Curs 2

În cele ce urmează vom considera un alt model major de rețea de neuroni în care activarea unui perceptron oarecare este determinată de distanța dintre vectorul de intrare și un vector prototip.

O proprietate interesantă și importantă a acestor tipuri de rețele numite rețele de funcții radiale, denumite în engleză RBF, este aceea că unifică un număr de concepte importante cum ar fi: aproximarea funcțiilor, regularizare, interpolare (în engleză - noisy interpolation -), estimarea densităților de probabilitate, Teoria clasării optimale și funcții de potențial - potential function.

O consecință a acestui punct de vedere unificator este și faptul că procedeul de antrenare substanțial mai rapid decât cele utilizate pentru rețelele RTM. Această rezultă din interpretarea care poate fi dată stării reprezentării formată din stratul funcțional oarecare și care conduce la o procedură de învățare în doi pași. În primul pas parametrii care guvernează funcțiile radiale (corespunzătoare perceptronilor de pe stratul oarecare) sunt determinați relativ rapid folosind metode nesupervizate (adică metode în care sunt utilizate doar datele de intrare nu și valorile țintă corespunzătoare). Ne dorim pas ca în învățarea prezentă determinarea ponderilor de pe ultimul strat să se reducă la rezolvarea unui problem liniar care se rezolvă rapid.

(Principiu metodei de antrenare supervizată, ca)

Def 9 O rețea de funcții radiale - RBF - este un triplet de forma:

$$((X, \Gamma), \{F_{w,p} : R^d \to R^K\}, \Psi)$$

unde:

- (X, Γ) este un graf orientat cu trei nivele având nodurile din multimea X, cu $\text{Card}(X) = d + M + K$, dispuse astfel:
  - pe stratul de intrare, d noduri senzoriale având un perceptron;
  - pe stratul funcțional oarecare, M noduri senzoriale având un perceptron;
  - pe stratul de ieșire, K noduri senzoriale având un perceptron.
- $F_{w,p} : R^d \to R^K$ este funcția implementată de rețea definită de

vezi verso

$F_{W,P}(x) = (y_1^{(x)}, y_2^{(x)}, \dots, y_K^{(x)})^t$ cu $F_{W,P}(x) = (y_1^{(x)}, y_2^{(x)}, \dots, y_K^{(x)})^t$

$$y_k(x) = \sum_{j=0}^{M} w_{kj}\, \phi_j(x) \qquad (\forall) k = \overline{1,K}$$

funcțiile de transfer ale celor K perceptroni de pe stratul de ieșire

Cu $\{w_{kj}\}_{k=\overline{1,K}}^{j=\overline{0,M}}$ coeficienții de ponderare corespunzători celor M perceptroni de pe stratul ascuns ai celor K perceptroni de pe stratul de ieșire ($\{w_{k0}\}_{k=\overline{1,K}}$ sunt considerați deplasările perceptronilor de pe stratul de ieșire).

iar

$$\phi_j(u) = \exp(-u) \qquad j = \overline{1,M}$$

$$\phi_0(x) = 1 \qquad (\forall) x \in \mathbb{R}^d$$

sunt funcțiile radiale ce funcționează ca funcții de transfer ale celor M perceptroni de pe stratul ascuns.

Funcțiile de integrare ale perceptronilor de pe stratul ascuns sunt de forma

$$u \equiv G_P(x) = \tfrac{1}{2}(x-\mu_j)^t \Sigma_j^{-1} (x-\mu_j) \qquad j = \overline{1,M}$$

în depinzând de parametrii $P = \{\mu_j, \Sigma_j\}_{j=\overline{1,M}}$, unde $\{\mu_j\}_{j=\overline{1,M}} \in \mathbb{R}^d$ se numesc vectori prototip iar $\Sigma_j \in \mathcal{M}_{d\times d}(\mathbb{R})$ simetrice.

