UNIVERSITATEA "POLITEHNICA" TIMIŞOARA

FACULTATEA DE ELECTRONICĂ ŞI TELECOMUNICAȚII

TEZĂ DE DOCTORAT

Recunoaştere facială bazată pe procesare neuronală paralelă şi metoda operatorului de interes

Autor:
Asist.ing.Cătălin-Daniel CĂLEANU

Conducător ştiinţific:
Prof.dr.ing. Corneliu I. TOMA

2001

CUPRINS

CAP.5 Metode folosite în recunoaşterea facială

CAP.6 Propuneri privind realizarea unui sistem neuronal de recunoaştere facială

CAP.7 Implementarea unui sistem de recunoaştere facială

CAP. 8 Concluzii

ANEXA 1

ANEXA 2

ANEXA 3

ANEXA 4

ANEXA 5

CAPITOLUL 1

Introducere

În cadrul acestei teze este abordată problema prelucrării digitale a imaginilor (PDI) prin intermediul rețelelor neuronale artificiale (RNA). Ca aspect particular al celor sus-menționate este detaliată problema detecției și recunoașterii faciale prin intermediul RNA.

În acest scop teza este structurată după cum urmează:

CAP. 1 prezintă obiectul și structura tezei.

CAP. 2 se ocupă de motivațiile abordării unei astfel de teme și de formularea problemei.

CAP. 3 se referă la avantajele oferite de RNA și prezintă sintetic etapele ce le presupune o aplicație generală a PDI. Apoi sunt identificate, pentru fiecare dintre aceste etape, modalități de implementare prin intermediul rețelelor neuronale artificiale: RNA în filtrarea temporală și spațială, RNA în codarea și compresia imaginilor, RNA în segmentarea imaginilor, RNA în reprezentarea imaginilor, RNA în recunoașterea și interpretarea imaginilor. Sunt evidențiate avantajele pe care le oferă soluțiile neuronale față de soluțiile clasice sau convenționale (în sensul în care nu folosesc RNA) și se desprinde ideea că RNA se constituie într-o alternativă viabilă pentru domeniul PDI.

CAP. 4 oferă o succintă trecere în revistă a chestiunilor legate de tipurile de RNA și algoritmii aferenți de antrenament. Autorul tezei identifică, pe baza analizei algoritmilor de antrenament pentru RNA cu propagare înainte a semnalului (feedforward) anumite probleme specifice folosirii RNA în PDI. Drept urmare se propun doi algoritmi de antrenament care să soluționeze aceste probleme. Se arată că cel bazat pe adaptarea fuzzy a ratei de învățare după o metodă propusă de autorul tezei, dă rezultate foarte bune în momentul în care vectorii de intrare au o dimensionalitate ridicată (număr de componente mai mare decât 10^3). Pentru cazul în care vectorul de

intrare are dimensionalitate mică sau medie (zeci sau sute de componente) se propune folosirea unui algoritm de antrenament derivat din cei bazaţi pe metoda gradientului conjugat.

CAP. 5 este dedicat realizărilor de până în prezent în domeniul recunoaşterii faciale. Autorul propune o împărţire a acestor metode în: convenţionale, pentru cele care nu folosesc reţele neuronale şi metode bazate pe RNA. Din prima categorie se prezintă unele dintre cele mai semnificative abordări, realizându-se totodată şi o circumscriere pentru acestea: metode bazate pe trăsături geometrice, metode bazate pe analiza componentelor principale, metode bazate pe compararea şabloanelor, metode bazate pe compararea reţelelor elastice. În continuare se realizează o privire de ansamblu asupra metodelor recunoaşterii faciale care folosesc reţele neuronale: metodă bazată pe folosirea reţelelor neuronale cu autoorganizare şi convoluţionale, metodă bazată pe reţele neuronale cu decizie probabilistică, metode bazate pe arbori neuronali, metode ce folosesc reţele neuronale bazate pe funcţii radiale, metode bazate pe reţele neuronale de tip perceptron multistrat (MLP) şi cu învăţare prin cuantizare vectorială (LVQ), metode bazate pe reţele neuronale tip grupuri de arbori cu toleranţă adaptivă, metode bazate pe reţele neuronale ART. În finalul capitolului sunt analizate avantajele şi dezavantajele metodelor prezentate şi se desprinde ideea referitoare la utilitatea unei metode hibride convenţional-neuronale.

