0$ istnieje sieć $\mathcal{N}[g]$, dla której dla wszystkich pochodnych $D^{\mathbf{k}} = \frac{\partial^{k_1+\cdots+k_n}}{\partial u_1^{k_1} \ldots \partial u_n^{k_n}}$, $0 \le k_1 + \cdots + k_n \le d$, równocześnie zachodzi

$$\sup_{\mathbf{u}\in\mathcal{U}} |D^{\mathbf{k}} f(\mathbf{u}) - D^{\mathbf{k}} \widehat{f}(\mathbf{u})| \le \epsilon$$

Aproksymacja funkcji całkowalnych

- funkcje aproksymowane: całkowalne z p-tą potęgą, $1 \leq p < \infty$, na zbiorze ograniczonym $\mathcal{U} \in \mathbb{R}^n$
- odległość wyznaczana przez normę

$$\|f\|_p = \Big(\int_{\mathcal{U}} |f(\mathbf{u})|^p \, d\mathbf{u}\Big)^{1/p}$$

- [Hornik, 1991] jeżeli funkcja aktywacji g jest

  ograniczona i *różna od stałej*

  to sieć $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie (w sensie $\|f\|_p$) aproksymować dowolną funkcję całkowalną z p-tą potęgą na zbiorze ograniczonym
- tzn. dla każdej funkcji całkowalnej z p-tą potęgą na zbiorze ograniczonym $\mathcal{U} \in \mathbb{R}^n$ i każdego $\epsilon > 0$ istnieje sieć $\mathcal{N}[g]$ dla której

$$\|f - \widehat{f}\|_p < \epsilon$$

- warunek Leshno wyznacza warunek konieczny i wystarczający aproksymacji funkcji całkowalnych z p-tą potęgą na zbiorze ograniczonym

Równoczesna aproksymacja funkcji całkowalnych i ich pochodnych

- funkcje aproksymowane: funkcja i jej pochodne do rzędu rzędu m są

  ciągłe i całkowalne z p-tą potęgą

  na zbiorze ograniczonym $\mathcal{U}$

- [Hornik, Stinchcombe, White, 1990] jeżeli funkcja aktywacji g jest

  d-krotnie różniczkowalna w sposób ciągły, *ograniczona* i *różna od stałej*

  to sieć $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie aproksymować (w sensie normy $\|f\|_p$) na zbiorze ograniczonym dowolną funkcję ciągłą oraz jej pochodne do rzędu d włącznie

- tzn. dla każdej funkcji, której pochodne do rzędu d są ciągłe i p-krotnie całkowalne ($d \geq 1$, $1 \leq p < \infty$) na zbiorze ograniczonym i każdego $\epsilon > 0$ istnieje sieć $\mathcal{N}[g]$ dla której

$$\|D^{\mathbf{k}} f - D^{\mathbf{k}} \widehat{f}\|_p < \epsilon$$

  dla każdego $\mathbf{k} : |\mathbf{k}| \leq d$

Aproksymacja funkcji nieciągłych

- funkcje aproksymowane: dowolne (mierzalne) określone na zbiorze domkniętym i ograniczonym
- [Luzin] funkcja mierzalna może być zmodyfikowana do funkcji ciągłej przez zmianę wartości na zbiorze o arbitralnie małej mierze
- jeżeli funkcja aktywacji g jest

  ciągła, ograniczona i różna od stałej

  to sieć $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie (w sensie supremum) aproksymować na podzbiorze dowolnie mało różniącym się od dziedziny funkcji, dowolną funkcję (mierzalną) na zbiorze domkniętym i ograniczonym
- tzn. dla każdej funkcji mierzalnej na zbiorze domkniętym i ograniczonym i każdego $\epsilon > 0$ istnieją: sieć $\mathcal{N}[g]$ i zbiór domknięty i ograniczony $\mathcal{U}_\epsilon \subset \mathcal{U}$ dowolnie bliski zbiorowi $\mathcal{U}$ w tym sensie, że $\mu_L(\mathcal{U} - \mathcal{U}_\epsilon) \leq \epsilon$, dla których

$$\sup_{\mathbf{u} \in \mathcal{U}_\epsilon} |f(\mathbf{u}) - \widehat{f}(\mathbf{u})| \leq \epsilon$$

Aproksymacja funkcji zmiennej losowej

- $\mathbf{u}$ jest zmienną losowa określoną na zbiorze $\mathcal{U} \subset \mathbb{R}^n$; jej rozkład oddaje wagę punktów zbioru $\mathcal{U}$
- funkcje aproksymowane: funkcje p-tego rzędu zmiennej losowej $\mathbf{u}$, tzn. $\mathcal{E}|f(\mathbf{u})|^p < \infty$ dla pewnego p, $1 \le p < \infty$
- odległość funkcji w sensie p-tych momentów, tzn. $d(f, \widehat{f}) = \left(\mathcal{E}\left(f(\mathbf{u}) - \widehat{f}(\mathbf{u})\right)^p\right)^{1/p}$
- [Hornik, 1991] jeżeli funkcja aktywacji g jest

  ograniczona i *różna od stałej*

  to sieć $\mathcal{N}[g]$ o dostatecznie dużej liczbie neuronów ukrytych może dowolnie dokładnie (w sensie p-tych momentów) aproksymować dowolną funkcję p-tego rzędu zmiennej losowej $\mathbf{u}$
- tzn. dla każdej funkcji f dla której $\mathcal{E}|f(\mathbf{u})|^p < \infty$ i każdego $\epsilon > 0$ istnieje sieć $\mathcal{N}[g]$ dla której

  $$\mathcal{E}|f(\mathbf{u}) - \widehat{f}(\mathbf{u})|^p < \epsilon$$

Zestawienie twierdzeń aproksymacyjnych

	funkcja aproksymowana			twierdzenie	autor
	dziedzina $\mathcal{U} \in \mathbb{R}^n$	własności	odległość funkcji		
@ funkcji ciągłych	domknięty, ograniczony	$\mathcal{C}$	$\sup \lvert f - \widehat{f} \rvert$	g ograniczona, $\mathcal{C}$, $\neq$ const $\Rightarrow$ @	Hornik 1991
@ funkcji ciągłych	domknięty, ograniczony	$\mathcal{C}$	$\operatorname{ess\,sup} \lvert f - \widehat{f} \rvert$	g lokalnie istotnie ograniczona, domknięcie zbioru p. nieciągłości ma miarę $0 \Rightarrow$ (g nie jest wielomianem $\Leftrightarrow$ @)	Leshno 1993
@ wraz z pochodnymi funkcji ciągłych	domknięty, ograniczony	$\mathcal{C}_d$	$\max\limits_{k \le d} \sup \lvert D^k f - D^k \widehat{f} \rvert$	g ograniczona, $\mathcal{C}_d$, $\neq$ const $\Rightarrow$ @	HSW 1990
@ funkcji całkowalnych	ograniczony	$\mathcal{L}_p$	$\lVert f - \widehat{f} \rVert_p$	g ograniczona, $\neq$ const $\Rightarrow$ @	Hornik 1991
@ wraz z pochodnymi funkcji całkowalnych	ograniczony	$\mathcal{L}_p$, $\mathcal{C}_d$	$\max\limits_{k \le d} \lVert D^k f - D^k \widehat{f} \rVert_p$	g ograniczona, $\mathcal{C}_d$, $\neq$ const $\Rightarrow$ @	HSW 1990
@ dowolnych funkcji	ograniczony, domknięty	dowolne (mierzalne)	$\sup\limits_{\mathbf{u} \in \mathcal{U}_\epsilon} \lvert f - \widehat{f} \rvert$, $\mu_L(\mathcal{U} - \mathcal{U}_\epsilon) < \epsilon$	g ograniczona, $\mathcal{C}$, $\neq$ const $\Rightarrow$ @	Hornik 1991
@ funkcji losowych	ograniczony	$\mathcal{E}\lvert f(\mathbf{u}) \rvert^p < \infty$	$\mathcal{E}\lvert f - \widehat{f} \rvert^p$	g ograniczona, $\neq$ const $\Rightarrow$ @	Hornik 1991

@- aproksymacja, HSW- Hornik, Stinchcombe, White, 1990

8. Techniki minimalizacji a sieci neuronowe

Bibliografia

[0-1] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W. T. Vetterling, *Numerical Recipies. The Art of Scientific Computing*, Cambridge University Press, Cambridge 1986, 1992

[0-2] K.W. Brodlie, Ch. III.1., Secs. 3–6 in *The State of the Art in Numerical Analysis*, D.A.H. Jacobs (Ed.), Academic Press, London 1977

[0-3] J.A. Freeman and D.M. Skapura, *Neural Networks : algorithms, applications, and programming techniques*, Addison-Wesley, Reading, Massachusetts, USA, 1991

[0-4] P. Werbos, *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*, Wiley, New York 1994

Błąd aproksymacji

- skończony zbiór obrazów $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_N\}$, $\mathcal{U}_L = \mathcal{U}$

$$Q = \frac{1}{2}\,\overline{\|\boldsymbol{\varepsilon}_i\|^2}$$

- $\mathcal{U}$ dowolny, $\mathcal{U}_L$ losowany zgodnie ze znanym rozkładem na $\mathcal{U}$

$$Q_0 = \frac{1}{2}\,\mathcal{E}\|\boldsymbol{\varepsilon}\|^2$$

- $\mathcal{U}$ dowolny, $\mathcal{U}_L$ losowany zgodnie z pewnym rozkładem na $\mathcal{U}$

$$\widehat{Q}_N(t) = \frac{1}{2}\,\mathcal{A}_N\|\boldsymbol{\varepsilon}(t)\|^2 \approx Q_0$$

Minimalizacja kosztu

- Funkcja błędu

$$\widehat{Q}_N(\mathbf{w}) = \frac{1}{N} \sum_{t=k-N+1}^{k} q(\mathbf{y}(t;\mathbf{w}) - \mathbf{y}^*(t)) = \frac{1}{N} \sum_{t=k-N+1}^{k} \|\mathbf{y}(t;\mathbf{w}) - \mathbf{y}^*(t)\|^2$$

- Metody gradientowe

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta\, r(\boldsymbol{\delta}(k))$$

gdzie

$\mathbf{w}$ wektor wszystkich wag sieci

$\boldsymbol{\delta}(k) = \widehat{Q}'_N(\mathbf{w}(k))$ gradient $\widehat{Q}_N$ w k-tym kroku minimalizacji

$\mathbf{H}(k) = \widehat{Q}''_N(\mathbf{w}(k))$ hessian $\widehat{Q}_N$ w k-tym kroku minimalizacji

r wektorowa funkcja gradientu określająca nowy kierunek

η krok minimalizacji

k indeks kroku minimalizacji

Tryby minimalizacji

(N, M, L): długość okna (przedział uśredniania) N, przesunięcie okna po wykonaniu kroków minimalizacji M, liczba kroków minimalizacji dla jednej estymaty gradientu

tryb natychmiastowy $(1, 1, 1)$

koszt oczekiwany estymowany jest przez koszt chwilowy (1-elementowe uśrednianie kosztu), minimalizacja po każdej prezentacji

tryb wsadowy $(N, N, 1)$

uśrednienie $N > 1$ kosztów chwilowych przed każdą minimalizacją; dla skończonego $\mathcal{U}_L$ przyjmuje się $N = |\mathcal{U}_L|$ (epoka)

wielokrotne użycie gradientu (N, N, L)

prezentacja N nowych wejść, $L > 1$ kroków minimalizacji

ruchome okno (N, M, L)

przesunięcie okna o długości N o $M < N$ chwil (usunięcie M najstarszych i prezentacja M nowych wejść), uśrednianie w oknie o długości N, wykonanie $L \geq 1$ kroków minimalizacji

Metoda największego spadku

- wzór Taylora 1 rzędu

$$Q(\mathbf{w} + \mu\,\mathbf{r}) = Q(\mathbf{w}) + \mu\,\mathbf{r}^T Q'(\mathbf{w}) + o(\mu)$$

- dla $\mathbf{r} = -Q'(\mathbf{w})$

$$Q(\mathbf{w} + \mu\,\mathbf{r}) = Q(\mathbf{w}) - \mu\,\|Q'(\mathbf{w})\|^2 < Q(\mathbf{w})$$

- algorytm korekcji wag

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\,\boldsymbol{\delta}(k)$$

- algorytm jest zwykle połączony z obliczaniem gradientu metodą propagacji zwrotnej

Gradient sprzężony

- funkcja kwadratowa w $\mathbb{R}^d$, minimalizacja w kierunku
  - minimalizacja($\mathbf{r}$) $*$ minimalizacja($\mathbf{s}$) $\neq$ minimalizacja($\mathbf{r}, \mathbf{s}$)
  - zmiana gradientu o $\mu\, Q''(\mathbf{w})\, \mathbf{r}(k)$
- *kierunki* $\mathbf{r}$, $\mathbf{s}$ *sprzężone*: $\mathbf{r}^T Q''\, \mathbf{s} = \mathbf{0}$
- *metoda gradientu sprzężonego* $\mathbf{r}(k)^T \mathbf{H}(k)\, \mathbf{s}(s) = 0$ dla $s = 1, \ldots, k-1$
- $\mathbf{r}(k) = -\boldsymbol{\delta}(k) + \beta(k-1)\, \mathbf{r}(k-1), \qquad \mathbf{r}(0) = -\boldsymbol{\delta}(0)$

$$\beta(k) = \frac{\|\boldsymbol{\delta}(k+1)\|^2}{\|\boldsymbol{\delta}(k)\|^2} \qquad \text{Fletcher-Reeves}$$

$$\beta(k) = \frac{(\boldsymbol{\delta}(k+1) - \boldsymbol{\delta}(k))^T\ \boldsymbol{\delta}(k+1)}{\|\boldsymbol{\delta}(k)\|^2} \qquad \text{Polak-Ribière}$$

  restart po d krokach

- zbieżne superliniowo; praktycznie: liniowo
- dla funkcji kwadratowej: F-R, P-R identyczne, zbieżność w d krokach

Metoda Newtona

- wzór Newtona 2 rzędu

$$Q(\mathbf{w}+\mu\,\mathbf{r}) = Q(\mathbf{w}) + \mu\,\mathbf{r}^T Q'(\mathbf{w}) + \frac{1}{2}\,\mu^2\,\mathbf{r}^T Q''(\mathbf{w})\,\mathbf{r} + o(\mu^2)$$

- gradient Q jako funkcja $\mathbf{r}$

$$\mu\,Q'(\mathbf{w}+\mu\,\mathbf{r}) = \mu\,Q'(\mathbf{w}) + \mu^2\,Q''(\mathbf{w})\,\mathbf{r} + o(\mu^2)$$

- kierunek poprawy

$$\mathbf{r} = -Q''(\mathbf{w})^{-1} Q'(\mathbf{w})$$

- *algorytm Newtona*

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\,\mathbf{H}^{-1}(k)\,\boldsymbol{\delta}(k)$$

- zbieżny w d krokach dla funkcji kwadratowej przy minimalizacji w kierunku; problemy: Hessian, wolny przebieg lub rozbieżność z dala od ekstremum

Metody zmiennej metryki

- aproksymacja Q'' powinna spełniać

$$\widehat{Q}''(\mathbf{w})(\mathbf{w}-\mathbf{w}^*) = Q'(\mathbf{w}) - Q'(\mathbf{w}^*)$$