Funcțiile de integrare ale perceptronilor de pe stratul de ieșire sunt produsul scalar în $\mathbb{R}^{M+1}$ în depinzând de parametrii $W = \{w_{kj}\}$

- Ψ este regula de "învățare" ce stabilește corespondența dintre funcția $F_{W,P}$ (adică de determinarea a parametrilor W și P) și care se realizează în doi pași:
  1) printr-o metodă nesupervizată pentru parametrii P.
  2) printr-o metodă supervizată pentru parametrii W.

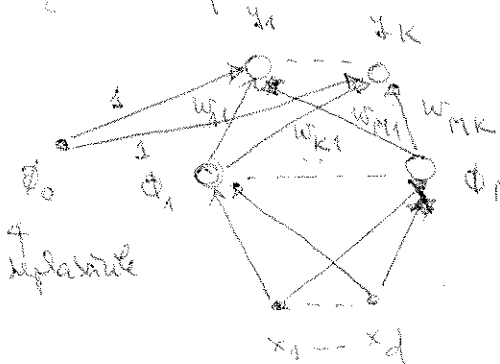

Fig. 14. Exemplu de RBF

Inteligență artificială

Curs 2

Observații 1) Funcțiile radiale sunt cazuri particulare ale funcțiilor nucleu, adică funcții (nenegative) care au proprietatea asimptotică orizontală la $\pm\infty$; care își concentrează majoritatea "masei" în jurul unui punct dat; Sunt utilizate de mai mulți și sunt numite funcții radiale locale.
O altă alegere de funcții radiale locale, decât cea exponențială utilizată în RBF, este:

$$\phi(x) = (x^2+\sigma^2)^{-\alpha}, \quad \alpha > 0.$$

În rețelele cu funcții radiale neliniare sunt utilizate, mai des, funcțiile

$$\phi(x) = x^2 \ln x \quad \text{sau} \quad \phi(x) = (x^2+\sigma^2)^{\beta} \quad 0<\beta<1$$

MATLAB este implementată funcția radială din def 9

2) Spuneam, în introducere, că activarea unui perceptron este dată de distanța dintre vectorul de intrare și un vector prototip. Funcția de integrare implementată de perceptronii de pe stratul ascuns din def 9 este inspirată din densitatea de probabilitate a variabilei normale multidimensionale, motiv pentru care funcțiile $\phi \circ G_{\mu,\Sigma}$ s.n. și funcții radiale gaussiene, ce măsoară distanța "redusă" (adimensionalizată datorită metricii Σ^{-1}) de la vectorul de intrare la vectorul prototip. Dacă vectorul de intrare provine dintr-o populație repartizată normal multivariat atunci vectorul prototip este valoarea unei variabile aleatoare cu matrice de varianță-covarianță Σ. Pentru reducerea numărului de parametri ce trebuiesc "învățați" (de ex. μ și $\frac{d(d+1)}{2}$ pt Σ, adică în total $\frac{d(d+3)}{2}$ parametri pt. fiecare perceptron $j=\overline{1,m}$ de pe stratul ascuns) în practică se utilizează funcții de integrare de tipul $G_{\mu,\sigma}(x) = \frac{1}{2\sigma_j^2}\|x-\mu_j\|^2$ (matricea de varianță-covarianță se reduce la $\sigma^2 I_{d\times d}$, în acest caz fiind necesară doar d+1 parametri de învățat pt. fiecare din cei m perceptroni de pe stratul ascuns).
Cum perceptronii prezentați în acest curs sunt perceptroni fără memorie coeficienții $\mu_j, \frac{1}{2\sigma_j^2}$, $j=\overline{1,m}$ care trebuiesc fie dați (vezi observația 3) fie estimați sunt "memorați" de rețea sub forma ponderilor

vezi verso

(în MATLAB este implementat acest ultim caz)

ce ponderează conexiunile intrărilor cu perceptronii de pe stratul respectiv deplasările acestor perceptroni, astfel:

