Recunoaşterea texturilor cu ajutorul reţelelor neuronale RBF

Conf. dr. Dan Laurenţiu Lacrămă
Universitatea "Tibiscus" din Timişoara

ABSTRACT. This paper is concerned wiyh the use of RBF Neural Networks in texture recognition. The network is build using the Dynamic Decay Algorithm in order to establish the RBF neurons centers. The basic software and interfaces implemantation is done in C++

1 Introducere

Reţelele neuronale RBF joacă in momentul de faţă un rol din ce în ce mai important în domeniul recunoaşterii formelor. Mulţi cercetători lucrează la investigarea teoretică a acestui tip de reţele. Pe de altă parte au apărut multe aplicaţii de succes. Principalele motive pentru aceste rezultate sunt usurinţa implementării şi eficienţa dovedită în clasificare.

Structura de bază a unei reţele neuronale RBF este prezentată în Figura 2.1. şi este formată din trei niveluri. Această structură este de obicei suficientă. Deşi seamănă cu structura perceptonului multistrat, fiecare nivel este diferit în cea ce priveşte funcţionarea. Primul nivel este pentru conectarea reţelei cu lumea exterioară iar al doilea nivel este nivelul ascuns. El realizează transformarea neliniară din spaţiul vectorului de intrare spre spaţiul vectorului intern care are în mod uzual o dimensiune mare. Ultimul nivel este nivelul de ieşire şi transformă spaţiul vectorului intern în cel de ieşire. Neuronii din stratul ascuns sunt de tip Radial Basis Function, celelalte straturi folosesc neuroni liniari LWI.

Metoda de procesare este unidirectională (feed forward), iar învățarea poate fi supervizată sau hibridă adică o combinație a învățarii prin autoantrenare şi a învățării supervizate. În general acestă rețea rezolvă problema funcției de aproximare deci se poate folosi pentru clasificare şi predicție.

Proiectarea unei rețele nouronale supervizate poate fi realizată într-o varietate de moduri diferite. Algoritmul cu propagarea inversă a erorii poate fi văzut ca o aplicație a unei metode de optimizare, cunoscută în statistică ca 'aproximare statistică'. Proiectarea rețelelor RBF poate fi vazută ca o problemă de aproximare în spații multidimensionale. Din acest punct de vedere învățarea este echivalentă găsirii suprafeței minime care include datele de antrenare, unde termenul 'suprafață minimă' este luat în sens statistic. Generalizarea este echivalentă folosirii suprafeței multi-dimensionale pentru a interpola datele de antrenament. În contextul rețelelor neuronale neuronii ascunşi asigură un set de funcții care constituie o bază arbitrară pentru vectorii de intrare, când sunt expandați în stratul ascuns. Aceste funcții poartă denumirea de 'radial basis functions'.

Problema clasificării modelelor transpusă într-un spațiu multi-dimensional neliniar e mai probabil a fi liniar separabilă decât într-un spațiu n-dimensional cu n de valoare mică. De aici s-a ajuns la numărul mare de neuroni din stratul ascuns. E posibil să scădem dimensiunea stratului ascuns dacă centrii spațiului ascuns sunt făcuți adaptabili. Când o rețea neuronală RBF e folosită la clasificări, problema este rezolvată prin transformarea ei într-un spațiu de mare dimensiune într-o manieră neliniară. Explicația teoretică este furnizată de teorema separabilității a lui Cover.

Instruirea rețelei neuronale radiale presupune două etape:

- stabilirea valorii pentru vectorul centru şi pentru dispersia fiecărui neuron RBF;
- determinarea printr-o metodă de instruire a ponderilor.

Instruirea poate fi iterativă sau neiterativă. În cazul în care matricea ponderilor este singulară putem determina ponderile folosind o metodă iterativă. Proiectarea şi instruirea rețelor RBF depind esențialmente de

modul de alegere a centrilor. Principalelor strategii care s-au impus în practică sunt următoarele:

- Alegerea aleatoare a unor centrii fixi;
- Calcularea centrilor prin auto-organizare;
- Selecția supervizată a centrilor;
- Metoda celor mai mici pătrate ortogonale;
- Metoda ajustării dinamice decadice (eng. Dynamic Decay Adjustment DDA).

2 Descrierea algoritmului DDA

Algoritmul DDA oferă o antrenare uşoară şi constructivă pentru rețelele neuronale RBF. Antrenarea folosind algoritmul DDA deseori conduce la o acuratețe la fel de mare în clasificare ca şi la rețelele "Multi Layer Perceptrons", dar cu antrenare mult mai rapidă.