- metoda *Davidona-Fletchera-Powella* (DFP)

$$\Delta P(k) = \frac{\Delta\mathbf{w}(k)\,\Delta\mathbf{w}(k)^T}{\Delta\mathbf{w}(k)^T\,\Delta\boldsymbol{\delta}(k)} - \frac{P(k)\,\Delta\boldsymbol{\delta}(k)\,\Delta\boldsymbol{\delta}(k)^T\,P(k)}{\Delta\boldsymbol{\delta}(k)^T\,P(k)\,\Delta\boldsymbol{\delta}(k)}$$

- metoda *Broydena-Fletchera-Goldfarba-Shanno* (BFGS)

$$\Delta P(k) = \Delta P(k)_{\mathsf{DFP}} + \Delta\boldsymbol{\delta}(k)^T P(k)\Delta\boldsymbol{\delta}(k)\,\mathbf{z}\,\mathbf{z}^T$$

$$\text{gdzie } \mathbf{z} = \frac{\Delta\mathbf{w}(k)}{\Delta\mathbf{w}(k)^T\Delta\boldsymbol{\delta}(k)} - \frac{P(k)\,\Delta\boldsymbol{\delta}(k)}{\Delta\boldsymbol{\delta}(k)^T P(k)\,\Delta\boldsymbol{\delta}(k)}$$

gdzie $P(k) = \widehat{\mathbf{H}}^{-1}(k)$, $\Delta\mathbf{x}(k) = \mathbf{x}(k+1) - \mathbf{x}(k)$

- warunki początkowe: $P(0) = \mathbf{1}$, $P(1)$ – metoda największego spadku

Metoda Levenberga-Marquardta

- ważenie metody największego spadku i metody Newtona

$$r(\boldsymbol{\delta}) = (\mathbf{H} + \lambda\,\mathbf{I})^{-1}\boldsymbol{\delta}$$

parametr Marquardta λ zmniejszany w czasie minimalizacji do 0

λ "duże" (daleko od ekstremum) – metoda największego spadku

λ "małe" (w pobliżu ekstremum) – metoda Newtona

- kwadratowa funkcja błędu

$$Q(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}(y_i(\mathbf{w}) - y_i^*)^2 = \frac{1}{2}\sum_{i=1}^{m}\varepsilon_i^2(\mathbf{w})$$

$$Q'(\mathbf{w}) = \sum_{i=1}^{m} y_i'(\mathbf{w})\,\varepsilon_i(\mathbf{w})$$

$$Q''(\mathbf{w}) = \sum_{i=1}^{m} y_i''(\mathbf{w})\,\varepsilon_i(\mathbf{w}) + y_i'(\mathbf{w})\,{y_i'}^T(\mathbf{w})$$

- wokół minimum ε_i jest bliskie 0, tzn $Q''(\mathbf{w}) \approx \sum_{i=1}^{m} y_i'(\mathbf{w})\,{y_i'}^T(\mathbf{w})$

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\left(\sum_{i=1}^{m} y_i'(\mathbf{w})\,{y_i'}^T(\mathbf{w}) + \lambda\,\mathbf{I}\right)^{-1}\boldsymbol{\delta}(k)$$

Specyficzne techniki neuronowe: Wygładzanie inercyjne wag

- *człon inercyjny* (ang. *momentum term*) $\alpha\,\Delta\mathbf{w}(k-1)$

$$\Delta\mathbf{w}(k) = -\mu\,\boldsymbol{\delta}(k) + \alpha\,\Delta\mathbf{w}(k-1), \quad 0 < \alpha < 1$$

- człon inercyjny *kumuluje efekt* $\boldsymbol{\delta}(k)$ w kierunku *spadku* gradientu i *redukuje efekt zmian* gradientu
- dla **zadanego ciągu** gradientów $\{\boldsymbol{\delta}(1), \ldots, \boldsymbol{\delta}(k)\}$

$$\begin{aligned}\Delta\mathbf{w}(k) &= \frac{-\mu}{1-\alpha q^{-1}}\,\boldsymbol{\delta}(k) \qquad\qquad (q^{-1} \text{ — } \textit{operator opóźnienia jednostkowego})\\ &= -\mu\,(1+\alpha q^{-1}+\alpha^2 q^{-2}+\ldots)\,\boldsymbol{\delta}(k)\\ &= -\mu\,\big(\boldsymbol{\delta}(k)+\alpha\boldsymbol{\delta}(k-1)+\alpha^2\boldsymbol{\delta}(k-2)+\ldots\big)\end{aligned}$$

czyli gradient $\boldsymbol{\delta}(k)$ zastąpiony przez sumę ważoną gradientów ze współczynnikiem wygładzania α

$$\Delta\mathbf{w}(k) = -\mu\,F(q)\,\boldsymbol{\delta}(k), \qquad F(q) = \sum_{i=0}\alpha^i q^{-i}$$

Specyficzne techniki neuronowe: Reguła delta-delta

- pochodna kosztu względem współczynnika uczenia μ_i dla algorytmu $w_i(k+1) = w_i(k) - \mu_i(k)\,\delta_i(k)$

$$\frac{dQ}{d\mu_i(k)} = -\boldsymbol{\delta}_i(k)\,\delta_i(k-1)$$

- korekcja współczynnika uczenia

$$\mu_i(k+1) = \mu_i(k) + \gamma\,\delta_i(k)\,\delta_i(k-1)$$

- duża wrażliwość na wybór γ; wzrost μ_i gdy dwie kolejne pochodne $\delta_i(k)$ są tego samego znaku
- nazwa delta-delta wywodzi się z oznaczenia gradientu przez δ

Specyficzne techniki neuronowe: Reguła delta-bar-delta

- wygładzanie gradientu

$$\overline{\delta}(k) = \xi\,\overline{\delta}_i(k-1) + (1-\xi)\,\delta_i(k), \quad 0 < \xi < 1$$

$$\mu_i(k+1) = \mu_i(k) + \gamma\,\delta_i(k)\,\overline{\delta}_i(k-1)$$

- dodatkowe zabezpieczenia: μ liniowo rośnie gdy znak gradientu stały, maleje wykładniczo gdy zmienny

$$\mu_i(k+1) = \mu_i(k) + \begin{cases} \kappa & \text{gdy } \delta_i(k)\,\overline{\delta}_i(k-1) > 0 \\ -\beta\mu_i(k) & \text{gdy } \delta(k)\,\overline{\delta}(k-1) < 0 \\ 0 & \text{w pozostałych przypadkach} \end{cases}$$

$0 < \kappa < 0.05, \quad 0.1 < \beta < 0.3$

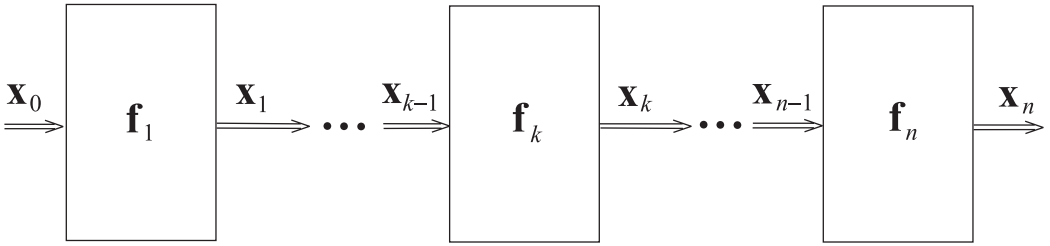
9. Propagacja zwrotna gradientu

Bibliografia

[0-1] P.J. Werbos "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Ph.D. Thesis, Harvard University, Cambridge, MA, 1974.

[0-2] D.E. Rummelhart, G.E. Hinton, and R.J. Williams "Learning internal representation by errror propagation," in *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*, D.E. Rummelhart and J.L. McClelland, Eds., vol. 1, Chap. 8, Cambridge, MA, MIT PRess, 1986

System warstwowy



$$f_1 : X_0 \mapsto X_1, \qquad x_1 = f_1(x_0)$$
$$\dots$$
$$f_k : X_{k-1} \mapsto X_k, \qquad x_k = f_k(x_{k-1})$$
$$\dots$$
$$f_n : X_{n-1} \mapsto X_n \qquad x_n = f_n(x_{n-1})$$

$$x_n = f_n(f_{n-1}(\dots(f_1(x_0))))$$

Pochodne dla systemu warstwowego

$$x_n = f_n(f_{n-1}(\ldots(f_1(x_0))))$$

- notacja

$$f'_k = \frac{df_k}{dx_{k-1}} \qquad \text{dla } i = 1, \ldots, n$$

$$x'_{k|\ell} = \frac{dx_k}{dx_\ell} \qquad \text{dla ustalonych } x_0, \ldots x_{\ell-1}, \quad \ell < k$$

- obliczenie pochodnej

$$\frac{dx_n}{dx_0} = f'_n(x_{n-1})\, f'_{n-1}(x_{n-2}) \ \ldots \ f'_2(x_1)\, f'_1(x_0)$$

gdzie

$$x_k = f_k(f_{k-1} \cdots f_1(x_0)), \quad k = 1, \ldots, n$$

Propagacja prosta gradientu

- grupowanie "od wejścia do wyjścia"

$$x'_{n|0} = \frac{dx_n}{dx_0} = f'_n(x_{n-1})\; \underbrace{f'_{n-1}(x_{n-2}) \ldots \underbrace{f'_2(x_1)\; \underbrace{f'_1(x_0)}_{x'_{1|0}}}_{x'_{2|0}}}_{x'_{n-1|0}}$$

- algorytm propagacji prostej gradientu

$$\begin{aligned} x'_{0|0} &= 1 \\ x'_{k|0} &= f'_k(x_{k-1})\; x'_{k-1|0} \\ x_k &= f_k(x_{k-1}) \qquad k = 1, \ldots, n \end{aligned}$$

- pochodne i wartości funkcji obliczane są w jednym kroku rekurencyjnym, dla $k = 1, \ldots, n$.

Propagacja zwrotna gradientu

- grupowanie "od wyjścia do wejścia"

$$x'_{n|0} = \frac{dx_n}{dx_0} = \underbrace{\underbrace{\underbrace{f'_n(x_{n-1})}_{x'_{n|n-1}}\, f'_{n-1}(x_{n-2})}_{x'_{n|n-2}} \ldots f'_2(x_1)}_{x'_{n|1}}\, f'_1(x_0)$$

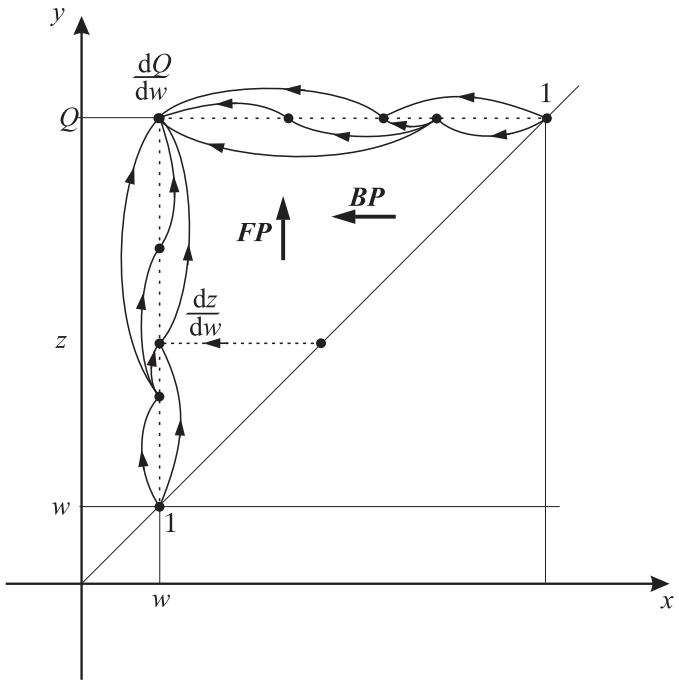
- algorytm propagacji zwrotnej
  - przejście "do przodu": wartości zmiennych

$$x_k = f_k(x_{k-1}), \qquad k = 1, \ldots, n$$

  - powrót: wartości pochodnych

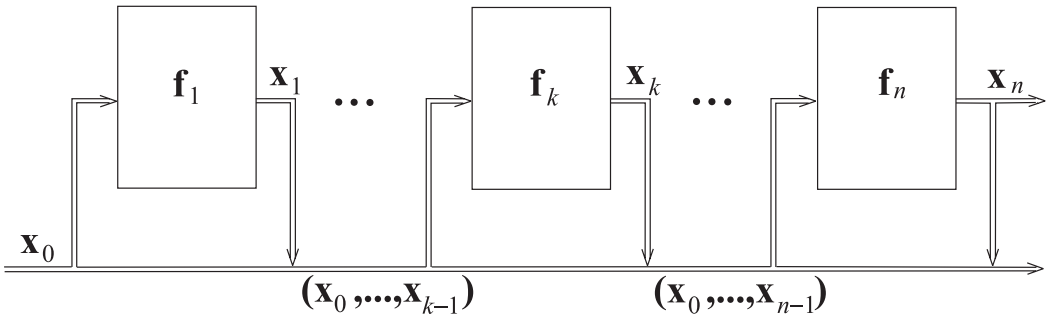
$$x'_{n|n} = 1$$
$$x'_{n|\ell} = x'_{n|\ell+1}\, f'_{\ell+1}(x_\ell), \qquad \ell = n-1, \ldots, 0$$

Porównanie



- propagacja gradientu (FP): pochodne zmiennych pośrednich *względem docelowej zmiennej niezależnej*
- propagacja zwrotna gradientu (BP): pochodne *docelowej zmiennej zależnej* względem zmiennych pośrednich
- obliczenia w sieci: potrzebne pochodne *wskaźnika błędu* względem różnych zmiennych.