CAP 6. propune pe baza observaţiilor din capitolele anterioare o arhitectură pentru un sistem de recunoaştere facială. Contribuţia autorului tezei se manifestă aici prin propunerea folosirii unui algoritm de extragere a trăsăturilor, metoda operatorului de interes, care a fost folosit până în prezent doar la navigaţia roboţilor mobili autonomi. Se demonstrează avantajele pe care la aduce folosirea acestei metode de extragere a trasăturilor: reducerea dimensionalităţii vectorilor ce urmează a fi procesaţi de clasificatori şi de aici timp redus de procesare. Totodată, prin extragerea doar a informaţiei definitorii, esenţiale, din imaginile faciale se obţine o certă îmbunătăţire a performanţelor clasificării. Se investighează efectul rezoluţiei imaginilor şi influenţa dimensiunii ferestrei de extragere a trăsăturilor asupra ratei clasificării. Mai mult,

autorul propune o arhitectură paralel-ierarhică de RNA tip perceptron multistrat (RNA-MLP) şi arată că prin procesarea paralelă a unor regiuni disjuncte din imaginea facială pot fi obţinute rezultate mai bune decât în cazul folosirii unui singur clasificator global.

În CAP. 7 se tratează problematica implementării sistemului prezentat în capitolul anterior. Sunt rezumate cerinţele care trebuie să fie îndeplinite de către o aplicaţie ce are ca obiect detecţia şi recunoaşterea facială şi sunt discutate posibilităţile software şi hardware de realizare a sistemului. Dintre modalităţile software sunt comparate implementările MATLAB v.5.3 şi Microsoft Visual C++ 5.0. Se decide că mediul MATLAB este potrivit pentru dezvoltarea aplicaţiei şi se evidenţiază proprietăţile statice şi dinamice pe care trebuie să le ofere interfaţă grafică utilizator pentru această aplicaţie. În final se prezintă modul de operare a programului realizat după cerinţele şi principiile sus-menţionate. Se propune însă şi o soluţie hardware bazată pe procesorul de semnal de uz general, preţ scăzut, TMS320VC33 folosit ca şi coprocesor în conjuncţie cu un microcontroller de tip 8051.

CAP. 8 cuprinde concluziile referitoare la tema prezentată, contribuţiile originale ale autorului şi direcţii viitoare de cercetare.

În ANEXE este prezentat codul sursă MATLAB pentru toate experimentele prezentate în cadrul tezei, inclusiv pentru aplicaţia de detecţie şi recunoaştere automată a unei imagini faciale iar BIBLIOGRAFIE cuprinde totalitatea referinţelor bibliografie, în ordinea menţionării acestora în textul prezentei teze.

CAPITOLUL 2

Intenţii şi motivaţii

2.1 Intenţii

În prezenta teză autorul doreşte să abordeze problematica prelucrărilor digitale de imagini din perspectiva reţelelor neuronale artificiale. În mod particular s-a ales tema **detecţiei şi recunoaşterii faciale folosind reţele neuronale artificiale**. Motivele acestei alegeri sunt subliniate în subcapitolul următor.

În principal există două tipuri de probleme [1] specifice recunoaşterii faciale:

a) Găsirea unei persoane într-o bază de date largă, de exemplu cele ale poliţiei. Aceste sisteme returnează o listă cu persoanele care seamănă cel mai mult cu cea căutată. De regulă o singură imagine sau un număr mic de imagini sunt disponibile pentru persoana căutată iar recunoaşterea nu trebuie efectuată în timp real [2].
b) Identificarea în timp real a unei persoane, de exemplu într-un sistem de supraveghere automată sau accesul unui grup de persoane la anumite facilităţi (clădiri, camere, calculatoare) [3]. De regulă sunt disponibile multiple imagini pentru persoanele ce vor trebui să fie recunoscute în timp real.

În cadrul prezentei lucrări este abordată în principal a doua categorie de sistem de recunoaştere facială. În consecinţă, autorul îşi propune să realizeze un **sistem de recunoaştere facială în timp real**, bazat pe reţele neuronale, care să ofere următoarele posibilităţi:

- Pornirea/oprirea unui flux video. Totodată aplicaţia trebuie să fie capabilă să captureze în mod automat cadre din fluxul video.
- Preprocesarea imaginilor achiziţionate.
- Detecţia automată a feţelor umane în imaginile preprocesate.
- Recunoaşterea imaginilor faciale izolate la punctul anterior prin compararea acestora cu cele deja existente în baza de date.
- Posibilităţi de manipulare a unei baze de date.

2.2 Motivaţii

Prezentul subcapitol îşi propune să răspundă la două întrebări: **de ce recunoaştere facială şi de ce recunoaştere facială bazată pe reţele neuronale** ?

Principalele motivele pentru care s-a ales o astfel de temă sunt următoarele:

- în ciuda eforturilor susţinute, materializate prin numeroase studii, conferinţe etc., **problema recunoaşterii faciale rămâne deschisă**. Până în prezent nu există nici un sistem de recunoaştere facială capabil să rivalizeze cu performanţele pe care le oferă în această problemă un sistem de vedere biologic, spre exemplu cel uman.
- recunoaşterea facială reprezintă o **tehnică pasivă şi nonintruzivă** pentru verificarea identităţii persoanelor. Se arată că vânzările unor astfel de sisteme depăşesc 100 mil. $ [4].
- problema recunoaşterii faciale este importantă prin prisma **aplicaţiilor** imediate ale acesteia:
  - Sisteme de supraveghere automată pentru locuri publice intens populate (aeroporturi, gări, stadioane, străzi, controlul paşapoartelor etc.) [5].
  - Controlul accesului la diferite facilităţi (clădiri, camere, computere) [6].
  - Interacţiunea om-maşină [7]
  - Găsirea bazată pe conţinut a informaţiilor conţinute în secvenţele video din bibliotecile digitale [8].