O rețea RBF antrenată folosind algoritmul DDA este similară în structură cu o rețea MLP cu un singur strat ascuns. Numărul de unități din stratul de intrare reprezintă dimensiunea spațiului de intrare. Stratul ascuns conține neuronii RBF. Neuronii sunt adăugați în timpul antrenamentului. Startul de intrare este complet conectat stratului ascuns, adică fiecare neuron din stratul de intrare este conectat cu fiecare neuron din stratul ascuns. Fiecare neuron din stratul de ieşire reprezintă o posibilă clasă, rezultând o codare binară de la 1 la n. Pentru clasificare este folosită convenția "câştigătorul ia totul", adică neuronul cu cea mai ridicată activare determină clasa. Oricare neuron din stratul ascuns este conectat exact la un singur neuron din stratul de ieşire.

Principalele diferențe între rețeaua RBF şi MLP sunt funțiile de activare şi regulile de propagare ale stratului ascuns. În loc de a folosi funcțiile sigmoid sau alte funcții neliniare, rețeaua RBF foloseşte funcții gaussiene localizate ca funcții de activare. Calcularea distației euclidiene la un vector de referință înlocuieşte produsul scalar de la rețelele MLP:

$$R_i(\vec{x}) = \exp(-\frac{\|\vec{x} - \vec{r}_i\|^2}{\sigma_i^2}) \qquad (2.1.)$$

Dacă rețeaua primește la intrare vectorul x, R_i indică activarea unui neuron RBF cu vectorul de referinta r_i şi deviația standard σ_i.

Stratul de ieşire calculează ieşirea pentru fiecare clasă în felul următor:

$$f(\vec{x}) = \sum_{i=1}^{m} A_i \cdot R_i(\vec{x}) \qquad (2.2.)$$

unde m indică numărul de neuroni RBF şi A_i fiind greutatea fiecărui neuron RBF.

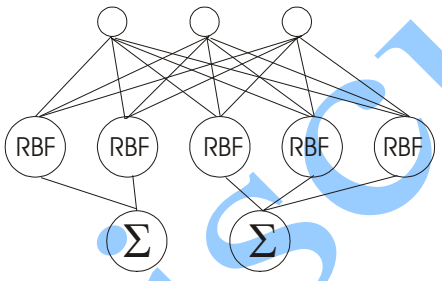

Figura 2.1. Rețeaua neuronală RBF

Un exemplu de rețea RBF este prezentată în Figura 2.1. Se remarcă inexistența vreunei legături directe între intrare şi ieşire. Vectorul greutăților conexiunilor dintre stratul de intrare şi un neuron RBF din stratul ascuns reprezintă centru pentru funcția de activare. Distanța euclidiană dintre vectorul de intrare la vectorul de referință(prototip) este folosit ca şi intrare pentru neuronul RBF ceea ce conduce la un raspuns local. Dacă vectorul de intrare este apropiat de prototip, neuronul va avea o activare ridicată. Contrar, activarea va fi aproape de 0 pentru distanțe mari. Fiecare neuron de ieşire va face o sumă ponderată a tuturor activărilor neuronilor RBF aparținând clasei respective.

Algoritmul DDA introduce ideea de *potrivire* şi *conflict* în *zona de conflict*. Doi parametrii: θ+ şi θ- sunt introduşi în acest scop.

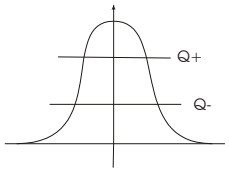

Figura 2.2. Parametrii θ+ şi θ

Cei doi parametrii sunt utilizați la definirea zonei de conflict, unde nici un alt prototip al clasei de conflict nu este permis a exista. Normal θ^+ este mai mare ca θ^-, ceea ce conduce la o zonă de conflict unde potrivirea sau conflictul nu sunt permise a exista. Utilizând aceşti parametrii algoritmul construieşte rețeaua dinamic şi ajustează neuronii individual. Principalele proprietăți ale aloritmului DDA sunt:

- **antrenare constructivă** – noi neuroni RBF sunt creați în funcție de necesități. Rețeaua este creată de la zero şi numărul necesar de neuroni RBF este determinată în timpul antrenării. Neuronii RBF sunt ajustați dinamic în timpul antrenării.
- **antrenare rapidă** – de obicei antrenarea se termină dupa 5 epoci, datorită naturii algoritmului. Sfârşitul antrenării este indicat clar.
- **convergența garantată** – algoritmul de învățare asigură terminarea procesului de învățare.
- **doi parametrii necritici** – doar cei doi parametrii θ^+ şi θ^- trebuie ajustați manual. Din fericire acesti doi parametrii nu sunt critici. Experimental s-a observat că valorile $\theta^+ = 0.4$ şi $\theta^- = 0.2$ sunt cele mai potrivite.
- **proprietăți garantate** ale rețelei – după antrenare rețeaua păstrează câteva condiții pentru toate imaginile: clasificările greşite sunt sub o anumită limită iar clasificările corecte sunt peste o anumită limită.