System uporządkowany



$$f_1 : X_0 \mapsto X_1 \qquad x_1 = f_1(x_0)$$

$$\dots$$

$$f_k : X_0 \times \cdots \times X_{k-1} \mapsto X_k \qquad x_k = f_k(x_0, \dots, x_{k-1})$$

$$\dots$$

$$f_n : X_0 \times \cdots\cdots \times X_{n-1} \mapsto X_n \qquad x_n = f_n(x_0, \dots\dots, x_{n-1})$$

Pochodne dla systemów uporządkowanych

- oznaczenia (tutaj $x_k \in \mathbb{R}$)

$$f'_{k|\ell}(x_0, \ldots, x_{k-1}) = \frac{\partial}{\partial x_\ell} f_k(x_0, \ldots, x_{k-1}), \quad 0 \le \ell < k$$

- Dla pierwszych ℓ zmiennych $x_0, \ldots, x_{\ell-1}$, $\ell < k$, ustalonych, definiujemy

$$x'_{k|\ell} = \frac{dx_k}{dx_\ell}$$

- zależności równoważne

$$x'_{k|\ell} = f'_{k|\ell} + \sum_{i=\ell+1}^{k-1} f'_{k|i}\, x'_{i|\ell} \quad 0 \le \ell < k \le n \qquad \text{(FP)}$$

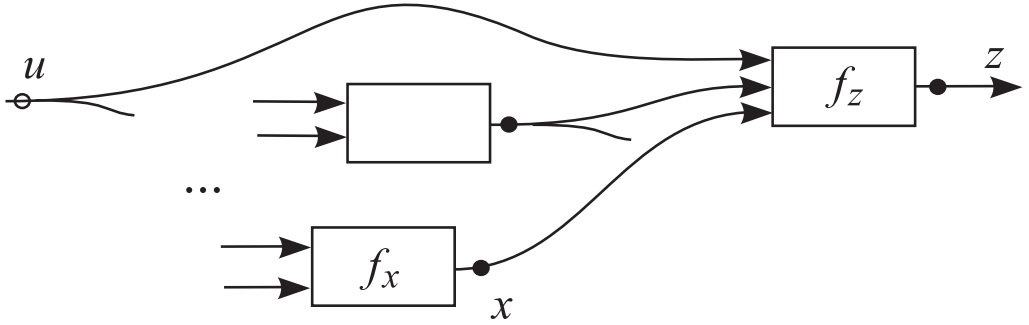
$$x'_{k|\ell} = f'_{k|\ell} + \sum_{j=\ell+1}^{k-1} x'_{k|j}\, f'_{j|\ell} \quad 0 \le \ell < k \le n \qquad \text{(BP)}$$

Propagacja prosta gradientu - postać lokalna

- dla dowolnych dwóch zmiennych $u = x_k$, $z = x_\ell$, $\ell < k$

$$\frac{dz}{du} = \frac{\partial f_z}{\partial u} + \sum_x \frac{\partial f_z}{\partial x} \frac{dx}{du}$$

gdzie $f_z = f_\ell$ jest funkcją definiującą z, a sumowanie rozciąga się na wszystkie zmienne x *które bezpośrednio wpływają na* z (tzn. argumenty f_z)

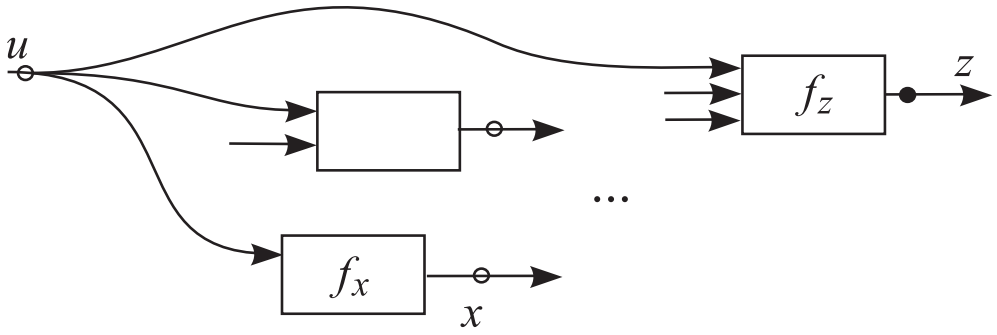


Propagacja zwrotna gradientu - postać lokalna

- dla dowolnych dwóch zmiennych $u = x_k$, $z = x_\ell$, $\ell < k$

$$\frac{dz}{du} = \frac{\partial f_z}{\partial u} + \sum_x \frac{dz}{dx}\,\frac{df_x}{du}$$

gdzie f_x oznacza funkcję która definiuje x, a sumowanie rozciąga się na wszystkie zmienne x *na które bezpośrednio wpływa* u (wszystkie funkcje, dla których u jest jednym z argumentów.)



Ilustracja (system warstwowy)

$$y = \sin(\exp(x^3 \cos(x^3)))$$

System warstwowy $(x_0 = x,\ x_4 = y)$

$$\begin{aligned} x_1 &= x_0^3 \\ x_2 &= x_1 \cos(x_1) \\ x_3 &= \exp(x_2) \\ x_4 &= \sin(x_3) \end{aligned}$$

propagacja prosta

$$\begin{aligned} x'_{4|0} &= \cos(x_3)\, x'_{3|0} \\ x'_{3|0} &= \exp(x_2)\, x'_{2|0} \\ x'_{2|0} &= \big(\cos(x_1) - x_1\ \sin(x_1)\big)\, x'_{1|0} \\ x'_{1|0} &= 3\, x_0^2\, x'_{0|0} \\ x'_{0|0} &= 1 \end{aligned}$$

propagacja zwrotna

$$\begin{aligned} x'_{4|0} &= 3\, x'_{4|1}\, x_0^2 \\ x'_{4|1} &= \big(\cos(x_1) - x_1\ \sin(x_1)\big)\, x'_{4|2} \\ x'_{4|2} &= x'_{4|3}\ \exp(x_2) \\ x'_{4|3} &= x'_{4|4}\ \cos(x_3) \\ x'_{4|4} &= 1 \end{aligned}$$

Ilustracja (wielowymiarowa)

$$y = \sin\bigl(\exp(x^3) + 4\cos(x^3)\bigr)$$

system warstwowy
$(x_0 = x,\ x_3 = y)$

$$\begin{aligned} x_1 &= x^3 \\ x_{21} &= \exp(x_1) \\ x_{22} &= \cos(x_1) \\ x_3 &= \sin(x_{21} + 4x_{22}) \end{aligned}$$

propagacja gradientu

$$\begin{aligned} x'_{3|0} &= \cos(x_{21}+4x_{22})\bigl(x'_{21|0} + 4x'_{22|0}\bigr) \\ x'_{21|0} &= \exp(x_1)\, x'_{1|0} \\ x'_{22|0} &= -\sin(x_1)\, x'_{1|0} \\ x'_{1|0} &= 3\, x^2 \end{aligned}$$

propagacja zwrotna gradientu

$$\begin{aligned} x'_{3|0} &= 3\, x'_{3|1}\, x^2 \\ x'_{3|1} &= x'_{3|22}\, \exp(x_1) - x'_{3|21} \sin(x_1) \\ x'_{3|21} &= \cos(x_{21} + x_{22}) \\ x'_{3|22} &= 4\, \cos(x_{21} + x_{22}) \end{aligned}$$

Ilustracja (system uporządkowany

system uporządkowany

$$Q = x_3 = f_3(x_0, x_1, x_2)$$
$$x_2 = f_2(x_0, x_1)$$
$$x_1 = f_1(x_0)$$

propagacja gradientu

$$x'_{3|0} = f'_{3|0} + f'_{3|1}\, x'_{1|0} + f'_{3|2}\, x'_{2|0}$$
$$x'_{2|0} = f'_{2|0} + f'_{2|1}\, x'_{1|0}$$
$$x'_{1|0} = f'_{1|0}$$

propagacja zwrotna gradientu

$$x'_{3|0} = f'_{3|0} + x'_{3|2}\, f'_{2|0} + x'_{3|1}\, f'_{1|0}$$
$$x'_{3|1} = f'_{3|1} + x'_{3|2}\, f'_{2|1}$$
$$x'_{3|2} = f'_{3|2}$$

dla obu algorytmów

$$x'_{3|0} = f'_{3|0} + f'_{3|1}\, f'_{1|0} + f'_{3|2}\, f'_{2|0} + f'_{3|2}\, f'_{2|1}\, f'_{1|0}$$

Postać macierzowa

$$\mathbf{F}' = \{f'_{i|j}\}_{\substack{i=0,\dots,n \\ j=0,\dots,n}} = \begin{bmatrix} 0 & & & \\ f'_{1|0} & 0 & & \\ \vdots & \vdots & \ddots & \\ f'_{n|0} & f'_{n|1} & \cdots f'_{n|n-1} & 0 \end{bmatrix}, \quad \mathbf{X}' = \{x'_{i|j}\}_{\substack{j=0,\dots,n \\ i=0,\dots,n}} = \begin{bmatrix} 1 & & & \\ x'_{1|0} & 1 & & \\ \vdots & \vdots & \ddots & \\ x'_{n|0} & x'_{n|1} & \cdots x'_{n|n-1} & 1 \end{bmatrix}$$

dla systemów warstwowych

$$\mathbf{F}' = \begin{bmatrix} 0 & & & \\ f'_1 & 0 & & \\ & \ddots & \ddots & \\ & & f'_n & 0 \end{bmatrix}$$

propagacja gradientu $(\mathbf{F}' - I)\,\mathbf{X}' = -I$

propagacja zwrotna gradientu $\mathbf{X}'\,(\mathbf{F}' - I) = -I$

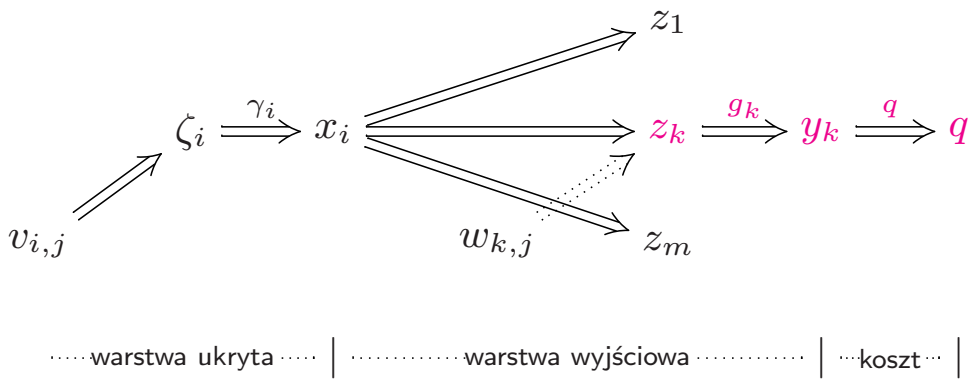
10. Propagacja zwrotna w perceptronie wielowarstwowym

Gradienty dla sieci dwuwarstwowej: graf wpływów

- równania warstw

warstwa ukryta	$x_i = \gamma_i(\zeta_i), \quad \zeta_i = \sum_{j=0}^{n} v_{i,j}\, u_j, \quad i = 1, \ldots, h$
warstwa wyjściowa	$y_k = g_k(z_k), \quad z_k = \sum_{j=0}^{h} w_{k,j}\, x_j, \quad k = 1, \ldots, m$
koszt chwilowy	$q = q(y_1, \ldots, y_m)$

- grafy wpływów dla v i w



- grafy dla v i w mają część wspólną

Gradienty dla sieci dwuwarstwowej: warstwa wyjściowa

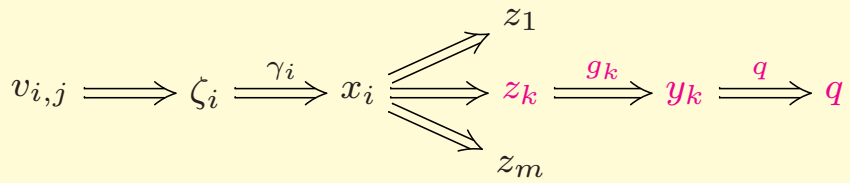
$$w_{k,j} \Longrightarrow z_k \overset{g_k}{\Longrightarrow} y_k \overset{q}{\Longrightarrow} q$$

$$\left.\begin{aligned} \frac{dq}{dw_{k,j}} &= \frac{dq}{dz_k}\, x_j \\ \frac{dq}{dz_k} &= \frac{dq}{dy_k}\, g_k{}'(z_k) \end{aligned}\right\} \text{BP od wyjścia warstwy do wag}$$

$$\left.\frac{dq}{dy_k} = q_k'(\mathbf{y})\right\} \text{wskaźnik kosztu}$$

$$r_k = q_k'(\mathbf{y})\, g_k{}'(z_k), \qquad \frac{dq}{dw_{k,j}} = r_k\, x_j$$

Gradienty dla sieci dwuwarstwowej: warstwa ukryta



warstwa ukryta | warstwa wyjściowa | koszt |

$$\frac{dq}{dv_{i,j}} = \frac{dq}{d\zeta_i}\, u_j$$

$$\frac{dq}{d\zeta_i} = \frac{dq}{dx_i}\, \gamma_i{}'(\zeta_i)$$

} BP w warstwie do wag

$$\frac{dq}{dx_i} = \sum_{k=1}^{m} \frac{dq}{dz_k}\, w_{k,i}$$

$$\frac{dq}{dz_k} = \frac{dq}{dy_k}\, g_k{}'(z_k)$$

} BP przez warstwę

$$\frac{dq}{dy_k} = \frac{\partial q(\mathbf{y})}{\partial y_k} = q'_k(\mathbf{y})$$

} wskaźnik kosztu

$$r_k = q'_k(\mathbf{y})\, g_k{}'(z_k), \qquad \varrho_i = \sum_{k=1}^{m} r_k\, w_{k,i}\, \gamma_i{}'(\zeta_i), \qquad \frac{dq}{dv_{i,j}} = \varrho_i\, u_j$$

BP gradientu dla sieci dwuwarstwowej

(warstwa wyjściowa)

$$r_i = q_i'(\mathbf{y})\, g_i{}'(z_i)$$

$$\frac{dq}{dw_{i,j}} = r_i\, x_j$$

(warstwa ukryta)

$$\varrho_i = \sum_{k=1}^{m} r_k\, w_{k,i}\, \gamma_i{}'(\zeta_i)$$

$$\frac{dq}{dv_{i,j}} = \varrho_i\, u_j$$

(warstwa wyjściowa)

$$\mathbf{r} = q'(\mathbf{y}) \odot \mathbf{g}'(\mathbf{z})$$

$$\frac{dq}{d\mathbf{W}} = \mathbf{r}\,\mathbf{x}^T$$

(warstwa ukryta)

$$\boldsymbol{\varrho} = (\mathbf{W}^T\mathbf{r}) \odot \boldsymbol{\gamma}'(\boldsymbol{\zeta})$$

$$\frac{dq}{d\mathbf{V}} = \boldsymbol{\varrho}\,\mathbf{u}^T$$

Przykład: BP gradientu dla standardowej sieci dwuwarstwowej

warstwa wyjściowa liniowa, chwilowy koszt kwadratowy $q = \frac{1}{2}\|\mathbf{y} - \mathbf{y}^*\|^2$,
funkcje aktywacji $\gamma_i(\zeta) = 1/(1+\exp(-\alpha\zeta))$, $[\gamma_i'(\zeta) = \alpha\, x(1-x)]$

(warstwa wyjściowa)

$$r_i = y_i - y_i^*$$

$$\frac{dq}{dw_{i,j}} = r_i\, x_j$$

(warstwa ukryta)

$$\varrho_i = \alpha \sum_{k=1}^{m} r_k\, w_{k,i}\, x_i\, (1 - x_i)$$

$$\frac{dq}{dv_{i,j}} = \varrho_i\, u_j$$

(warstwa wyjściowa)

$$\mathbf{r} = \mathbf{y} - \mathbf{y}^*$$

$$\frac{dq}{d\mathbf{W}} = \mathbf{r}\,\mathbf{x}^T$$

(warstwa ukryta)