- Medicină [9].

Principalul motiv pentru care s-a ales abordarea domeniului prelucrărilor de imagini, în particular problema recunoaşterii faciale, pe baza reţelelor neuronale artificiale poate fi sintetizat printr-un citat din [10]:

"*... a one-year-old baby is much better and faster at recognizing objects, faces, and so on than even the most advanced AI system running on the fastest supercomputer.*"

Cu alte cuvinte, mamiferele în general, primatele în special, omul în final, au abilităţi recunoscute în privinţa recunoaşterii formelor. Autorul tezei consideră utilă inspirarea din modelul sistemelor biologice de vedere, bazate pe procesare neuronală, în momentul în care se doreşte realizarea unei arhitecturi pentru un sistem artificial de recunoaştere a feţelor. Faptul că mecanismul uman de recunoaştere include arii neuronale ale creierului specializate în recunoaştere facială este un lucru dovedit prin multiple cercetări asupra activităţii neuronilor [11] sau prin simptomul denumit prosopagnozie [12] care se manifestă prin aceea că la un pacient cu anumite arii cerebrale lezate, deşi sistemul de vedere funcţionează normal, subiectul nu e capabil să recunoască feţe umane. În plus, sistemele neuronale oferă şi alte avantaje descrise detalitat în următorul capitol.

În concluzie, problema recunoaşterii faciale, încă nerezolvată în totalitate, este importantă prin natura aplicaţiilor sale iar abordarea acesteia prin intermediul reţelelor neuroanle artificiale se justifică prin performanţele deosebite obţinute în aceeaşi problemă de reţelele neuronale biologice.

CAPITOLUL 3

Rețele neuronale artificiale în prelucrarea digitală a imaginilor

3.1 Avantajele folosirii rețelelor neuronale artificiale

Rețelele **n**euronale **a**rtificiale (RNA) reprezintă o încercare de a simula, cel puțin parțial, structura şi funcțiile creierului specifice organismelor vii.

Ca o definiție generală, se poate spune că RNA reprezintă un sistem de procesare al semnalelor, compus dintr-un număr mare de procesoare elementare interconectate, denumite neuroni artificiali sau noduri, şi care cooperează pentru rezolvarea unor sarcini specifice. Modalitatea de adaptare la condițiile specifice mediului constă în modificarea ponderilor asociate conexiunilor dintre neuroni şi eventual a structurii RNA.

Astfel de modele conexioniste oferă anumite **avantaje**, caracteristice sistemelor neuronale reale (biologice) şi care nu sunt întâlnite în cazul sistemelor de calcul tradiționale, secvențiale [13], [14]:

- Datorită gradului ridicat de paralelism, funcționarea defectuoasă sau chiar pierderea unui număr de neuroni nu afectează semnificativ performanța sistemului global. RNA reprezintă deci **sisteme tolerante la erori**;
- Proprietatea deosebit de importantă a RNA este aceea de a **învăța** şi de a se **adapta**;
- Posibilitatea de a opera cu **date imprecise**;
- **Capacitatea de generalizare**, în sensul în care RNA va opera corect şi cu date de intrare care nu au fost prezentate în timpul procesului de antrenament;
- **Capacitatea de a aproxima** orice funcție continuă neliniară cu gradul de acuratețe dorit. Astfel RNA pot fi folosite cu succes în modelarea sistemelor neliniare;
- Datorită numărului mare de intrări şi ieşiri, RNA modelează cu uşurință **sisteme**

multivariabilă;

- Implementările hardware ale RNA, de exemplu prin intermediul circuitelor integrate pe scară largă (VLSI), fac posibilă utilizarea RNA pentru cazul aplicaţiilor în **timp real**.

3.2 Prelucrarea digitală a imaginilor

Interesul pentru **prelucrarea digitală a imaginilor** (PDI) a fost generat de aplicaţiile practice ale acestui domeniu: îmbunătăţirea informaţiei conţinute în imagini pentru a înlesni interpretarea acestora de către om şi prelucrarea datelor unei scene necesară percepţiei artificiale.

În fig.3.2.1 sunt prezentaţi principalii paşi [15] pe care trebuie să-i parcurgă o aplicaţie PDI cu caracter general.