Algoritmul DDA este compus din doi paşi. În timpul antrenamentului la fiecare imagine un nou neuron RBF este introdus sau greutatea unui neuron este ajustata (“commit”). În ambele cazuri neuronii RBF aflați în conflict cu neuronul câştigător sunt ajustați în sensul reducerii greutății(“shrink”). Aceasta garantează că fiecare imagine din setul de

antrenare este acoperită de neuronul RBF corespunzător clasei corecte şi nici unul din neuronii RBF ai claselor în conflict nu are un răspuns nepotrivit.

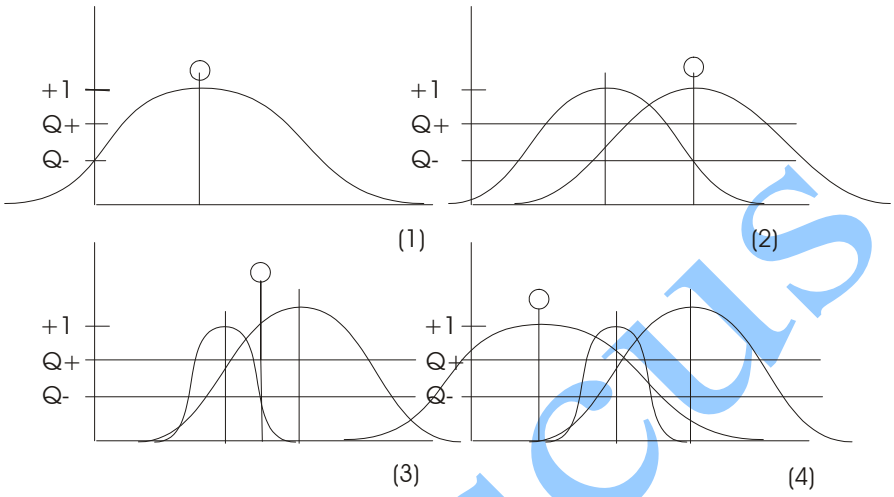

Figura 3.1. Etapele antrenării rețelei

Doi parametrii sunt introduşi la acest nivel: θ^+ şi θ^-. Pentru a realiza un nou prototip nici unul dintre neuronii RBF ai clasei corecte nu trebuie să aibă activarea mai mare ca θ^+ şi pentru a realiza operația de "shrink", nici unul dintre neuronii RBF ai claselor în conflict nu trebuie să aibă activarea mai mare ca θ^-. În imaginea de mai jos sunt prezentate câteva etape ale antrenării.Figura 3.1:

1. imagine de clasa A este întâlnită şi un nou neuron RBF este creat
2. imagine de clasa B este întâlnită, un nou neuron RBF este creat şi deviația primului neuron este micşorată.
3. nouă imagine de clasă B este întâlnită, şi primului neuron i se micşorează din nou deviația.
4. nouă imagine de clasă B crează un nou neuron.

După terminarea antrenamentului două condiții sunt adevărate pentru toate perechile intrare – ieşire (x,c): **a.** cel puțin un prototip al clasei c are activarea mai mare sau egală cu θ^+; **b.** toate prototipurile claselor conflictuale au activările mai mici sau egale cu θ^-.

3 Descrierea funcționării programelor

Pentru a rezolva problema recunoaşterii texturilor cu rețele neuronale RBF, am abordat următoarea soluție: am implementat trei aplicații distincte care rezolva, fiecare, câte o sub-problemă.

3.1 Neural Network Trainer

Această aplicație realizează crearea şi antrenarea unei rețele neuronale. Programul are ca intrare un fişier Neural Network Training (*.nnt), iar ca ieşire un fişier Neural Network Description (*.nnd). Fişierele de tipul Neural Network Training sunt fişiere de descriere a rețelei neuronale ce urmează a fi create şi de specificare a setului de date de antrenament.

Fişierele de tipul Neural Network Description sunt fişiere de descriere ce descriu rețele neuronale antrenate şi gata de testare şi utilizare. Aplicația poate fi setată pe o prioritate mai mare sau mai mică decât prioritatea normală a unei aplicații standard Windows, aceasta opțiune fiind utilă în cazul unei antrenări îndelungate.