$w_j^0(1) = \frac{1}{2\sigma_j^2}$ deplasările - bias-urile - perceptronilor $j = \overline{1,M}$

$w_j^k(1) = \mu_k^j$ cu $k = \overline{1,d}$ iar $\mu_j = (\mu_1^j, \mu_2^j, \ldots, \mu_d^j)$

3) Dacă $M = N$ (adică numărul de perceptroni de pe stratul ascuns este egal cu numărul de vectori din selecția de învățare) atunci conform lui Powell 1987 citat de Bishop în 1995, RBF implementează funcția ce interpolează exact punctele din mulțimea de învățare S, adică

$F_{W,\sigma}(x): \mathbb{R}^d \to \mathbb{R}$ cu $F_{W,\sigma}(X_i) = Y_i$, $i = \overline{1,N}$ și $(X_i, Y_i) \in S$

Într-adevăr, dacă scriem pe $F_{W,\sigma}$ (spre simplificare notat cu F când nu este nici un risc de confuzie)

$F(x) = \sum_{i=1}^{M} w_i \phi(\|x - X_i\|)$ cu ϕ funcție radială iar $\{w_i\}$ parametrii

atunci punând condiția de interpolare exactă (adică $F(X_i) = Y_i$) se obține sistemul

$\Phi w = t$ unde $\Phi \in M_{N \times N}(\mathbb{R})$ cu $\Phi_{ij} = \phi(\|X_i - X_j\|)$

$w \in M_{N \times 1}(\mathbb{R})$ iar $t \in M_{N \times 1}(\mathbb{R})$ cu $t_i = Y_i$

cu W vectorul necunoscutelor, iar dacă Φ^{-1} există (lucru arătat de Micchelli 1986 pentru o clasă largă de funcții, în ipoteza că punctele din S sunt distincte) atunci

$w = \Phi^{-1} t$ ceea ce demonstrează afirmația referitoare la interpolarea exactă a unei mulțimi de puncte.

Un mare număr de lucrări (Broomhead & Lowe 1988, Moody & Darken 198[illegible] etc.) au arătat că dacă vectorii prototip nu sunt dați inițial atunci M, numărul de perceptroni de pe stratul ascuns, poate fi ales mult mai mic decât N, numărul de vectori din selecția de învățare fără ca acest lucru să influențeze asupra proprietății de aproximare optimală pe care RBF-ul îl are (conform lui Girosi & Poggio 19[illegible] citat de Bishop 1995) [O schemă de aproximare este optimală dacă în mulțimea funcțiilor implementate - mulțimea funcțiilor generate de toate valorile parametrilor W - există o funcție care minimizează eroarea de aproximare].

Curs 2

• **Antrenarea RBF**

Tratăm în def 9 că antrenarea RBF presupune două etape: o etapă supervizată pentru determinarea ponderilor de conectare dintre perceptronii de pe stratul 1 și cei de pe stratul 2, și o etapă nesupervizată pentru determinarea vectorilor prototip și a factorilor de scară $\{\sigma_j^2\}$. Vom discuta mai întîi etapa de învățare supervizată, cum am discutat-o în detaliu în cazul RPM cu funcții de transfer liniare.

1) Învățarea supervizată

Fie ieșirile celor K perceptroni de pe stratul 2

$$y_k(x) = \sum_{j=0}^{M} w_{kj}\,\phi_j(x)$$

(cu convenția din def 9 referitoare lui $\phi_0(x)$)

iar matricial

$$y(x) = W\phi \quad \text{unde} \quad W = \{w_{kj}\} \text{ și } \phi = (\phi_j).$$

Se consideră riscul funcțional empiric

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K}\left(y_k(x_i) - t_{ik}\right)^2$$

(ținta este aici vectorială)

Derivând și impunând condiția de anulare a derivatelor (ca în § "Rețele de perceptroni pt. problema clasării liniare") rezultă

$$\Phi^T \Phi W^T = \Phi^T T \quad \text{cu} \quad T = (t_{ik}),\ \Phi = (\phi_j(x_k))$$

$$\Longrightarrow \quad W^T = \Phi^{\dagger} T \quad \text{cu } \Phi^{\dagger} \text{ - pseudo-inversa lui } \Phi.$$

În practică sistemul de mai sus se rezolvă folosind descompunerea în valori singulare pentru a evita problemele legate de proasta condiționare a matricii Φ.