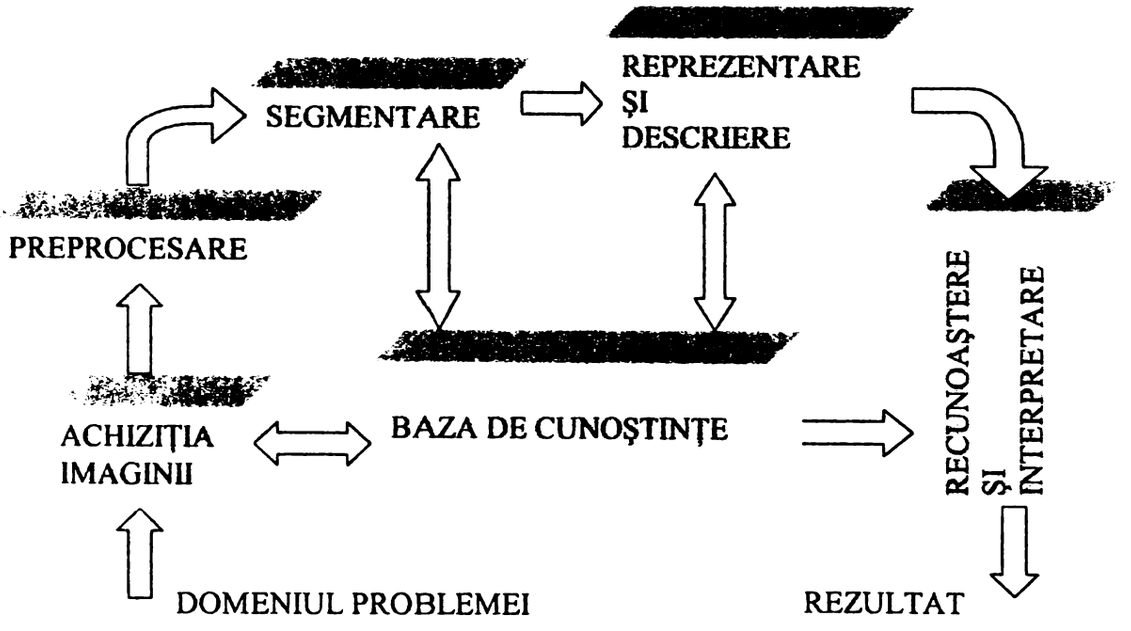


Fig.3.2.1 Etapele PDI cf. [15]

Primul pas îl reprezintă **achiziţia** imaginii sau obţinerea unei imagini digitale. Pentru aceasta este nevoie de un senzor de imagine şi de posibilitatea digitizării acestui semnal. Un astfel de exemplu îl constituie camerele TV.

După obţinerea imaginii digitale urmează cel de al doilea pas, cel al **preprocesării** imaginii. Rolul preprocesării este crucial pentru reuşita aplicaţiei, prin aceasta urmărindu-se accentuarea şi extragerea acelor informaţii care contribuie la creşterea şanselor de succes ale celorlalte procese.

Următoarea etapă este legată de **segmentarea** imaginii, adică împărţirea acesteia în părţile sale constituente sau obiecte. În general, segmentarea automată a imaginilor este una dintre operaţiile cele mai dificile. În urma segmentării se obţin date în forma unui şir de pixeli constituind conturul unei regiuni.

Reprezentarea contururilor este adecvată când se doreşte evidenţierea caracteristicilor externe ale formelor. **Reprezentarea regională** este de dorit dacă atenţia este focalizată asupra proprietăţilor interne (textură, schelet). Uneori există posibilitatea ca cele două reprezentări să coexiste.

Descrierea sau selecţia trăsăturilor urmăreşte extragerea informaţiei pentru diferenţierea unei clase sau a unui obiect.

Ultimul pas al PDI implică **recunoaşterea şi interpretarea** imaginii. Recunoaşterea este procesul prin care se atribuie o etichetă unui obiect şi se bazează pe informaţia provenită din descriere. Interpretarea presupune asigurarea unui înţeles pentru o mulţime de entităţi.

Cunoştinţele despre domeniul problemei sunt codate într-un sistem de prelucrare a imaginilor sub forma unei baze de date ce cuprinde anumite cunoştinţe, adică **baza de cunoştinţe**.

De remarcat faptul că nu toate aplicaţiile PDI implică utilizarea tuturor modulelor prezentate în fig.3.2.1.

3.3 Aplicaţii ale RNA în prelucrarea digitală a imaginilor

Datorită proprietăţilor menţionate anterior (vezi §3.1), RNA îşi găsesc o largă gamă de aplicaţii în domeniul PDI. Practic, există numeroase aplicaţii rezolvate prin intermediul RNA, pentru fiecare dintre etapele PDI (vezi fig.3.2.1). Este justificată astfel apariţia unui număr special al IEEE Trans. Image Processing destinat aplicaţiilor RNA în PDI [16]. În continuare vor fi prezentate pe scurt câteva exemple în acest sens.

3.3.1 RNA în filtrarea temporală şi spaţială

Un rol important în PDI îl are preprocesarea datelor provenite de la senzori. Ea poate contribui la reducerea zgomotului spaţial şi temporal provenit de la sistemul de achiziţie sau la îmbunătăţirea informaţiei conţinute în scenă. O serie de RNA statice sau dinamice pot fi programate să realizeze filtrări adaptive [17].