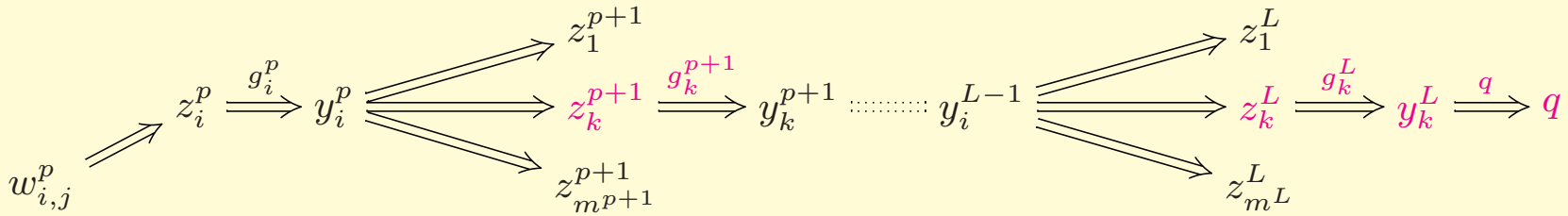
$$\boldsymbol{\varrho} = \alpha\,(\mathbf{W}^T\mathbf{r}) \odot \mathbf{x} \odot (1 - \mathbf{x})$$

$$\frac{dq}{d\mathbf{V}} = \boldsymbol{\varrho}\,\mathbf{u}^T$$

Gradienty dla sieci wielowarstwowej

warstwa ℓ, $\ell = 1, \ldots, L$ $\quad z_i^\ell = \sum_{j=0}^{m^{\ell-1}} w_{i,j}^\ell \, y_j^{\ell-1}, \quad y_i^\ell = g_i^\ell(z_i^\ell), \quad i = 1, \ldots, m^\ell$

koszt $\quad q = q(y_1^L, \ldots, y_{m^L}^L)$



warstwa p | warstwa $p+1$ | $\cdots$ | warstwa wyjściowa L | koszt

$$\frac{dq}{dw_{i,j}^p} = \frac{dq}{dz_i^p} \, u_j^p$$

$$\frac{dq}{dz_i^p} = \frac{dq}{dy_i^p} \, g_i^{p\prime}(z_i^p)$$

BP w warstwie p do wag

$$\frac{dq}{dy_i^{\ell-1}} = \sum_{k=1}^{m^\ell} \frac{dq}{dz_k^\ell} \, w_{k,i}^\ell$$

$$\frac{dq}{dz_k^\ell} = \frac{dq}{dy_k^\ell} \, g_k^{\ell\prime}(z_k^\ell)$$

BP przez warstwę ℓ, $\quad \ell = p+1, \ldots, L$

$$\frac{dq}{dy_k^L} = \frac{\partial q(\mathbf{y})}{\partial y_k^L} = q_k'(\mathbf{y})$$

wskaźnik kosztu

BP gradientu dla sieci wielowarstwowej

$$\frac{dq}{dy_p} = \frac{\partial q(\mathbf{y})}{\partial y_p} = q'_p(\mathbf{y})$$

$$\frac{dq}{dy_i^{\ell-1}} = \sum_{k=1}^{m^\ell} \frac{dq}{dy_k^\ell}\, g_k^{\ell\,\prime}(z_k^\ell)\, w_{k,i}^\ell$$

$$\ell = L, \ldots, p+1$$

$$\frac{dq}{dw_{i,j}^p} = \frac{dq}{dy_i^p}\, g_i^{p\prime}(z_i^p)\, u_j^p$$

wskaźnik jakości

BP przez warstwę

$\ell = L, \ldots, p+1$

BP w warstwie p do wag

$$\frac{dq}{d\mathbf{y}} = \frac{\partial q(\mathbf{y})}{\partial \mathbf{y}} = q'(\mathbf{y})$$

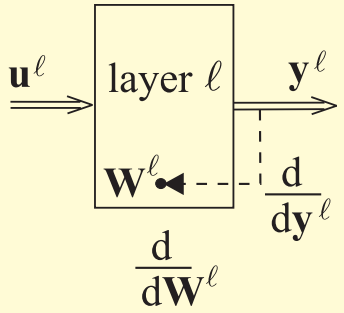
$$\frac{dq}{d\mathbf{y}^{\ell-1}} = \mathbf{W}^{\ell\,T}\left(\frac{dq}{d\mathbf{y}^\ell} \odot \mathbf{g}^{\ell\,\prime}(\mathbf{z}^\ell)\right)$$

$$\ell = L, \ldots, p+1$$

$$\frac{dq}{d\mathbf{W}^p} = \left(\frac{dq}{d\mathbf{y}^p} \odot \mathbf{g}^{p\,\prime}(\mathbf{z}^p)\right) \mathbf{u}^{p\,T}$$

Operator BP

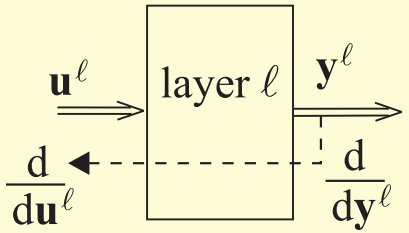
BP do wag warstwy



$$\nabla_{\mathbf{W}} = \mathcal{B}_{\mathbf{W}}(\nabla_{\mathbf{y}}) = \Big(\nabla_{\mathbf{y}} \odot \mathbf{g}'(\mathbf{z})\Big)\,\mathbf{u}^T$$

$$\nabla_{\mathbf{W}} = \frac{d\,\cdot}{d\mathbf{W}}$$
$$\nabla_{\mathbf{y}} = \frac{d\,\cdot}{d\mathbf{y}}$$
$$\nabla_{\mathbf{u}} = \frac{d\,\cdot}{d\mathbf{u}}$$

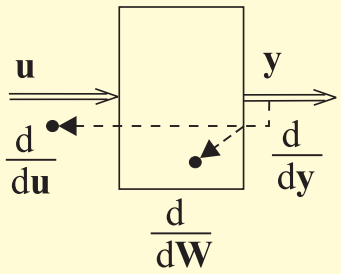
BP przez warstwę



$$\nabla_{\mathbf{u}} = \mathcal{B}_{\mathbf{u}}(\nabla_{\mathbf{y}}) = \mathbf{W}^T\Big(\nabla_{\mathbf{y}} \odot \mathbf{g}'(\mathbf{z})\Big)$$

moduł BP dla warstwy

$$(\nabla_{\mathbf{W}}, \nabla_{\mathbf{u}}) = \mathcal{B}(\nabla_{\mathbf{y}})$$

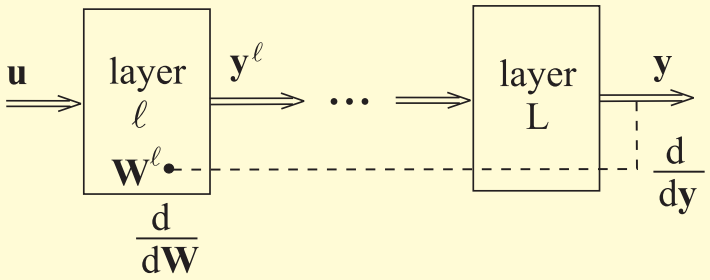


$$\mathbf{r} = \nabla_{\mathbf{y}} \odot \mathbf{g}'(\mathbf{z})$$

$$\nabla_{\mathbf{W}} = \mathbf{r}\,\mathbf{u}^T$$
$$\nabla_{\mathbf{u}} = \mathbf{W}^T\mathbf{r}$$

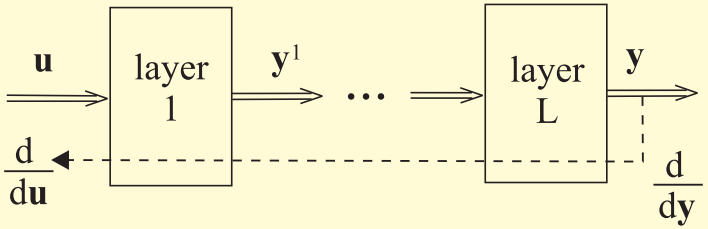
Propagacja zwrotna przez sieć

BP w sieci do wag



$$\nabla_{\mathbf{W}^\ell} = \mathcal{B}_{\mathbf{W}^\ell}\, \mathcal{B}_{\mathbf{u}^{\ell+1}} \ldots \mathcal{B}_{\mathbf{u}^L}\, \nabla_{\mathbf{y}^L}$$

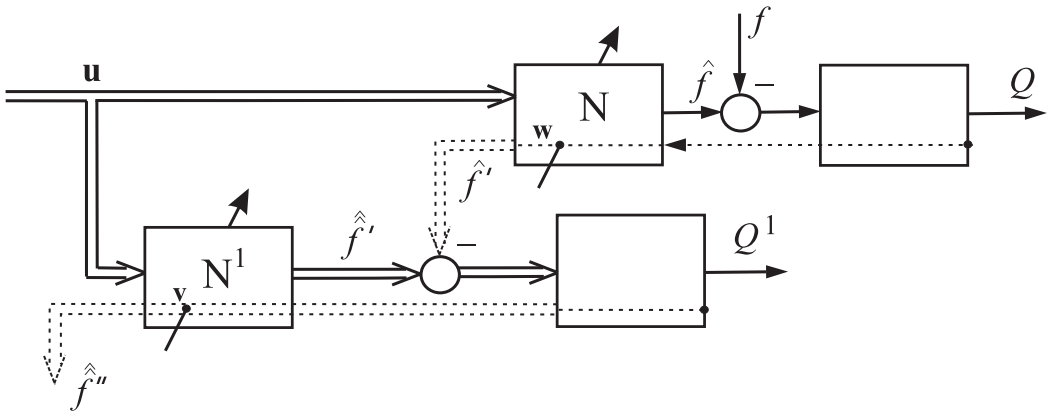
BP przez sieć



$$\nabla_{\mathbf{u}^1} = \mathcal{B}_{\mathbf{u}^1}\, \mathcal{B}_{\mathbf{u}^2} \ldots \mathcal{B}_{\mathbf{u}^L}\, \nabla_{\mathbf{y}^L}$$

Drugie pochodne

- $y^* = f(\mathbf{u})$ jest wielkością pożądaną dla sieci $\mathbf{N}$ o wejściu $\mathbf{u}$
- $\mathbf{N}$ aproksymuje funkcję f: $\mathbf{N}(\mathbf{u}) = \widehat{f}(\mathbf{u})$
- BP przez sieć $\mathbf{N}$ oblicza wartości gradientu funkcji $\widehat{f}$: $\widehat{f}'(\mathbf{u}) = \dfrac{d\mathbf{N}(\mathbf{u})}{d\mathbf{u}} \approx f'(\mathbf{u})$
- $\widehat{f}'(\mathbf{u})$ są wielkościami pożądanymi dla sieci $\mathbf{N}^1$ o wejściu $\mathbf{u}$
- $\mathbf{N}^1$ aproksymuje funkcję $\widehat{f}'$: $\mathbf{N}^1(\mathbf{u}) = \widehat{\widehat{f}}'$
- BP przez sieć $\mathbf{N}^1$ aproksymuje wartości aproksymowanego hessianu $\widehat{\widehat{f}}''$



11. Jakość aproksymacji funkcji

Bibliografia

[0-1] N. Cristiannini, J Shave-Taylor, *An Introduction to Support Vector Machines* Cambridge University Press, Cambridge, UK 2000

[0-2] S. Haykin, *Neural Networks. A Comprehensive Foundation. Second Edition*, Prentice Hall, Inc., Upper Saddle River, New Jersey, USA, 1999

Aproksymacja

- *aproksymator* $\widehat{f} : \mathcal{U} \mapsto \mathcal{Y}$,
  $\widehat{f} \in \widehat{\mathcal{F}}$ rodzina aproksymatorów
  - aproksymator neuronowy $\widehat{f}(\mathbf{u};\boldsymbol{\vartheta}) = \mathbf{N}_{\boldsymbol{\vartheta}}(\mathbf{u})$, $\boldsymbol{\vartheta} = (\mathbf{w}, \mathbf{b}) \in \mathbb{R}^d$
  - dla klasyfikacji: $\mathcal{Y} = \{0, 1\}$
- środowisko: dane $(\mathbf{u}, \mathbf{y})$ generowane zgodnie z **nieznanym** rozkładem $\mathcal{P}$ na $\mathcal{U} \times \mathcal{Y}$
  - $\mathcal{P}$ skupiony na krzywej $\{\mathbf{y} = f(\mathbf{u}),\ \mathbf{u} \in \mathcal{U}\}$: aproksymacja funkcji
  - $\mathcal{P}$ skupiony na $\mathcal{U} \times \{-1, 1\}$: klasyfikacja binarna
- *koszt* aproksymacji $q\big(\mathbf{y},\ \widehat{f}(\mathbf{u})\big)$
  - koszt średniokwadratowy: $q\big(\mathbf{y}, \widehat{f}(\mathbf{u})\big) = \|\mathbf{y} - \widehat{f}(\mathbf{u})\|^2$
  - dla klasyfikacji binarnej: $q\big(\mathbf{y}, \widehat{f}(\mathbf{u})\big) = 1 - \delta_{\mathbf{y},\widehat{f}(\mathbf{u})} = \begin{cases} 0 & \text{klasyfikacja poprawna} \\ 1 & \text{klasyfikacja błędna} \end{cases}$
- *ryzyko*: koszt oczekiwany $Q(\widehat{f}) = \mathcal{E} q\big(\mathbf{y}, \widehat{f}(\mathbf{u})\big) = \int q\big(\mathbf{y}, \widehat{f}(\mathbf{u})\big)\, d\mathcal{P}(\mathbf{u}, \mathbf{y})$
  - dla kosztu średniokwadratowego: $Q(\widehat{f}) = \mathcal{E}\|\mathbf{y} - \widehat{f}(\mathbf{u})\|^2$
  - dla klasyfikacji $Q(\widehat{f}) = \pi(\widehat{f}) = \mathcal{P}\{\mathbf{y} \neq \widehat{f}(\mathbf{u})\}$

Budowa aproksymatora

- *zbiór trenujący* $\mathcal{S}_N = \{(\mathbf{u}_1, \mathbf{y}_1), \ldots, (\mathbf{u}_N, \mathbf{y}_N)\}$ o elementach generowanych niezależnie zgodnie z $\mathcal{P}$ (*próba losowa prosta, próba i.i.d.*)
- *metoda aproksymacji*: algorytm doboru *aproksymatora* $\widehat{f}$ z rodziny funkcji $\widehat{\mathcal{F}}$ na podstawie zbioru trenującego $\mathcal{S}_N$
- *aproksymacja parametryczna*: dobór aproksymatora z zadanej rodziny funkcji $\widehat{\mathcal{F}} = \{\widehat{f}_{\boldsymbol{\vartheta}}, \boldsymbol{\vartheta} \in \Theta \subset \mathbb{R}^n\}$ (np. sieć neuronowa); sprowadza się do doboru (estymacji) parametru $\boldsymbol{\vartheta}$

Ryzyko empiryczne

- *ryzyko empiryczne*: koszt średni $Q_{\mathcal{S}_N}(\widehat{f}) = \mathcal{A}_N q\big(\mathbf{y}, \widehat{f}(\mathbf{u})\big) = \frac{1}{N}\sum_{i=1}^{N} q\big(\mathbf{y}_i, \widehat{f}(\mathbf{u}_i)\big)$
  - dla kosztu średniokwadratowego $Q_{\mathcal{S}_N}(\widehat{f}) = \mathcal{A}_N \|\mathbf{y} - \widehat{f}(\mathbf{u})\|^2$
  - dla klasyfikacji $Q_{\mathcal{S}_N}(\widehat{f}) = \widehat{\pi}_N(\widehat{f}) =$ częstość błędu
- metoda estymacji: *minimalizacja ryzyka empirycznego*
  aproksymator $\widehat{f}_N$ **jest wielkością losową** (losowe dane uczące $\mathcal{S}_N$)
- *(a)* czy $Q_N(f) \underset{N\to\infty}{\longrightarrow} Q(f)$ dla każdej funkcji f ? **uwaga: nie implikuje (b)**

  (b) czy $Q(\widehat{f}_N) \underset{N\to\infty}{\longrightarrow} Q(\widehat{f})$? **np. dla kosztów kwadratowych implikuje (c)**

  (c) czy $\widehat{f}_N \underset{N\to\infty}{\longrightarrow} \widehat{f}$?
- *błąd generalizacji* $Q(\widehat{f}_N) = \mathcal{E} q\big(\mathbf{y}, \widehat{f}_N(\mathbf{u}) \,|\, \mathcal{S}_N\big)$ (**zmienna losowa**)
- aproksymacja jest *prawdopodobnie w przybliżeniu poprawna* (PAC - *probably approximately correct*) jeśli z prawdopodobieństwem $> 1-\delta$ błąd generalizacji jest $< \epsilon$

$$P\{Q(\widehat{f}_N) < \epsilon(N, \delta, \widehat{\mathcal{F}})\} > 1 - \delta$$