2) Învățarea nesupervizată

Bishop 1995 enumeră mai multe metode de învățare nesupervizată: metoda celor K-medii (K-means clustering algorithm); metoda celor mai mici pătrate ortogonale (~~least~~ orthogonal least squares); metoda mixturilor gaussiene (gaussian mixture models).

În MATLAB este implementată metoda celor mai mici pătrate ortogonale - OLS - (funcția solverb) motiv pt. care o vom prezenta în cele ce urmează. În plus, metoda OLS optimizează și arhitectura rețelei în sensul că stabilește numărul minim M de perceptroni de pe stratul ascuns necesari și riscul empiric pentru funcția implementată de RBF să fie minim.

Cu titlul informativ metoda K-mediilor este implementată în SPAD pentru metodele de clasificare automată.

Curs 2

Procedura de estimare a parametrilor β și a lui M este iterativă; în plus are ca "efect lateral" - side effect - și determinarea parametrilor W.

Pasul 0 (Inițializări)

$j = 0$ - indice ce numără perceptronii pe stratul ascuns

$E_j = \text{Inf}$ - valoare ce memorează pasul empiric al RBF până la algoritmul OLS

$S_j = \emptyset$ - mulțimea vectorilor prototip (submulțime a vectorilor S din eșantionul de antrenare)

$\bar{S}_j = S$ deci def $\bar{S}_j = (S, s_j)$

$I_j = \{1, 2, \dots, N\}$ - mulțimea indicilor vectorilor din $\bar{S}_j$

Pasul 1 (Antrenarea rețelei)

Pentru $i \in I_j$

Pasul 1.1 (antrenarea nesupervizată)

~~Se construiește rețeaua cu~~

$S_c = S_j \cup \{X_i\}$; $\bar{S}_c = \bar{S}_j \setminus \{X_i\}$; $\mathcal{P} = \{S_c, 1\}$

Se construiește RBF cu mulțimea $\mathcal{P} = \mathcal{P}$

Pasul 1.2 (antrenarea supervizată)

Pt rețeaua de la pasul 1.1 se calculează, prin antrenare supervizată, mulțimea ponderilor W_i și riscul empiric funcțional E_i pe baza eșantionului de învățare $\bar{S}_c$.

Pasul 2 (construirea rețelei)

Se calculează $i^* = \arg\min E_i$.

~~Rețeaua găsită este de~~

$S_{j+1} = S_j \cup \{X_{i^*}\}$; $\bar{S}_{j+1} = (S, s_{j+1})$; I_{j+1} = indicii vectorilor din $\bar{S}_{j+1}$

$E_j = E_{i^*}$; ~~$W_j = W_{i^*}$~~ $M = j+1$;

$\beta = \{S_{j+1}, 1\}$; $W = \{W_{i^*}\}$

Rețeaua găsită are parametrii (β, W) are M perceptroni pe stratul ascuns, și realizează un risc empiric funcțional $= E$.

Pasul 3 (S-a terminat algoritmul?)