- **Filtrarea temporală**. Principala operaţie în eliminarea zgomotului prin filtrare adaptivă este interpolarea valorii x(k-n/2) a semnalului de intrare din n + 1 eşantioane prelevate dintr-un semnal continual. Cu alte cuvinte RNA primeşte la intrare o serie de valori discrete, $X(k) = [x(k), x(k-1), ..., x(k-n)]^T$ după care se estimează valoarea centrală $y(k) \approx x(k-n/2)$. Chiar şi un singur neuron poate să funcţioneze ca un filtru adaptiv calculând produsul scalar al vectorilor augmentaţi $X_a(k)$, adică vectorul corespunzător intrărilor şi $W_a(k)$ adică vectorul pondere. Elementele vectorului ponderilor sinaptice $W_a(k)$ sunt modificate după un algoritm de învăţare de tip LMS. Iniţial, filtrul este antrenat folosind eşantioane contaminate cu zgomot pentru care valorile semnalului necontaminat $y_d(k)$ sunt cunoscute.
- **Filtrarea spaţială**. Reţeaua neuronală statică descrisă anterior pentru filtrare adaptivă poate fi uşor extinsă pentru filtrare spaţială bidimensională. Intrările reţelei vor fi alimentate prin intremediul unei ferestre cu care se va baleia întreaga imagine. Numărul total de neuroni necesari implementării unei filtrări 2D va fi egal cu numărul de pixeli din imagine.

Mai recent au fost propuse [18] şi alte structuri neuronale (filtre neuronale generalizate adaptive, GANF) destinate filtrărilor neliniare.

3.3.2 RNA în codarea şi compresia imaginilor

Procesul de codare al unei imagini cu scopul de a reduce volumul datelor care se transmit printr-un canal de comunicaţie se numeşte compresia imaginii.

- **Compresia imaginilor prin RNA cu autoasociere**. Reprezintă una dintre metodele cele mai des citate în literatură şi constă în aplicarea aceleiaşi imagini la intrarea respectiv ieşirea unei RNA de tip perceptron multistrat cu un singur strat ascuns. Rata compresiei este dată de raportul dintre numărul neuronilor stratului de intrare/ieşire (egal cu numărul pixelilor din imagine) şi numărul neuronilor stratului ascuns.
- **Codarea Gabor a imaginilor prin intermediul RNA**. Daugman [19] a dezvoltat un set de funcţii bidimensionale (funcţii Gabor) care pot fi folosite pentru codarea, compresia şi analiza imaginii. Se arată că este posibil ca o imagine să se aproximeze printr-o sumă de funcţii 2D Gabor $G = [G^1, G^2, ..., G^l, ..., G^L]^T$. Fiecare $G^l \in R^{L(I \times J)}$ reprezintă o matrice bidimensională de dimensiune I x J în care elementele matricii sunt valori ale eşantioanelor spaţiale ale produsului dintre o sinusoidă 2D şi o anvelopă gaussiană. În termeni matematici, analiza imaginilor bazată pe funcţii Gabor implică determinarea vectorului $W(k)$ astfel încât:

$$Y(k) \cong X(k) = \sum_{l=1}^{L} w_l(k) G_l \qquad (3.3.2.1)$$

Una din metodele de determinare pentru aceste ponderi constă în aplicarea la intrarea unei RNA a vectorului funcţiilor Gabor iar la ieşire se va obţine imaginea codată. Ajustarea ponderilor se va face după un algoritm de învăţare tip LMS pe baza semnalului de eroare, calculat ca fiind diferenţa dintre imaginea codată şi cea originală, $X(k) - Y(k)$. Daugman a demonstrat [19] reconstrucţia aproape perfectă a unei imagini în niveluri de gri cu 8 biţi pe pixel şi cu o rată de compresia egală cu 3. Cu o rată de

compresie de 8 se obține o degradare minoră a imaginii.

3.3.3 RNA în segmentarea imaginilor

- **Segmentarea imaginilor prin RNA cu învățare de tip competitiv.** În [20] - [22] autorul tezei prezintă metode de segmentare nesupervizată a texturilor bazate pe rețele neuronale cu învățare de tip competitiv. În [21] spre exemplu, este propusă o soluție de segmentare bazată pe o structură modular-ierarhică de rețele neuronale de tip hartă de trăsături cu auto-organizare (Self Organising Feature Map, SOFM). Trăsăturile extrase din imaginea originală (omogenitate, entropie, contrast, momente etc.) nu reușesc întotdeauna să diferențieze orice pereche de texturi, dar combinația acestora poate duce la o segmentare reușită a imaginii. Se alocă pentru fiecare imagine-trăsătură o RNA-SOFM care va efectua o segmentare nesupervizată pentru aceasta. Răspunsul tuturor RNA-SOFM este prelucrat prin intermediul unei RNA-SOFM ierarhic superioară (fig.3.3.3.1), care va furniza în final imaginea complet segmentată (fig.3.3.3.2-3.3.3.6).