Minimalizacja ryzyka i ryzyka empirycznego

(a) przy "łagodnych" założeniach $Q_N(\boldsymbol{\vartheta}) \xrightarrow[N\to\infty]{P} Q(\boldsymbol{\vartheta})$ dla każdego $\boldsymbol{\vartheta}$

(np. $(\mathbf{u}_i, \mathbf{y}_i)$, $i = 1, \ldots, N$, niezależne, identyczne rozkłady)

czyli dla każdego $\boldsymbol{\vartheta} \in \mathbb{R}^d$, $\epsilon > 0$, $\delta > 0$ i dostatecznie dużego $N \geq N_0(\boldsymbol{\vartheta}, \epsilon, \delta)$

$$P\{|Q_N(\boldsymbol{\vartheta}) - Q(\boldsymbol{\vartheta})| > \epsilon\} \leq \delta$$

- N_0 zależy od $\boldsymbol{\vartheta}$ więc nie gwarantuje bliskości *wykresów* Q_N oraz Q
- **jeżeli zbieżność jest jednostajna** (N_0 nie zależy od $\boldsymbol{\vartheta}$) tzn. dla $N > N_0(\epsilon, \delta)$

$$P\{\sup_{\boldsymbol{\vartheta}} |Q_N(\boldsymbol{\vartheta}) - Q(\boldsymbol{\vartheta})| > \epsilon\} \leq \delta$$

zatem również $P\{Q_N(\boldsymbol{\vartheta}) - Q(\boldsymbol{\vartheta}) < -\epsilon\}$ i $P\{Q_N(\boldsymbol{\vartheta}) - Q(\boldsymbol{\vartheta}) > \epsilon\}$ dla każdego $\boldsymbol{\vartheta}$
czyli z prawdopodobieństwem $\leq \delta$

$$\begin{aligned} Q(\boldsymbol{\vartheta}^*_N) - Q_N(\boldsymbol{\vartheta}^*_N) &> \epsilon \quad (\text{"lewy ogon"}) \\ Q_N(\boldsymbol{\vartheta}^*) - Q(\boldsymbol{\vartheta}^*) &> \epsilon \quad (\text{"prawy ogon"}) \end{aligned}$$

dodając stronami i uwzględniając, że $Q_N(\boldsymbol{\vartheta}^*_N) \leq Q_N(\boldsymbol{\vartheta}^*)$ mamy

$$P\{Q(\boldsymbol{\vartheta}^*_N) - Q(\boldsymbol{\vartheta}^*) > 2\,\epsilon\} \leq \delta$$

(b), (c) $Q(\boldsymbol{\vartheta}^*_N) \xrightarrow[N\to\infty]{P} Q(\boldsymbol{\vartheta}^*)$ zatem dla kwadratowego kosztu $\boldsymbol{\vartheta}^*_N \xrightarrow[N\to\infty]{P} \boldsymbol{\vartheta}^*$

Klasyfikacja, skończona rodzina klasyfikatorów

- dla klasyfikacji

$$Q(\widehat{f}) = \mathcal{P}\{y \neq \widehat{f}(\mathbf{u})\}$$

$$Q_{\mathcal{S}_N}(\widehat{f}) = \frac{1}{N} \#\{(\mathbf{u},y) : y \neq \widehat{f}(\mathbf{u})\}$$

- zbiór klasyfikatorów $\widehat{\mathcal{F}} = \{\widehat{f}^1, \ldots, \widehat{f}^n\}$,
- prawdopodobieństwo dużego błędu generalizacji dla klasyfikatorów zgodnych

$$P\{Q_{\mathcal{S}_N}(\widehat{f}_N) = 0 \wedge Q(\widehat{f}_N) > \epsilon\}$$

$$\leq n\, P\{Q_{\mathcal{S}_N}(\widehat{f}^i) = 0 \wedge Q(\widehat{f}^i) > \epsilon\} \;\leq\; n\,(1-\epsilon)^N \;\leq\; n\exp(-\epsilon N) \;=\; \delta$$

  dla $\epsilon = \frac{1}{N}\ln\frac{n}{\delta}$

- zbyt duża złożoność n rodziny klasyfikatorów zwiększa możliwy błąd generalizacji ϵ dla zadanej liczby pomiarów

Klasyfikacja, nieskończona rodzina klasyfikatorów

- prawdopodobieństwo dużego błędu generalizacji dla klasyfikatorów zgodnych

$$p_N = P\{Q_{\mathcal{S}_N}(\widehat{f}_N) = 0 \wedge Q(\widehat{f}_N) > \epsilon\} \;\leq\; 2\,P\{Q_{\mathcal{S}_N}(\widehat{f}_N) = 0 \wedge Q_{\mathcal{S}'_N}(\widehat{f}_N) > \epsilon/2\}$$

  problem sprowadzony do skończonej liczby klasyfikacji

- *funkcja wzrostu*: liczba klasyfikacji N punktów (co najwyżej 2^N)

$$\nu_{\widehat{\mathcal{F}}}(N) = \max_{\mathbf{u}_1,\ldots,\mathbf{u}_N} \#\{\big(\widehat{f}(\mathbf{u}_1),\ldots,\widehat{f}(\mathbf{u}_N)\big) : \widehat{f} \in \widehat{\mathcal{F}}\}$$

- jeśli $\nu(N) = 2^N$, to mówimy, że zbiór $\{\mathbf{u}_1,\ldots,\mathbf{u}_N\}$ jest *rozbity* przez $\widehat{\mathcal{F}}$ i wówczas $\{(\widehat{f}(\mathbf{u}_1),\ldots,\widehat{f}(\mathbf{u}_N)) : \widehat{f} \in \widehat{\mathcal{F}}\} = \{-1,1\}^N$
- *wymiar Vapnika-Chervonenkisa* $d = \mathrm{VCdim}(\widehat{\mathcal{F}})$ rodziny $\widehat{\mathcal{F}}$: liczba elementów najliczniejszego zbioru jaki może być rozbity przez $\widehat{\mathcal{F}}$
- $\nu(N) \leq \left(\frac{\mathrm{e}\,N}{d}\right)^d$ dla $N > d$
- dla $N \geq d$, $N > 2/\epsilon$ otrzymujemy $p_N \leq 2\left(\frac{2\,\mathrm{e}\,N}{d}\right)^d 2^{-\epsilon N/2} = \delta$ zatem $\epsilon = \frac{2}{N}\left(d\,\ln\frac{2\,\mathrm{e}\,N}{d} + \ln\frac{2}{\delta}\right)$
- skończony wymiar VCdim gwarantuje możliwość uczenia w sensie PAC dla dowolnego rozkładu P

Rozbicie

- rodzina funkcji $\widehat{\mathcal{F}}$; funkcji $\widehat{f}$ odpowiada zbiór $\mathcal{U}_{\widehat{f}} = \{\mathbf{u} \in \mathbb{R}^n : \widehat{f}(\mathbf{u}) > 0\}$
  rodzina zbiorów $\mathbb{U}$
  - ▶ sieć binarna: $\mathbb{U} = \{\mathcal{U}_+(\mathbf{w}),\ \mathbf{w} \in \mathbb{R}^d\}$;
- zbiór skończony $\mathcal{B} = \{\mathbf{u}_1, \ldots, \mathbf{u}_N\}$
  elementy zbioru $\mathcal{B}$ są klasyfikowane
- $\mathcal{B}_0 \subset \mathcal{B}$ jest *wybierany* (ang. $picked\ up$) przez $\mathbb{U}$ jeśli

$$\exists\ \mathcal{U} \in \mathbb{U}: \qquad \mathcal{B}_0 = \mathcal{U} \cap \mathcal{B}$$

  - ▶ sieć binarna: dla pewnych wag $\mathcal{B}_0$ jest poprawnie klasyfikowany
- zbiór skończony jest *rozbijany* (ang. shattered) przez $\mathbb{U}$, jeśli każdy jego podzbiór jest zbierany przez $\mathbb{U}$.
  - ▶ sieć binarna: każdy podzbiór może być poprawnie sklasyfikowany

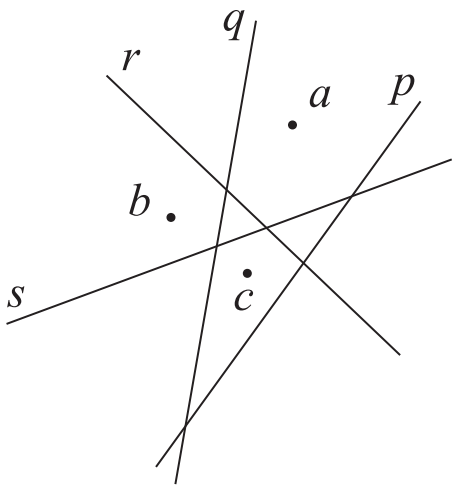
Przykład: rozbicie przez półproste w $\mathbb{R}$

- $\mathcal{I}_x = (-\infty, x), \qquad \mathbb{I} = \{\mathcal{I}_x,\ x \in \mathbb{R}\}.$
- $\{b\}$ zbierany przez $\mathbb{I}$ (przez $\mathcal{I}_{b+1} \in \mathbb{I}$)
  $\Longrightarrow$ zbiory jednoelementowe są rozbijane przez $\mathbb{I}$
- $\{b_1, b_2\}$, $b_1 < b_2$: $\quad b_2$ nie jest zbierany przez $\mathbb{I}$
  $\Longrightarrow$ zbiory dwuelementowe nie są rozbijane przez $\mathbb{I}$

Przykład: rozbicie przez półpłaszczyzny w $\mathbb{R}^2$

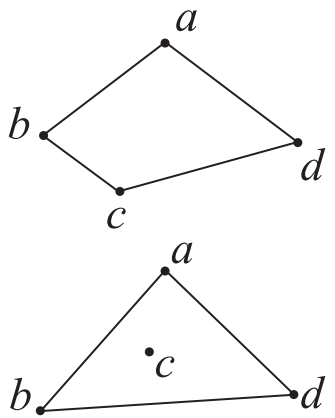
$\mathbb{S}^2$ = rodzina półpłaszczyzn w $\mathbb{R}^2$

istnieją zbiory 3-elementowe rozbijane przez $\mathbb{S}^2$



$\{a\}$ zbierany przez r^+,
$\{a, c\}$ zbierany przez q^-
...

zbiory 4-elementowe nie są rozbijane przez $\mathbb{S}^2$



Wymiar Vapnika-Chervonenkisa

- *wymiar Vapnika-Chervonenkisa* rodziny $\mathbb{U}$

  $$\mathrm{VC}(\mathbb{U}) = \text{największa liczność zbioru rozbijanego przez rodzinę } \mathbb{U}$$

  ▶ sieć binarna: największa liczność zbioru, którego wszystkie podzbiory mogą być poprawnie klasyfikowane

- $\mathrm{VC}(\mathbb{U}) = d$
  - *istnieją* d-elementowe podzbiory rozbijane przez $\mathbb{U}$

    ▶ sieć binarna: istnieje zbiór d-elementowy, którego wszystkie podzbiory mogą być poprawnie klasyfikowane
  - *mogą istnieć* d-elementowe podzbiory nie rozbijane przez $\mathbb{U}$
    (o elementach w "pozycji szczególnej")

    ▶ sieć binarna: mogą istnieć zbiory d-elementowe, których podzbiory nie dadzą się sklasyfikować poprawnie

- przykłady: $\mathrm{VC}(\mathbb{I}) = 1, \quad \mathrm{VC}(\mathbb{S}^2) = 3.$

Funkcja wzrostu

- $\nu_{\mathbb{U}}(\mathcal{B}) =$ liczba podzbiorów zbioru $\mathcal{B}$ zbieranych przez $\mathbb{U}$
- jeżeli $\mathcal{B}$ jest rozbijany przez $\mathbb{U}$ to $\nu_{\mathbb{U}}(\mathcal{B}) = 2^{|\mathcal{B}|}$
- *funkcja wzrostu*: maksymalna liczba podzbiorów zbiorów N-elementowych zbierana przez $\mathbb{U}$

$$\nu_{\mathbb{U}}(N) = \max_{\mathcal{B}:|\mathcal{B}|=N} \nu_{\mathbb{U}}(\mathcal{B})$$

$$= \begin{cases} 2^N & \text{dla } N \leq \mathrm{VC}(\mathbb{U}) \\ < 2^N & \text{dla } N > \mathrm{VC}(\mathbb{U}) \end{cases}$$

▶ klasyfikacja binarna: $\nu_{\mathbb{U}}(N) =$ maksymalna klasyfikowana poprawnie liczba podzbiorów zbiorów N-elementowych

Funkcja wzrostu — Przykłady

- $\nu_{\mathbb{I}}(1) = 2$, $\nu_{\mathbb{I}}(2) = 3$

  $$\nu_{\mathbb{I}}(N) = N + 1$$

- $\nu_{\mathcal{S}^2}(3) = 8$, $\nu_{\mathcal{S}^2}(4) = 14$

  $$\nu_{\mathcal{S}^2}(N) = \begin{cases} 2^N & \text{dla } N \leq 3 \\ N^2 - N + 2 & \text{dla } N \geq 3 \end{cases}$$

- $$\nu_{\mathcal{S}^n}(N) = \begin{cases} 2^N & \text{dla } N \leq n+1 \\ 2\sum_{i=0}^{n} \binom{N-1}{i} & \text{dla } N \geq n+1 \end{cases}$$

VC dla sieci binarnych a VC dla zbiorów

Zbiory	Sieci	Perceptron Rosenblatta
$\mathcal{U} \subset \mathbb{R}^n$ $\mathcal{U}_{\mathbf{w}} = \{\mathbf{u} \in \mathcal{U} \colon \widehat{f}_{\mathbf{w}}(\mathbf{u}) = 1\}$ $\mathbb{U} = \{\mathcal{U}_{\mathbf{w}} : \mathbf{w} \in \mathbb{R}^d\}$	zbiór wejść sieci $\mathcal{U}_{\mathbf{w}} = \{\mathbf{u} \in \mathcal{U} \colon \mathbf{N}_{\mathbf{w}}(\mathbf{u}) = 1\}$ $\mathbb{N} = \{\mathbf{N}_{\mathbf{w}}, \mathbf{w} \in \mathbb{R}^d\}$	$\mathcal{U}_{\mathbf{w},b} = \{\mathbf{u} \in \mathcal{U} : \mathbf{w}^T\mathbf{u} + b > 0\}$ $\mathbb{U} = \mathbb{S}^n$ $\mathbb{N} = \{\mathbf{N}_{\mathbf{w},b}, \mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}\}$
$\nu_{\mathbb{U}}(N)$ $\nu_{\mathbb{U}}$ – funkcja wzrostu	maksymalna liczba klasyfikacji zbioru N-elementowego za pomocą sieci $\mathbb{N}$	maksymalna liczba klasyfikacji liniowych zbioru N-elementowego $\nu_{\mathbb{S}^n}(N) = \begin{cases} 2^N & \text{dla } N \le n+1 \\ 2\sum_{i=0}^{n}\binom{N-1}{i} & \text{dla } N \ge n+1 \end{cases}$
$\mathcal{U}$ jest rozbijany przez $\mathbb{U}$	$\mathcal{U}$ jest rozbijany przez $\mathbb{N}$ sieć może dokonać dowolnej klasyfikacji elementów zbioru $\mathcal{U}$	$\mathcal{U}$ jest rozbijany przez $\mathbb{N}$ sieć może dokonać dowolnej klasyfikacji (liniowej) elementów zbioru $\mathcal{U}$
$\mathrm{VC}(\mathbb{U})$	$\mathrm{VC}(\mathbb{N})$ liczność największego zbioru dla którego $\mathbb{N}$ może dokonać dowolnej klasyfikacji	$\mathrm{VC}(\mathbb{S}^n) = \mathrm{VC}(\mathbb{N}) = n+1 = d$ liczność największego zbioru dla którego $\mathbb{N}$ może dokonać dowolnej klasyfikacji liniowej