Dacă $[(E > \varepsilon - \text{dat}) \vee j < M_{max}]$ atunci $j \leftarrow j+1$ și salt la Pasul 1;

altfel STOP

(M_{max} = număr maxim de perceptroni pe stratul ascuns, admis)

→ vezi verso

Observație 1) Dacă $NMIP = N$ atunci $E = 0$ dar pentru a obține o bună generalizare a algoritmului $NMIP << N$ chiar dacă este posibil ca $E > 0$ pentru ~~selecția~~ selecția de antrenare S

2) Procedura OLS implică resurse de calcul deosebite - este o metodă "computer intensive" - datorită pasului 1.2 care se execută de $card(S_e) = N - j$ ori, pt. un j fixat, variind ~~una de la 1 la NMIP-dat~~ unde trebuie rezolvat un sistem algebric liniar de jK necunoscute prin metoda descompunerii în valori singulare.

3) Datorită faptului că $NMIP \neq N$ metoda OLS nu este în general optimală căci la pasul 1.1 parametrii β sunt stabiliți independent de valorile țintă T.

Inteligență artificială

Curs 2

Datorită observațiilor de mai sus (pct 3) în special) s-a încercat implementarea unei metode de antrenare a RBF total supervizată în sensul că și parametrii Φ sunt estimați astfel încât să minimizeze riscul empiric funcțional calculat pe selecția de învățare. Urmând acest demers la sistemul de ecuații liniare

$W' = \Phi^{+} T$ se mai adaugă sistemele de ecuații

neliniare

$$\frac{\partial E}{\partial \sigma_j} = \sum_{i=1}^{N} \sum_{k=1}^{K} \{y_k(X_i) - Y_i^k\} w_{kj} \exp\left(-\frac{\|X_i - \mu_j\|^2}{2\sigma_j^2}\right) \cdot \frac{\|X_i - \mu_j\|^2}{\sigma_j^3} = 0 \quad j = \overline{1,M}$$

și

$$\frac{\partial E}{\partial \mu_j^l} = \sum_{i=1}^{N} \sum_{k=1}^{K} \{y_k(X_i) - Y_i^k\} w_{kj} \exp\left(-\frac{\|X_i - \mu_j\|^2}{2\sigma_j^2}\right) \cdot \frac{(X_i^l - \mu_j^l)}{\sigma_j^2} = 0 \qquad \begin{matrix} j = \overline{1,M} \\ l = \overline{1,d} \end{matrix}$$

Observații 1) Spre deosebire de OSL metoda de antrenare total supervizată necesită cunoașterea prealabilă a numărului de perceptroni de pe stratul ascuns.

2) Datorită sistemelor neliniare și metoda total supervizată este "computationally intensive" și duce la aflarea unor minime locale ale riscului empiric. Totuși există speranța că dacă funcțiile radiale sunt bine localizate atunci pentru un vector din selecția de antrenare doar o mică parte din totalul funcțiilor radiale vor fi semnificativ activate și deci doar acele funcții vor fi actualizate la răspunsul corespunzător vectorului de intrare. În această lumină procedura de antrenare poate fi accelerată simțitor dacă, pentru a evita multimea calculelor nesemnificative, din start sunt alese acele funcții radiale ce sunt relevante pentru selecția de antrenare S. Omohundro (1987) a publicat tehnici eficiente de stabilire a funcțiilor radiale relevante.

3) Rezolvarea numerică a sistemelor neliniare de mai sus implică folosirea unor metode iterative de tipul celor descrise în cursul 1 pt perceptronul formal. Aceste metode necesită inițializarea lor cu valori de start. În cazul RPM acești valori erau generate aleator; în cazul RBF folosirea unora dintre metodele de antrenare nesupervizată ~~pentru generarea valorilor inițiale este~~ este des recomandată în literatura de specialitate.

vezi verso

4) Antrenarea total supervizată a RBF nu garantează că funcțiile radiale rămân localizate (funcțiile sunt localizate dacă mărimea $\{x \in \mathbb{R}^d / \|\Phi_j(x)\| > \varepsilon\}$ este relativ mică; de exemplu volumul unor hipersfere de rază ε). Se observă că, acest volum este descrescător în raport cu σ_j ($\sigma_j \to \pm\infty$ atunci când valorile sunt satisfăcute dar, din observații de mai sus, funcțiile nu mai sunt localizate). Simulările numerice au confirmat și ele această "umbră prezumtivă" arătând că o rebalansare a perceptronilor de pe stratul ascuns evaluează toate funcții de transfer (adică funcții radiale) cu o plajă foarte largă de răspuns (Moody & Darken 1989). Așadar, unul din avantajele de bază ale RBF, anume antrenarea rapidă și interpretabilitatea stratului ascuns, se pierde dacă se adoptă antrenarea total supervizată.