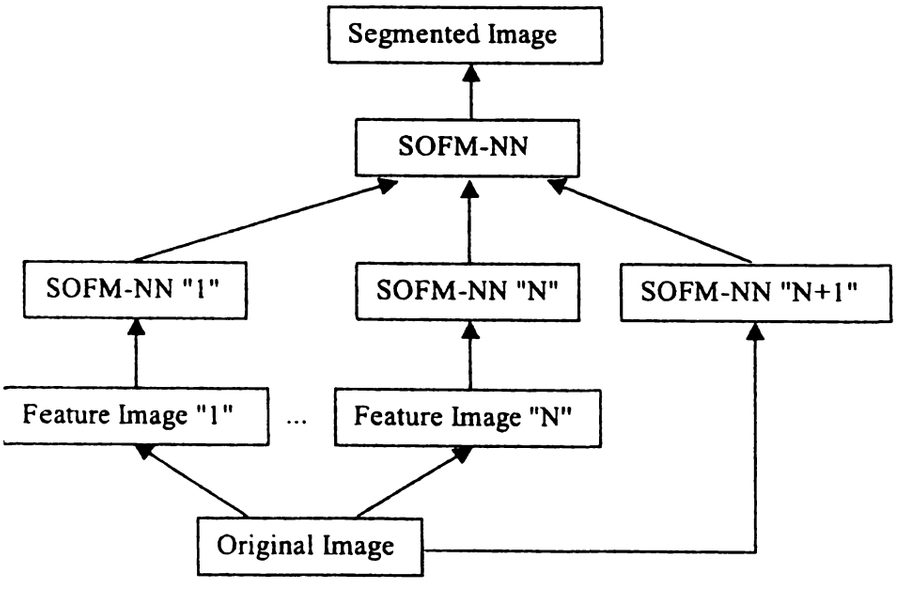
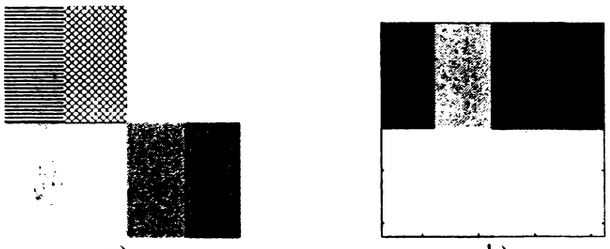
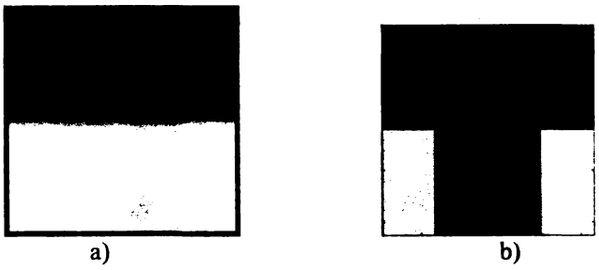


Fig.3.3.3.1 Structura unui sistem modular-ierarhic pentru segmentarea nesupervizată a texturilor propusă de autorul tezei în [22]



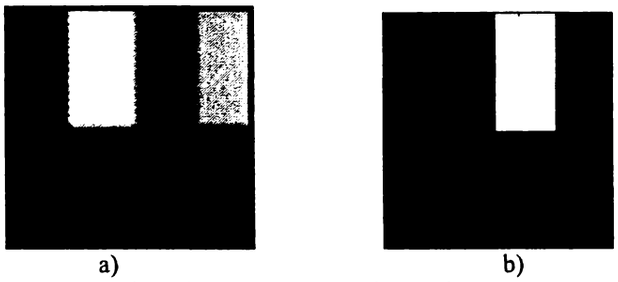
a) b)

Fig. 3.3.3.2. a) Original 8-texture image;
b) Partially unsupervised segmentation of original image



a) b)

Fig. 3.3.3.3. a) Entropy feature image;
b) Unsupervised segmentation of entropy feature image



a) b)

Fig. 3.3.3. 4 a) Contrast feature image
b) Unsupervised segmentation of contrast feature image

a) b)

Fig. 3.3.3.5 a) Second moment feature image;
b) Homogeneity feature image

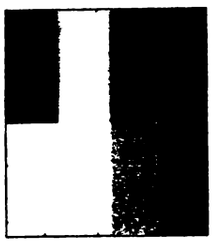


Fig. 3.3.3.6 Final segmented image (8 output neurons)

Se menționează în literatura RNA existența și a altor abordări ([23], [24]), principial diferite de cea expusă anterior, în problema segmentării imaginilor dar care presupun existența unei faze supervizate, ceea ce poate constitui un dezavantaj față de metodele propuse de autorul tezei în [20], [21] și [22].