Wymiar VC dla przykładowych sieci

typ sieci	liczba wag	VC($\mathbb{N}$)
sieć liniowa, jednowarstwowy perceptron Rosenblatta	$d = n + 1$	d
wielowarstwowy perceptron Rosenblatta	d	$O(d \ln d)$
sieć sigmoidalna	d	$O(d^2)$
$\mathrm{sign}(\sin(w\,u)$	1	∞

Ryzyko dla sieci binarnych

jeżeli $\mathrm{VC}(\mathbb{N}) = d < \infty$ to dla $N \geq d$

- π_N jest *jednostajnie zbieżny* do π
- dla każdego $\epsilon > 0$ z prawdopodobieństwem $\geq 1 - \left(\frac{2\,\mathrm{e}\,N}{d}\right)^d \exp(-\epsilon^2 N)$

$$\sup_{\mathbf{w}} |\pi_N(\mathbf{w}) - \pi(\mathbf{w})| \leq \epsilon$$

- dla "małych" $\pi(\mathbf{w})$, z prawdopodobieństwem $\geq 1 - \left(\frac{2\,\mathrm{e}\,N}{d}\right)^d \exp(-\epsilon^2 N/4)$

$$\sup_{\mathbf{w}} \frac{|\pi_N(\mathbf{w}) - \pi(\mathbf{w})|}{\sqrt{\pi(\mathbf{w})}} \leq \epsilon$$

- *ryzyko maksymalne* gwarantowane na poziomie ufności $1 - \alpha > 0$

$$\pi_N(\mathbf{w}) - \pi(\mathbf{w}) \leq \epsilon_0 = \sqrt{\frac{d}{N}\left(\ln\frac{2N}{d} + 1\right) - \frac{\ln(\alpha)}{N}}$$

- *ryzyko maksymalne* gwarantowane na poziomie ufności $1 - \alpha > 0$ dla "małych" $\pi(\mathbf{w})$

$$\pi_N(\mathbf{w}) - \pi(\mathbf{w}) \leq \epsilon_1 = 2\,\epsilon_0^2\left(1 + \sqrt{1 + \pi(\mathbf{w})/\epsilon_0^2}\right)$$

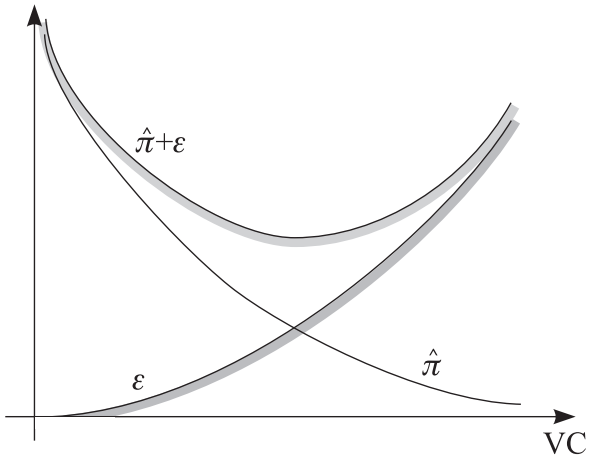
Minimalizacja ryzyka strukturalnego

- dopuszczone k błędów klasyfikacji

$$P\{Q_{\mathcal{S}_N}(\widehat{f}_N) = k/N \wedge Q(\widehat{f}_N) > \epsilon_k\} \leq \delta$$

gdzie dla $N \geq d$

$$\epsilon_k = \pi_d + \epsilon_d$$
$$= \frac{2k}{N} + \frac{4}{N}\left(d \ln \frac{2\,\mathrm{e}\,N}{d} + \ln \frac{4}{\delta}\right)$$



- dla rosnącego ciągu klasyfikatorów $\widehat{\mathcal{F}}^1 \subset \ldots \subset \widehat{\mathcal{F}}^m \subset \ldots$ gdzie $\delta^m = \delta/m$; otrzymamy

$$k^1 \geq \ldots \geq k^m \geq \ldots$$
$$d^1 \leq \ldots \leq d^m \leq \ldots$$

- *minimalizacja ryzyka strukturalnego*: wybrać d dla którego wpływ redukcji liczby błędnych klasyfikacji zbioru uczącego przeważa wpływ wzrostu złożoności rodzin klasyfikujących

12. Zjawiska dynamiczne w sieciach

Zjawiska dynamiczne w sieciach

- *układ statyczny*: wyjście w danej chwili zależy tylko od wejść w tej chwili

$$\mathbf{y}(t) = \mathbf{h}\big(\mathbf{u}(t)\big)$$

w przeciwnym przypadku układ jest *dynamiczny*

$$\mathbf{y}(t) = \mathcal{H}\big(\mathbf{u}(\tau),\, \tau \in \{0, t-1\}\big) \qquad \text{czas dyskretny}$$

$$\mathbf{y}(t) = \mathcal{H}\big(\mathbf{u}(\tau),\, \tau \in [0, t]\big) \qquad \text{czas ciągły}$$

- *sieć* jest *statyczna* / *dynamiczna* jeśli dla ustalonych wag jest układem statycznym / dynamicznym
- sprzężenia zwrotne sieci warstwowych
  - neurony dynamiczne (lokalne sprzężenia zwrotne wewnątrz neuronów)
  - neurony statyczne, sprzężenia zwrotne między neuronami
  - warstwy statyczne, sprzężenia zwrotne między warstwami
  - sieć statyczna, zewnętrzne sprzężenia zwrotne

- równania stanu sieci dynamicznej – czas ciągły

$$\mathbf{y}(t) = \mathbf{h}\big(\mathbf{u}(t), \mathbf{x}(t); \mathbf{w}\big)$$
$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{u}(t), \mathbf{x}(t); \mathbf{w}\big)$$

- równania stanu sieci dynamicznej – czas dyskretny

$$\mathbf{y}(t) = \mathbf{h}\big(\mathbf{u}(t), \mathbf{x}(t); \mathbf{w}\big)$$
$$\mathbf{x}(t) = \mathbf{f}\big(\mathbf{u}(t-1), \mathbf{x}(t-1); \mathbf{w}\big)$$

  - *dynamika synchroniczna*: aktualizacja stanu wszystkich neuronów równocześnie

$$y_i(t) = g(\mathbf{w}_i^T \mathbf{u}(t-1) + b_i), \qquad i = 1, \ldots, m$$

  - *dynamika asynchroniczna*: aktualizacja stanu tylko jednego neuronu w jednej chwili

$$y_i(t) = \begin{cases} g(\mathbf{w}_i^T \mathbf{u}(t) + b_i) & \text{jeżeli } i = \iota(t) \\ y_i(t-1) & \text{jeżeli } i \neq \iota(t) \end{cases}$$

    mechanizm wyboru $\iota(t) \sim \mathbf{U}_{\{1,\ldots,m\}}$, niezależne

Zastosowania sieci dynamicznych

- zagadnienia czasu skończonego
  - modelowanie nieliniowych układów dynamicznych
  - sterowanie adaptacyjne
- zagadnienia asymptotyczne
  - pamięci skojarzeniowe (asocjacyjne)
  - optymalizacja globalna

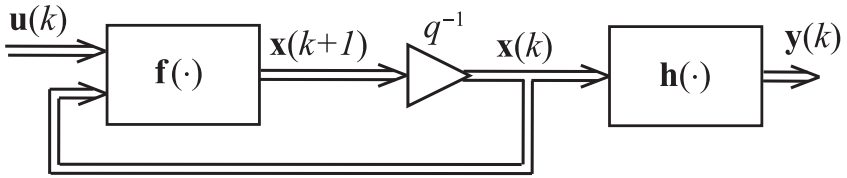
Struktura układu nieliniowego

- nieliniowy niezmienniczy układ deterministyczny $\mathcal{S} = (\mathbf{f}, \mathbf{h})$

$$\mathbf{x}(k+1) = \mathbf{f}\big(\mathbf{x}(k), \mathbf{u}(k)\big)$$
$$\mathbf{y}(k) = \mathbf{h}\big(\mathbf{x}(k)\big)$$

gdzie $\mathbf{f}(\mathbf{0}, \mathbf{0}) = \mathbf{0}, \quad \mathbf{h}(\mathbf{0}) = \mathbf{0}, \quad \mathbf{u}(k), \mathbf{y}(k) \in \mathbb{R}^m, \quad \mathbf{x}(k) \in \mathbb{R}^n.$

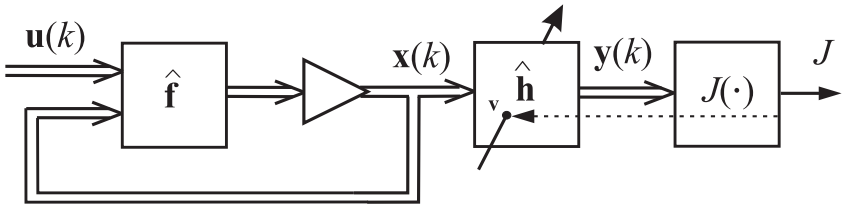
- koszt $Q = \frac{1}{2} \sum_{t=1}^{N} \|\mathbf{y}(t) - \mathbf{y}^*(t)\|^2$



- nieliniowości reprezentowane przez sieci

$$\mathbf{x}(t+1) = \mathbf{N}^{\mathbf{f}}\big(\mathbf{x}(t), \mathbf{u}(t);\, \mathbf{w}\big)$$
$$\mathbf{y}(t) = \mathbf{N}^{\mathbf{h}}\big(\mathbf{x}(t);\, \mathbf{v}\big)$$

Gradienty wag dla sieci wyjścia



- v wpływa na wszystkie współrzędne wyjścia w każdej chwili t

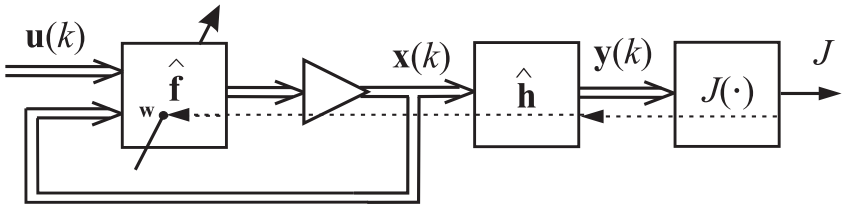
$$\frac{dQ}{dv} = \sum_{t=1}^{N} \sum_{j=1}^{p} \frac{dQ}{dy_j(t)} \frac{\partial \mathbf{N}_j^h(t)}{\partial v}$$

- każda współrzędna wyjścia $y(t)$ wpływa bezpośrednio na koszt

$$\frac{dQ}{dy_i(t)} = y_i(t) - y_i^*(t)$$

- ostatecznie $\dfrac{dQ}{dv} = \sum_{t=1}^{N} \sum_{j=1}^{n} \big(y_i(t) - y_i^*(t)\big) \dfrac{\partial \mathbf{N}_j^h(t)}{\partial v}$
- pochodne $\dfrac{\partial \mathbf{N}_j^h(t)}{\partial v}$ mogą być też obliczone przy użyciu BP

Gradienty wag dla sieci stanu



- w wpływa na wszystkie współrzędne stanu w każdej chwili t

$$\frac{dQ}{dw} = \sum_{t=1}^{N} \sum_{j=1}^{n} \frac{dQ}{dx_j(t)} \frac{\partial \mathbf{N}_j^f(t)}{\partial w}$$

- $x_i(t)$ wpływa na $\mathbf{y}(t)$ i (z wyjątkiem $t = N$) na $\mathbf{x}(t+1)$

$$\frac{dQ}{dx_i(t)} = \sum_{j=1}^{p} \frac{dQ}{dy_j(t)} \frac{\partial \mathbf{N}_j^h(t)}{\partial x_i(t)} + \sum_{j=1}^{n} \frac{dQ}{dx_j(t+1)} \frac{\partial \mathbf{N}_j^f(t)}{\partial x_i(t)} \quad \text{dla } t < N$$

$$\frac{dQ}{dx_i(N)} = \sum_{j=1}^{p} \frac{dQ}{dy_j(N)} \frac{\partial \mathbf{N}_j^h(N)}{\partial x_i(N)}$$

- $y(t)$ wpływa bezpośrednio na koszt chwilowy

$$\frac{dQ}{dy_i(t)} = y_i(t) - y_i^*(t)$$

Gradienty wag dla układu dynamicznego sieci

- wektory wrażliwości

$$Q_{\mathbf{x}}(t) = \left[\frac{dQ}{dx_1(t)} \quad \cdots \quad \frac{dQ}{dx_n(t)} \right]^T$$

- liniowe równania rekurencyjne w odwróconym czasie

$$Q_{\mathbf{x}}(t) = \mathbf{N}^{\mathbf{f}}_{\mathbf{x}}(t)^T\, Q_{\mathbf{x}}(t{+}1) + \mathbf{N}^{\mathbf{h}}_{\mathbf{x}}(t)^T \big(\mathbf{y}(t) - \mathbf{y}^*(t)\big), \qquad Q_{\mathbf{x}}(N{+}1) = \mathbf{0}$$

$$Q_{\mathbf{w}}(t) = Q_{\mathbf{w}}(t{+}1) + \mathbf{N}^{\mathbf{f}}_{\mathbf{w}}(t)^T Q_{\mathbf{x}}(t), \qquad Q_{\mathbf{w}}(N{+}1) = \mathbf{0}$$

$$Q_{\mathbf{v}}(t) = Q_{\mathbf{v}}(t{+}1) + \mathbf{N}^{\mathbf{h}}_{\mathbf{v}}(t)^T \big(\mathbf{y}(t) - \mathbf{y}^*(t)\big), \qquad Q_{\mathbf{v}}(N{+}1) = \mathbf{0}$$

gdzie $\dfrac{dQ}{dw}$ i $\dfrac{dQ}{dv}$ są elementami $Q_{\mathbf{w}}(1)$, $Q_{\mathbf{v}}(1)$

Porównanie: propagacja prosta

- Q zależy bezpośrednio od współrzędnych wyjścia we wszystkich chwilach

$$\frac{dQ}{dw} = \sum_{t=1}^{N} \sum_{j=1}^{n} \frac{\partial Q}{\partial y_j(t)} \frac{dy_j(t)}{dw}$$

- każde wyjście zależy od wszystkich współrzędnych stanu w tej samej chwili

$$\frac{dy_i(t)}{dw} = \sum_{j=1}^{p} \frac{\partial \widehat{h}_i(t)}{\partial x_j(t)} \frac{dx_j(t)}{dw}$$

- stan zależy od wag bezpośrednio i poprzez wszystkie poprzednie wartości stanu

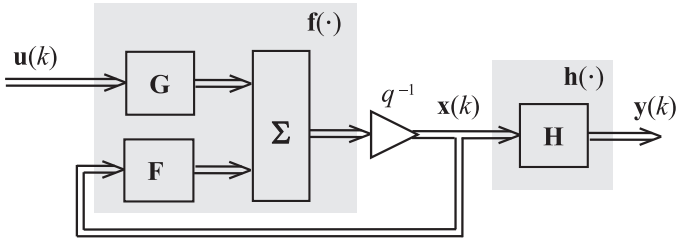
$$\frac{dx_i(t)}{dw} = \frac{\partial \widehat{f}_i(t)}{\partial w} + \sum_{j=1}^{n} \frac{\partial \widehat{f}_i(t)}{\partial x_j(t-1)} \frac{dx_j(t-1)}{dw}$$