5) MATLAB-ul raportează următorii timpi de antrenare:

Funcție	Metoda	Timp	Nr. de iterații	Operații
TRAINBP	Propagare înapoi	259.1	4123	216.033.235
TRAINBPX	Propagare înapoi rapidă	42.4	570	2.971.696
TRAINLM	Pe bază de algoritmului Levenberg-Marquardt	3.3	5	315.769
SOLVERB	Funcții radiale	1.9	5	38.305

Curs 2

• Comparație între RBF și RPM

RBF și RPM ~~sunt~~ au rolurile similare în sensul că pun la dispoziție tehnici de aproximare a funcțiilor relativ multidimensionale. În ambele cazuri funcțiile sunt exprimate sub forma de combinări de aplicații scalare parametrizate. Structura particulară a celor două tipuri de rețele este însă foarte diferită. Unele din deosebirile importante ale ~~vector~~ RBF comparativ cu RPM sunt:

(1) Funcția de ieșire a unui perceptron asemănător într-o RPM depinde de suma ponderată a intrărilor transformată de funcția de transfer monotonă. Această funcție este deci constantă pe suprafețe ~~formate~~ din ~~plane~~ hiperplane (d-1)-dimensionale ~~paralele~~, în spațiul d-dimensional. În contrast, perceptronul asemănător într-o RBF folosește ca funcție de intrare distanța de un vector prototip iar ca funcție de transfer o transformare ce este, de obicei, o funcție bicalitată. În concluzie această funcție este constantă pe hipersfere (d-1)-dimensionale concentrice (sau mai general pe hiperelipsoizi (d-1)-dimensionali) concentrici.

(2) O RPM ~~formează~~ o reprezentare distribuită în spațiul valorilor funcțiilor de transfer ~~totale~~ ale perceptronilor ascunși deoarece, pentru un vector de intrare dat, mai multe perceptroni ascunși contribuie la evaluarea ~~ieșirii~~. În timpul procesului de învățare valorile funcțiilor de transfer ale perceptronilor ascunși sunt astfel ajustate încât, combinate liniar ~~cu~~ ponderile ultimului strat să genereze ieșirea corectă pentru un interval de posibile valori de intrare. Punctele de minim local cât și erorile de trunchiere inerente unor metode numerice puternice reclamă ca procesul de convergență să fie lent chiar și în cazul unor metode de optimizare avansate. În contrast, o RBF cu funcții radiale localizate formează o reprezentare locală în raport cu un vector de intrare dat - în mod obișnuit doar foarte puține funcții de transfer sunt semnificativ nenule.

(3) RPM are în ~~cele mai~~ multe cazuri mai multe straturi, o topologie ~~complicată~~ și ~~neregulată~~ variată de funcții de activare. RBF are o topologie simplă formată din doar ~~două~~ straturi cu doar două tipuri de funcții de activare specifice.

vezi verso

(4) Toți parametrii unei RPM sunt, în mod obișnuit, determinați în același timp într-o procedură de antrenare globală bazată pe o metodă de învățare supervizată. O RBF, în schimb, este în mod obișnuit antrenată în două etape, cu parametrii funcțiilor radiale determinați printr-o metodă de învățare nesupervizată și folosește doar vectori de intrare iar ~~parametrii funcțiilor~~ ponderile vectorilor de ieșire sunt determinați printr-o metodă de învățare supervizată liniară rapidă și exactă. ~~se folosește doar~~