3.3.4 RNA în reprezentarea imaginilor

Un exemplu în acest sens îl constituie reprezentarea cu RNA a contururilor imaginilor. De regulă obiectul din imagine este transformat în trăsături spațiale (mărime, orientare, formă) și trăsături contextuale, după care urmează o etapă de codare a formei; în final identificarea și localizarea obiectului. Unitatea de bază în codarea conturului este derivată, sub aspect funcțional și arhitectural, din modelul neuronilor biologici din aria vizuală V4 [25].

3.3.5 RNA în recunoașterea și interpretarea imaginilor

- **RNA pentru recunoașterea caracterelor scrise de mână**. Reprezintă una dintre aplicațiile de succes ale RNA în PDI. Un rol deosebit în acest sens îl are rețeaua neuronală dezvoltată de către Fukushima, neocognitronul [26]. Neocognitronul utilizează aceeași structură a neuronilor la fiecare nivel de procesare a informației din sistemul de vedere neuronal. Aceasta reflectă unul din deziderateleneocognitronului și anume folosirea unei structuri de calcul neuronale care să realizeze aceeași funcție în întreaga arhitectură a rețelei, dar în niveluri din ce în ce mai înalte de abstractizare.

Abordări mai recente ale aceleiaşi probleme [27] folosesc RNA convoluţionale prin care se evită complexitatea ridicată a RNA feedforward prin tehnici de partajare a ponderilor (weight sharing) şi prin folosirea conexiunilor locale. RNA folosită în [27] reprezintă o modificare a neocognitronului în sensul în care neuronii de tip "S" folosesc aici modele de tip McCulloch-Pitts.

- **RNA în recunoaşterea automată a ţintelor**. Se pot deosebi două stadii esenţiale în aplicaţiile care implică recunoaşterea obiectelor de către RNA: extragerea trăsăturilor şi clasificarea propriu-zisă. Prima etapă presupune izolarea informaţiei esenţiale din imagine şi implicit o reducere a dimensionalităţii vectorilor ce urmează a fi clasificaţi ulterior. Ea este implementată prin metode statistice (PCA, SVD, filtre Gabor, metoda operatorului de interes etc.) sau metode bazate pe RNA (SOFM, GHA). Clasificarea este de regulă efectuată prin intermediul RNA-MLP (perceptron multistrat) sau RNA-RBF (RNA bazate pe funcţii radiale). Un exemplu ilustrativ pentru principiile enunţate mai sus este prezentat în [28].

3.4 Concluzii

Întrucât detecţia şi recunoaşterea facială presupun operaţii specifice PDI (filtrare, extragere de trăsături, clasificare) autorul tezei investighează soluţii [29] de rezolvare ale acestor probleme folosind RNA.

Una dintre concluziile importante este aceea că există un înalt grad de similitudine între noţiunile vehiculate în teoria RNA şi cea a metodelor clasice, statistice [30], [31] fapt ilustrat şi de tabelul 3.4.1.

În ceea ce priveşte performanţele acestor abordări distincte există deja publicate numeroase studii [32], [33], [34]. În problema clasificării spre exemplu, deosebit de importantă pentru o aplicaţie de recunoaştere facială, sunt comparaţi, în [35], 16 algoritmi de clasificare statistici şi neuronali asupra a două probleme diferite: clasificarea cifrelor scrise de mână şi clasificarea fonemelor. Rezultatele sunt prezentate rezumativ în tab.3.4.2.

Rețele neuronale artificiale	Metode clasice (statistice)
Învățare (learning)	Estimare (estimation)
Ponderi (weight)	Parametri (parameters)
Învățare supervizată (supervised learning)	Regresie (regression)
Clasificare (classification)	Discriminare (discrimination)
Învățare nesupervizată (non supervised learning)	Estimarea densității (density estimation)
Grupare (clustering)	Clasificare (classification)
Rețea neuronală	Model

Tab.3.4.1 Paralelism terminologic între metodele bazate pe RNA şi cele clasice (statistice) [31].

Clasificator	Eroare probl.1 [%]	Eroare probl.2 [%]
LDA (linear discriminant analysis)	9,8	24,4
QDA (quadratic discriminant analysis)	3,7	21,7
RDA (regularized discriminant analysis)	3,4	21,8
KDA (kernel discriminant analysis)	3,7	11,3
RKDA (reduced kernel discriminant analysis)	5,2	18,0
MLP (multilayer perceptron)	5,4	13,9
MLP+WD (multilayer perceptron+weight decay)	3,5	13,6
LLR (local linear regression)	2,8	12,9
FDA/MARS (multivariate adaptive regression splines)	6,3	14,0
1-NN (1 nearest neighbor)	4,2	11,7
3-NN (3 nearest neighbor)	3,8	12,5
L-3-NN (learning k-NN)	3,6	14,1
LVQ (learning vector quantization)	4,0	12,7

Tab.3.4.2 Performanțele comparative pentru clasificatori statistici şi neuronali în problemele: clasificarea cifrelor scrise de mână (probl.1) şi clasificarea fonemelor (probl.2) cf. [35].