13. Modele NARX układów nieliniowych

Obserwacja stanu układu liniowego

- układ liniowy

$$\mathbf{x}(t+1) = \mathbf{F}\,\mathbf{x}(t) + \mathbf{G}\,\mathbf{u}(t)$$
$$\mathbf{y}(t) = \mathbf{H}\,\mathbf{x}(t)$$



- *dla układów liniowych*

$$\begin{aligned}
\mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k)\\
\mathbf{y}(k+1) &= \mathbf{C}\mathbf{A}\mathbf{x}(k) + \mathbf{C}\mathbf{B}\mathbf{u}(k)\\
&\cdots\\
\mathbf{y}(k+n-1) &= \mathbf{C}\mathbf{A}^{n-1}\,\mathbf{x}(k) + \sum_{j=1}^{n-1} \mathbf{C}\mathbf{A}^{j-1}\mathbf{B}\,\mathbf{u}(k+n-1-j)
\end{aligned}$$

- linia opóźniająca TDL (*ang. tapped delay line*) $\mathbf{q}^{-n}\mathbf{y}(t) = \begin{bmatrix} \mathbf{y}(t) \\ \cdots \\ \mathbf{y}(t-n+1) \end{bmatrix}$,

- wyjście zależy od "bieżącego" stanu i "poprzednich" wejść

$$\begin{aligned}
\mathbf{y}(k+n) &= \mathbf{C}\mathbf{A}^{n}\,\mathbf{x}(k) + \sum_{j=1}^{n} \mathbf{C}\mathbf{A}^{j-1}\mathbf{B}, \mathbf{u}(k+n-j)\\
&= \mathrm{Lin}\big(\mathbf{x}(k), \mathbf{q}^{-n}\mathbf{u}(k+n)\big) \qquad \Longleftarrow \textbf{jak wyznaczyć } \mathbf{x}(k)
\end{aligned}$$

- układ jest *obserwowalny* jeżeli stan początkowy można określić na podstawie skończonej liczby przyszłych wartości wyjść
- dla układów liniowych: układ jest obserwowalny $\Longleftrightarrow$

  macierz obserwowalności $\mathbf{W}_o = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \cdots \\ \mathbf{CA}^{n-1} \end{bmatrix} \in \mathbb{R}^{nm \times n}$ ma rząd równy liczbie kolumn (n)
- dla układów liniowych obserwowalnych

$$\begin{aligned} \mathbf{x}(k) &= \mathsf{Lin}\big(\mathbf{q}^{-n}\mathbf{y}(k+n),\, \mathbf{q}^{-n+1}\mathbf{u}(k+n-1)\big) \\ \mathbf{y}(k+n) &= \mathsf{Lin}\big(\mathbf{q}^{-n}\mathbf{y}(k+n),\, \mathbf{q}^{-n}\mathbf{u}(k+n)\big) \end{aligned}$$

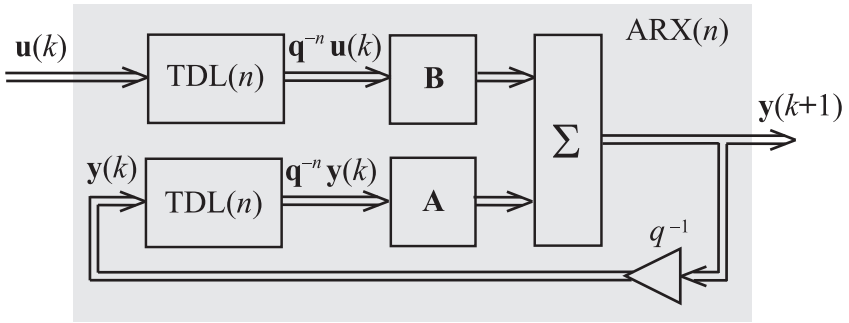
Reprezentacja ARX układu liniowego

- jeśli układ jest obserwowalny, to stan jest liniowo zależny od *co najwyżej* n opóźnionych wejść i wyjść - *reprezentacja ARX(n)*

$$\mathbf{y}(k) = \sum_{i=1}^{n} \mathbf{A}_i\, \mathbf{y}(k-i) + \sum_{i=1}^{n} \mathbf{B}_i\, \mathbf{u}(k-i)$$

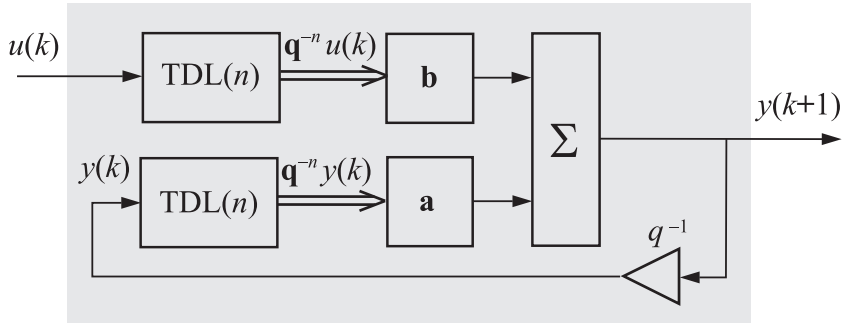
czyli, oznaczając $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \dots & \mathbf{A}_n \end{bmatrix}$

$$\mathbf{y}(t) = \mathbf{A}\, \mathbf{q}^{-n} \mathbf{y}(t-1) + \mathbf{B}\, \mathbf{q}^{-n} \mathbf{u}(t-1)$$



- dla układów SISO

$$y(t+1) = \mathbf{a}^T \mathbf{q}^{-n} y(t) + \mathbf{b}^T \mathbf{q}^{-n} u(t)$$



Obserwacja stanu układu nieliniowego

- dla układu $\mathcal{S} = (\mathbf{f}, \mathbf{h})$

$$\begin{aligned}
\mathbf{y}(k) &= \mathbf{h}\big(\mathbf{x}(k)\big) \\
\mathbf{y}(k+1) &= \mathbf{h}\big(\mathbf{f}(\mathbf{x}(k),\, \mathbf{u}(k))\big) = \boldsymbol{\psi}_1\big(\mathbf{x}(k),\, \mathbf{u}(k)\big) \\
\mathbf{y}(k+2) &= \mathbf{h}\big(\mathbf{f}(\mathbf{x}(k+1),\, \mathbf{u}(k+1))\big) = \boldsymbol{\psi}_2\big(\mathbf{x}(k),\, \mathbf{u}(k),\, \mathbf{u}(k+1)\big) \\
&\cdots \\
\mathbf{y}(k+n-1) &= \boldsymbol{\psi}_{n-1}\big(\mathbf{x}(k),\, \mathbf{q}^{-n+1}\mathbf{u}(k+n-1)\big)
\end{aligned}$$

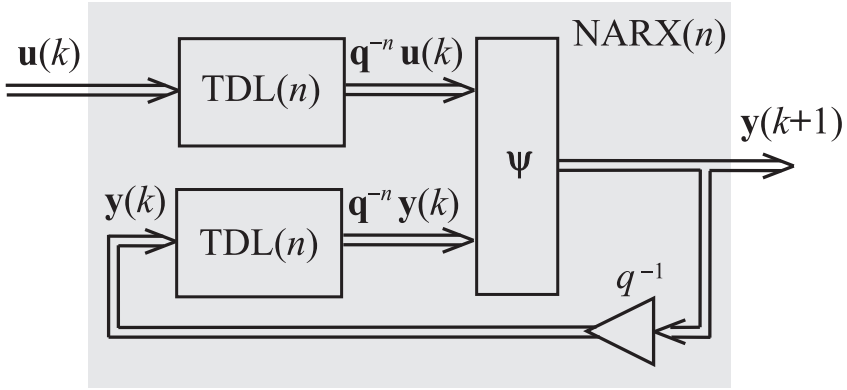
- $\mathbf{y}(k+n) = \boldsymbol{\psi}_n\big(\mathbf{x}(k),\, \mathbf{q}^{-n}\mathbf{u}(k+n)\big)$
- *jeżeli $\mathcal{S}_L$ jest obserwowalny* ($\mathcal{S}$ jest *lokalnie obserwowalny*) to

$$\begin{aligned}
\mathbf{x}(k) &= \boldsymbol{\phi}\big(\mathbf{q}^{-n}\mathbf{y}(k+n), \mathbf{q}^{-n+1}\mathbf{u}(k+n-1)\big) \\
\mathbf{y}(k+n) &= \boldsymbol{\psi}\big(\mathbf{q}^{-n}\mathbf{y}(k+n),\, \mathbf{q}^{-n}\mathbf{u}(k+n)\big)
\end{aligned}$$

Struktura NARX układu nieliniowego

- jeśli obiekt zlinearyzowany jest obserwowalny to **lokalnie** w pobliżu stanu równowagi

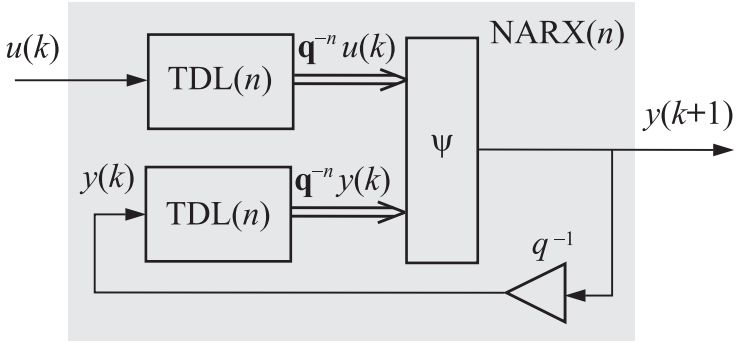
$$\mathbf{y}(k) = \psi\big(\mathbf{q}^{-n}\mathbf{y}(k),\, \mathbf{q}^{-n}\mathbf{u}(k)\big)$$

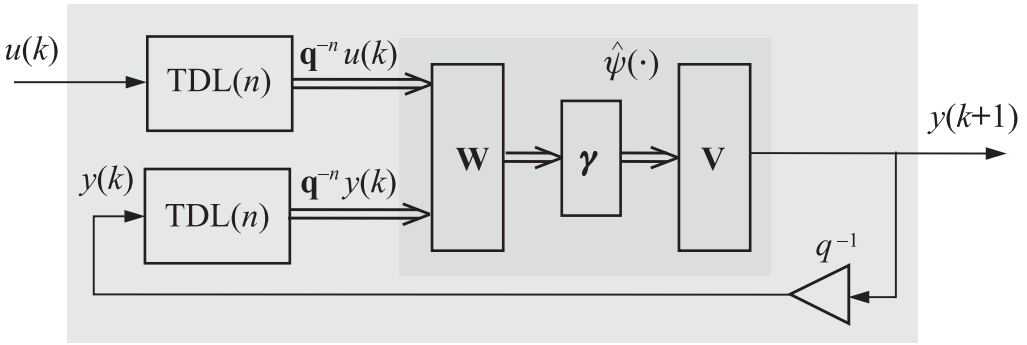


Model NARX z aproksymacją neuronową

- funkcje nieliniowe można aproksymować przez sieci neuronowe

$$\mathbf{y}(t+1) = \widehat{\psi}\big(\mathbf{q}^{-n}\mathbf{y}(t),\, \mathbf{q}^{-n}\mathbf{u}(t);\, \mathbf{w}\big)$$





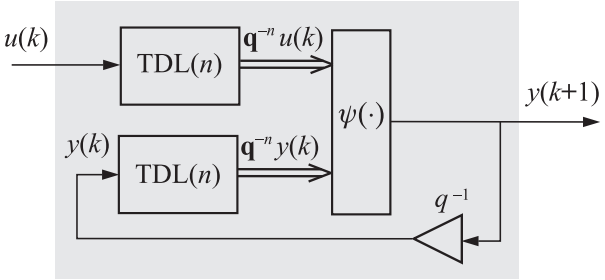
Globalna reprezentacja wejście-wyjście

- globalna reprezentacja jest nieznana
- jeśli układ autonomiczny jest obserwowalny

  $2n+1$ poprzednich wartości wyjścia wystarcza do określenia wyjscia
- jeśli układ jest odwracalny względem $\mathbf{x}(k)$

  $2n+1$ poprzednich wartości wejść i wyjść wystarcza do określenia wyjścia
- n is zwykle nieznane: **stosować NARX(m) dla odpowiednio dużych m**

Gradienty wag dla sieci NARX



- sieć NARX z jednym wejściem i jednym wyjściem

$$y(t+1) = \mathbf{N}\big(y(t), \ldots, y(t-n), u(t), \ldots, u(t-n);\, \mathbf{w}\big)$$

- koszt $Q = \sum_{t=1}^{N} q(y(t))$

- w wpływa na wyjście sieci w każdej chwili t

$$\frac{dQ}{dw} = \sum_{t=1}^{N} \frac{dQ}{dy(t)} \frac{\partial \mathbf{N}(t)}{\partial w}$$

- $y(t)$ wpływa na $y(t+1), \ldots, y(t+n)$ (poprzez TDL i sieć) i koszt

$$\frac{dQ}{dy(t)} = \sum_{s=t+1}^{t+n} \frac{dQ}{dy(s)} \frac{\partial \mathbf{N}(s)}{\partial y(t)} + \frac{dq}{dy(t)}$$

- obliczenia $\dfrac{dQ}{dy(t)}$ *wstecz w czasie* dla $t = N, \ldots, 1$, z warunkami $\dfrac{dQ}{dy(s)} = 0$ dla $s = N+1, \ldots, N+n$

- obliczenie $\dfrac{\partial \widehat{\psi}(t)}{\partial w}$: propagacja zwrotna przez sieć

14. Optymalizacja globalna i pamięci asocjacyjne

Bibliografia

[0-1] S. Haykin, *Neural Networks. A Comprehensive Foundation. Second Edition*, Prentice Hall, Inc., Upper Saddle River, New Jersey, USA, 1999

[0-2] J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. of the National Acad. of Sciences of the USA*, vol. **79**, 2554–2558, 1982

[0-3] J.J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. of the National Acad. of Sciences of the U.S.A.*, vol. **81**, 3088-3092, 1984

Stabilność próbkowanych układów dynamicznych

- układ dynamiczny

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$$

- *punkt równowagi* $\mathbf{x}_e$

$$\mathbf{x}_e = f(\mathbf{x}_e, \mathbf{0}, t) \qquad \forall t \geq t_0$$

- *stabilny punkt równowagi*: dla każdego $\epsilon > 0$, t_0, istnieje $\delta_{t_0} > 0$

$$\|\mathbf{x}(t_0) - \mathbf{x}_e\| < \delta_{t_0} \quad \Longrightarrow \quad \forall t \geq t_0 \quad \|\mathbf{x}(t) - \mathbf{x}_e\| < \epsilon$$

- *stabilność asymptotyczna*: dodatkowo, dla każdego t_0 istnieje $\gamma > 0$

$$\|\mathbf{x}(t_0) - \mathbf{x}_e\| < \gamma \quad \Longrightarrow \quad \|\mathbf{x}(t) - \mathbf{x}_e\| \underset{t\to\infty}{\longrightarrow} 0$$

- *obszar przyciągania*: kula $B(\mathbf{x}_e, \gamma) = \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_e\| < \gamma\}$ taka że jeśli $\mathbf{x}(t) \in B$ to $\lim \mathbf{x}(t) = \mathbf{x}_e$; istnieje dla stabilności asymptotycznej
- *stabilność globalna*: dowolnie duże γ
- *stabilność jednostajna*: δ nie zależy of t_0

Druga metoda Lapunowa

- układ dynamiczny

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{x}(t))$$

- *punkt równowagi* $\mathbf{x}_e = \mathbf{f}(\mathbf{x}_e)$
- *funkcja Lapunowa*: funkcja stanu, $V \in \mathcal{C}_1$: $\mathbb{R}^m \mapsto \mathbb{R}$ spełniająca warunki

$$V(\mathbf{x}_e) = 0$$

$$V(\mathbf{x}) > 0 \qquad \forall \mathbf{x} \in B(\mathbf{x}_e, r)$$

$$\Delta V(\mathbf{x}) = V(\mathbf{f}(\mathbf{x})) - V(\mathbf{x}) \leq 0 \qquad \forall \mathbf{x} \in B(\mathbf{x}_e, r)$$

- TW. LAPUNOWA: punkt równowagi $\mathbf{x}_e$ jest *stabilny* jeśli istnieje dla niego funkcja Lapunowa
- punkt równowagi jest *asymptotycznie stabilny* jeśli istnieje funkcja Lapunowa i $\Delta V(\mathbf{x}) \not\equiv 0$ wzdłuż **każdej trajektorii** w pewnej $B(\mathbf{x}_e, r_1)$.