3.2 Neural Network Tester

Această aplicație realizează testarea rețelei neuronale antrenată de Neural Network Trainer şi calculează unele date statistice cum ar fi rata erorilor. Ca intrare, aceasta aplicație are două tipuri de fişiere: Neural Network Description (*.nnd) şi Test Data Description (*.tdd), iar ieşirea o reprezintă datele statistice. Ca şi Neural Network Trainer, Neural Network Tester poate fi setată pe o prioritate mai mare sau mai mică decât prioritatea normală a unei aplicații standard Windows, aceasta opțiune fiind de asemenea utilă în cazul unor date de test mai mari.

3.3 Image Selector

Această aplicație are rolul de a pregătii datele de antrenament şi de testare pentru rețelele neuronale. În scopul clasificării texturilor (sau al altor tipuri de imagini) imaginile ce conțin texturi trebuie preprocesate pentru a pune în

evidență caracteristicile care deosebesc diferitele clase de texturi între ele. Acest lucru se realizează aplicând fiecărei imagini o stivă de proceduri de preprocesare. Stiva este creată de utilizator prin adăugarea procedurilor de preprocesare dintr-o librărie de astfel de proceduri.

Rolul aplicației Image Selector este de a crea un fişier de descriere a datelor de antrenament, ocazie cu care se vor specifica de asemenea stiva de proceduri de preprocesare cât şi tipul rețelei ce urmează a se crea (un fişier de tipul *.nnt). De asemenea este posibilă şi exportarea unui fişier de descriere a datelor de testare (tdd). Aplicația poate de asemenea realiza procesarea fişierelor imagine din listă şi poate afişa imaginea obținută în urma procesării (aplicării procedurii de preprocesare curente sau a întregii liste de preprocesare).

4 Rezultate experimentale

Pentru a evidenția importanța parametrii neuronilor şi a metodei de învățare am realizat următarea serie de teste:

Test 1 Standard Deviation 0.90, QMinus 0.20, QPlus 0.80 ***Discriminări Corecte: 84.34 %***

Test 2 Standard Deviation 0.90, QMinus 0.020, QPlus 0.040 ***Discriminări Corecte: 64.14 %***

Test 3 Standard Deviation 0.70, QMinus 0.20, QPlus 0.40 ***Discriminări Corecte: 82.32 %***

Toate precedente au fost realizate pentru doua clase de texturi, antrenarea s-a realizat pe un lot de 86 de texturi iar testarea pe un set de date de 396 de texturi.Se poate observa deci:

- o scădere dramatică a discriminărilor corecte în cazul antrenării cu valori mai mici ale parametrilor Q+ şi Q- ale algoritmului DDA. Această scădere se datorează unui antrenament defectuos, generînd neuroni cu deviații standard mici, cu putere de acoperire a spoțiului de intrare relativ mică.

- o uşoară scădere a procentajului de reuşită in cazul folosirii unei deviații standard pentru funcțiile gaussiene mai mici. (datorată unei acoperiri mai mici a spațiului datelor de test)

În construirea unei rețele neuronale RBF, antrenată prin metoda DDA se face un compromis între puterea de acoperire a spațiului de intrare şi "siguranța de discriminare" (1/(procentul de ieşiri inconcludente – procentul de ieşiri inconcludente)). Valori mici ale parametrilor Q+ şi Q- ai algoritmului DDA duc un mod de tratare a conflictelor care generează neuroni cu deviații mici (în urma aplicării operației shrink mai "puternic"), deci se obține o acoperire slabă a spațiului de intrare dar o siguranță de discriminare mai mare. Valori mari ale parametrilor Q+ şi Q- ai algoritmului DDA duc un mod de tratare a conflictelor care generează neuroni cu deviații mari (în urma aplicării operației shrink mai "slab") , deci se obține o acoperire bună a spațiului de intrare dar o siguranță de discriminare mai mică.Alegerea valorilor parametrilor Q+ şi Q- este echivalentă cu alegerea "grosimii" liniei de demarcație între spațiile claselor.Un rezultat care a atrage atenția asupra puterii de invarianță a momentelor Hu este discriminarea între categorii de texturi similare, dar de culori şi dimensiuni diferite: 97.15% discriminări corecte.

Bibliografie

[Boo94] **Booch, G.**, *Object-Oriented Analysis and Design with Applications*, Addison-Wesley, 1994;

[BT00] **Bates, J., Tompkins, T.,** *Utilizare Visual C++ 6.0*, Ed. Teora, Bucureşti, 2000;

[DM86] **Davis, L.S., Mitiche, A.**, *Computer Graphics and Image Processing*, Dover Publishing Co., Toronto, 1986