Zastosowania własności asymptotycznych sieci dynamicznych

- *pamięć adresowana zawartością*; wyszukiwanie na podstawie informacji niepełnej
- wzorce pamiętane a stany stabilne
- obszary przyciągania stanów stabilnych: prawidłowe wyszukiwanie
- *sieć jako pamięć*: wzorce odpowiadają stanom stabilnym sieci
- *sieć jako algorytm minimalizacji*: minimalizacja odpowiada poszukiwaniu stanów stabilnych sieci

Sieć Hopfielda

- dyskretna sieć Hopfielda: 1 warstwa, sign, asynchroniczna, $\mathbf{u}(t) = \mathbf{y}(t-1)$

$$y_i(t) = \begin{cases} \mathrm{sign}(\mathbf{w}_i^T \mathbf{y}(t-1) + b_i) & \text{jeżeli } i = \iota(t) \\ y_i(t-1) & \text{w przeciwnym przypadku} \end{cases}$$

zmiany wyjść: możliwe przemieszczenie do sąsiedniego wierzchołka kostki w $\mathbb{R}^m$.

Stabilność sieci Hopfielda

- punkty równowagi dla dyskretnej sieci Hopfielda

$$y_i = \text{sign}(z_i) = \text{sign}(\mathbf{w}_i^T \mathbf{y} + b_i), \qquad i = 1, \ldots, m$$

  postać macierzowa

$$\mathbf{y} = \text{sign}(\mathbf{z}) = \text{sign}(\mathbf{W}\,\mathbf{y} + \mathbf{b}), \quad \mathbf{y} \in \{-1, 1\}^m$$

- stany stabilne odpowiadają lokalnym minimom funkcji Lapunowa w $\{-1, 1\}^m$
- minima w $\mathbb{R}^m$ są na ogół inne niż w $\{-1, 1\}^m$, obszary przyciągania w $\mathbb{R}^m$ nie muszą zawierać punktów bliskich w metryce Hamminga

Funkcja Lapunowa dla sieci Hopfielda

- TWIERDZENIE jeżeli macierz $\mathbf{W}$ jest **symetryczna i ma nieujemną główną przekątną**

  $$\mathbf{W} = \mathbf{W}^T \qquad \text{i} \qquad w_{i,i} \geq 0 \quad \text{dla każdego } i$$

  to dyskretna *asynchroniczna sieć Hopfielda jest stabilna asymptotycznie*

- kandydat na funkcję Lapunowa $V(\mathbf{y}) = -\frac{1}{2}\,\mathbf{y}^T\mathbf{W}\,\mathbf{y} - \mathbf{y}^T\mathbf{b} + c$, odpowiednio duże c

  $$\Delta V(\mathbf{y}) = V(\mathbf{y}^+) - V(\mathbf{y}) \qquad \mathbf{y}^+ = \mathbf{N}(\mathbf{y}) = \mathbf{y} + \Delta\mathbf{y}$$

  $$= -\frac{1}{2}\,\Delta\mathbf{y}^T\mathbf{W}\,\Delta\mathbf{y} - \frac{1}{2}\,\Delta\mathbf{y}^T\mathbf{W}\,\mathbf{y} - \frac{1}{2}\,\mathbf{y}^T\mathbf{W}\,\Delta\mathbf{y} - \Delta\mathbf{y}^T\mathbf{b}$$

  $$= -\frac{1}{2}\,\Delta\mathbf{y}^T\mathbf{W}\,\Delta\mathbf{y} - \Delta\mathbf{y}^T\mathbf{z}^+ \qquad \text{symetria } \mathbf{W}$$

  $$= -\frac{1}{2}\,(y_\iota^+ - y_\iota)^2\,\underbrace{w_{\iota,\iota}}_{\geq 0} - \underbrace{(y_\iota^+ - y_\iota)\,z_\iota^+}_{\geq 0} \qquad \text{asynchronizm}$$

  $\Longrightarrow V$ jest funkcją Lapunowa

- jeśli $y_\iota^+ - y_\iota = 0$ to
  - wszystkie współrzędne pozostają niezmienione – punkt równowagi
  - albo $y_\iota^+ - y_\iota \neq 0$ dla pewnego przyszłego ι czyli V nie jest $\equiv 0$

  $\Longrightarrow$ stabilność asymptotyczna

Synchroniczna sieć Hopfielda

- kandydat na funkcję Lapunowa $V(\mathbf{y}) = -\mathbf{y}^{+T}\mathbf{W}\,\mathbf{y} - (\mathbf{y}^{+} - \mathbf{y})^T\mathbf{b} + c$

$$\begin{aligned}\Delta V(\mathbf{y}) &= V(\mathbf{y}^{+}) - V(\mathbf{y}) \\ &= -(\mathbf{y}^{++} - \mathbf{y})^T\mathbf{W}\,\mathbf{y}^{+} - (\mathbf{y}^{++} - \mathbf{y})^T\mathbf{b} \qquad \mathbf{y}^{++} = \mathbf{N}(\mathbf{y}^{+}), \quad \text{symetria } \mathbf{W} \\ &= -(\mathbf{y}^{++} - \mathbf{y})^T\mathbf{z}^{++} \\ &= -\sum_i \underbrace{(y_i^{++} - y_i)\, z_i^{++}}_{\geq 0} \leq 0\end{aligned}$$

  $\Longrightarrow$ funkcja Lapunowa, stabilność sieci synchronicznej

- możliwa okresowość, okres 2

Przykład: Problem podróżującego akwizytora

- *problem NP-zupełny* (NP: *niedeterministyczny wielomianowy*): hipotetycznie, złożoność rośnie szybciej niż liniowo z rozmiarem; jeśli istnieje algorytm rozwiązania w czasie wielomianowym, wszystkie są rozwiązywalne w czasie wielomianowym
- *problem podróżującego akwizytora* (TSP): znaleźć najkrótszą trasę łączącą m zadanych punktów (problem NP-zupełny)
- macierz odległości $d_{c,c'} = |x_c - x_{c'}|, \quad c, c' = 1, \ldots, m$
- macierz trasy $y_{c,p} = \begin{cases} 1 & \text{jeżeli miasto } c \text{ odwiedzono na } p\text{-tym etapie} \\ 0 & \text{w przeciwnym przypadku} \end{cases}$

- długość trasy (macierz Y kołowa)

$$Q_0 = \sum_c \sum_{c'} \sum_p \sum_{p'} d_{c,c'}\, y_{c,p}\, y_{c',p'} \left(1 - \delta_{c,c'}\right) \left(\delta_{p',p-1} + \delta_{p',p+1}\right)$$

- kara za podwójną wizytę

$$Q_1 = \sum_c \sum_p \sum_{p' \neq p} y_{c,p}\, y_{c,p'} = \sum_c \sum_{c'} \sum_p \sum_{p'} y_{c,p} y_{c',p'} \delta_{c,c'} \left(1 - \delta_{p,p'}\right)$$

- kara za wizytę dwu miast w tym samym etapie

$$Q_2 = \sum_p \sum_c \sum_{c' \neq c} y_{c,p}\, y_{c',p} = \sum_p \sum_{p'} \sum_c \sum_{c'} y_{c,p} y_{c',p'} \delta_{p,p'} \left(1 - \delta_{c,c'}\right)$$

- kara za nie wykonanie zadania $Q_3 = \left(\sum_c \sum_p y_{c,p} - m\right)^2$
- funkcja Lagrange'a $L = Q_0 + \beta_1 Q_1 + \beta_2 Q_2 + \beta_3 Q_3, \quad \beta_i > 0$
- porównanie z funkcją Lapunowa daje obciążenia i wagi:

$$b_{(c,p)} = -2\,\beta_3\, m$$

$$w_{(c,p)(c',p')} = -2\, d_{c,c'} \left(1 - \delta_{c,c'}\right) \left(\delta_{p',p-1} + \delta_{p',p+1}\right) + \\ -2\,\beta_1\, \delta_{c,c'} \left(1 - \delta_{p,p'}\right) - 2\,\beta_2\, \delta_{p,p'} \left(1 - \delta_{c,c'}\right) - 2\,\beta_3\, \delta_{c,c'}\, \delta_{p,p'}$$

Przykład: Podział obwodu na chipy

- podział m elementów na 2 chipy z ograniczeniem połączeń między płytkami
- macierz połączeń: $s_{i,j}$ = liczba połączeń między elementami i, j
- rozmieszczenie elementów: $y_i = \begin{cases} 1 & i\text{-ty element na chipie } 1 \\ -1 & i\text{-ty element na chipie } 2 \end{cases}$
- liczba połączeń między chipami:
  $Q_0 = \frac{1}{8}\sum_i \sum_j s_{i,j}\,(y_i - y_j)^2 = c - \frac{1}{4}\sum_i \sum_j s_{i,j}\,y_i\,y_j$
- równomierność rozmieszczenia: $Q_1 = (\sum_i y_i)^2 = \sum_i \sum_j y_i\,y_j$
- koszt $Q = Q_0 + \beta_1\,Q_1, \quad \beta_1 > 0$
- porównanie z funkcją Lapunowa daje obciążenia i wagi:
  $w_{i,j} = s_{i,j}/2 - 2\beta_1, \qquad b_i = 0$

Pamięć asocjacyjna – wzorce ortogonalne

- wzorce ortogonalne $\mathbf{y}^{(k)} \in \{-1,1\}^m$, $k = 1,\ldots,N \le m$, $\quad \mathbb{Y} = \{\mathbf{y}^{(k)},\ 1 \le k \le N\}$
- $\mathbf{y}^{(i)} \in \{-1,1\}^m$ ortogonalne, $\overline{\mathbb{Y}} = \{\mathbf{y}^{(k)},\ N+1 \le k \le m\} \perp \mathbb{Y}$
- $(\mathbb{Y}, \overline{\mathbb{Y}})$ jest bazą w $\mathbb{R}^m$; rozbicie dowolnego $\mathbf{y} \in \{-1,1\}$ względem bazy $(\mathbb{Y}, \overline{\mathbb{Y}})$

$$\mathbf{y} = \sum_{k=1}^{m} \alpha_k \, \mathbf{y}^{(k)}, \qquad \alpha_k = \frac{\mathbf{y}^T \mathbf{y}^{(k)}}{\|\mathbf{y}^{(k)}\|^2} = \frac{1}{m} \mathbf{y}^T \mathbf{y}^{(k)}, \quad -1 \le \alpha_k \le 1, \quad 1 \le k \le m$$

- sieć Hopfielda: $\mathbf{b} = 0$, $\mathbf{W} = \frac{1}{m} \sum_{k=1}^{N} \mathbf{y}^{(k)} \mathbf{y}^{(k)\,T} - N\,\mathbf{I}$ *(Reguła Hebba)*
  $\mathbf{W} = \mathbf{W}^T$, $w_{i,i} = 0$
- funkcja Lapunowa dla sieci ($\alpha_k = \alpha_k(\mathbf{y})$)

$$V(\mathbf{y}) = -\frac{1}{2}\,\mathbf{y}^T \mathbf{W}\,\mathbf{y} = -\frac{1}{2m} \sum_{i=1}^{m} \alpha_i \, \mathbf{y}^{(i)\,T} \sum_{k=1}^{N} \mathbf{y}^{(k)} \, \mathbf{y}^{(k)\,T} \sum_{j=1}^{m} \alpha_j \, \mathbf{y}^{(j)} + \frac{N}{2} \, \|\mathbf{y}\|^2$$

$$= -\frac{1}{2m} \sum_{k=1}^{N} |\alpha_k|^2 \, \|\mathbf{y}^{(k)}\|^4 + \frac{Nm}{2} = \frac{-m}{2} \sum_{k=1}^{N} |\alpha_k|^2 + \frac{Nm}{2}$$

- V minimalizowana dla $\alpha_k = \pm 1$ (kombinacje wzorców) i wówczas $V(\mathbf{y}) = 0$ (*stany mieszane*)
- *stany mieszane stabilne*: wzorce przeciwne, kombinacje nieparzystej liczby wzorców

Pamięć asocjacyjna - wzorce losowe

- wzorce losowe $\mathbf{y}^{(k)} \in \{-1,1\}^m, \quad k=1,\ldots,N \le m,$ reguła Hebba
- dla wzorca $\mathbf{y}^{(\alpha)}$

$$\mathbf{z}^{(\alpha)} = \mathbf{W}\,\mathbf{y}^{(\alpha)} = \mathbf{y}^{(\alpha)} + \frac{1}{m}\sum_{k\neq\alpha}\mathbf{y}^{(k)}\,\mathbf{y}^{(k)\,T}\mathbf{y}^{(\alpha)}$$

$$z_i^{(\alpha)} = y_i^{(\alpha)} + \underbrace{\frac{1}{m}\sum_{k\neq\alpha} y_i^{(k)} \sum_j \mathbf{y}_j^{(k)\,T}\mathbf{y}_j^{(\alpha)}}_{\text{przesłuch } v_i}$$

- rozkład dwumianowy $\mathbf{B}^{\pm}_{n,p}$: suma n niezależnych zmiennych $\{-1,1\}$ gdzie $P\{1\}=p$
  $\mathcal{E}\mathbf{B}^{\pm}_{n,p} = 0, \quad \mathcal{V}\mathbf{B}^{\pm}_{n,p} = 4np\,(1-p)$
- *przesłuch*: $mv_i \sim \mathbf{B}^{\pm}_{N^2-N,1/2}, \quad \mathcal{E}mv_i = 0, \quad \mathcal{V}mv_i = N^2-N, \qquad \xi = \frac{m}{N}v_i \rightsquigarrow \mathbf{N}_{0,1}$
- zmiana jednego bitu: $q = P\{v_i < -1 \,|\, y_i^{(\alpha)} = 1\} = P\{v_i > 1 \,|\, y_i^{(\alpha)} = -1\} = P\{\xi > \frac{m}{N}\}$
  $= 1 - \Phi\big(\frac{m}{N}\big)$
- powyżej $N_0 = 0.138\,m$ zmiana jednego bitu pociąga *lawinowe zmiany stanu*
- wzorzec odtwarzany bezbłędnie: $1-(1-q)^m \sim q\,m \to 0$ czyli $N_{\max} \sim \frac{m}{2\ln m}$
- wszystkie wzorce odtwarzane bezbłędnie : $1-(1-q)^{mN} \sim q\,mN \to 0$ czyli $N_{\max} \sim \frac{m}{4\ln m}$