Concluziile care se desprind în urma analizei performanţelor acestor algoritmi arată că nu există o abordare care să domine, prin prisma performanţelor şi că aceste performanţe depind de natura concretă a aplicaţiei [36].

Se poate concluziona faptul că datorită proprietăţilor ce caracterizează RNA (grad înalt de paralelism, sisteme tolerante la erori, capacitate de a învăţa, generaliza, opera cu date imprecise etc.) acestea se constituie într-o soluţie viabilă pentru domeniul PDI.

În plus această concluzie este argumentată şi prin prezentarea modului de soluţionare a unor probleme caracteristice PDI (filtrare, codare, compresie, segmentare, reprezentare, recunoaştere şi interpretare) prin intermediul RNA.

CAPITOLUL 4

Propuneri privind algoritmi rapizi de antrenament pentru rețele neuronale cu propagare înainte (feedforward)

4.1 Elemente de neurodinamică

În continuare sunt prezentate aspecte legate de calculul neuronal şi propunerile autorului tezei în legătură cu posibilitățile de îmbunătățire ale algoritmilor de antrenament pentru RNA feedforward.

4.1.1 Neuronul artificial

Neuronul artificial denumit uneori procesor elementar sau mai simplu nod, încearcă să imite structura şi funcționarea neuronului biologic. Există numeroase modele prezentate în literatură [37], dar cel mai răspândit are la bază modelul elaborat de McCulloch-Pitts în 1943. Astfel se poate considera că neuronul artificial [38] este format dintr-un **număr de intrări**, fiecare dintre acestea fiind caracterizată de propria **pondere sinaptică**. De exemplu, semnalul x_j prezent la intrarea sinapsei "j" este conectat la neuronul "k" prin multiplicare cu ponderea w_{kj} (fig.4.1.1.1).

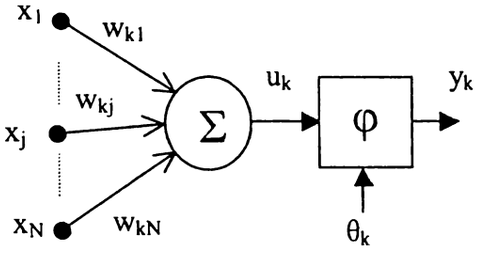


Fig. 4.1.1.1 Modelul neuronului artificial, cf. [38]

O altă componentă a modelului neuronului artificial prezentat în fig.4.1.1.1 o reprezintă **sumatorul** destinat însumării intrărilor ponderate.

Rezultatul obținut în urma însumării:

$$u_k = \sum_{j=1}^{N} w_{kj} \cdot x_j \qquad (4.1.1.1)$$

este denumit intrare netă.

Pentru limitarea amplitudinii semnalului de ieşire al neuronului, acesta este prevăzut cu o **funcție de activare** $\varphi(\cdot)$:

$$y_k = \varphi(u_k - \theta_k) \qquad (4.1.1.2)$$

în care θ_k reprezintă valoarea **pragului de activare** (treshold) al neuronului. Uneori intrarea netă este majorată prin termenul b_k denumit factor al deplasării scării (bias); deplasarea scării reprezintă deci negativul pragului de activare.

Valoarea:

$$v_k = u_k - \theta_k \qquad (4.1.1.3)$$

poartă denumirea de **potențial de activare**.

În ceea ce priveşte tipul funcției de activare, aceasta este de regulă o funcție neliniară (fig.4.1.1.2).

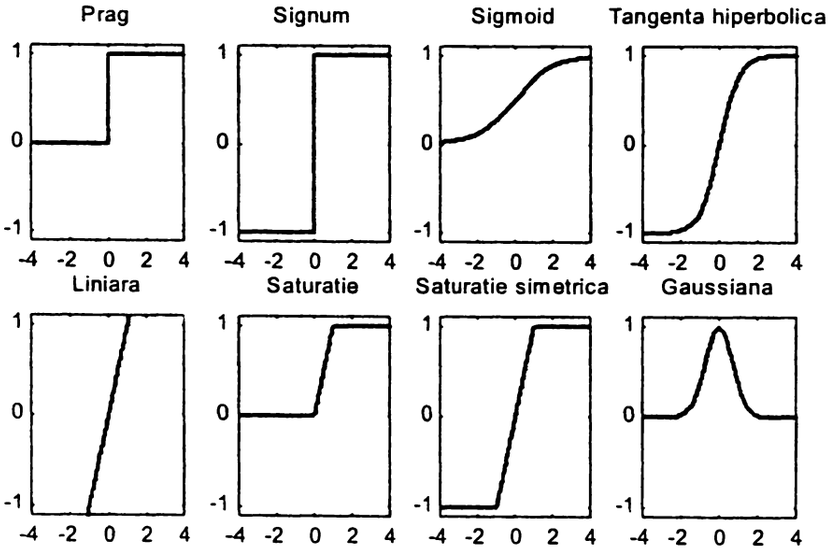


Fig. 4.1.1.2 Funcții de activare tipice, prezentate de autorul tezei în [39]

Analizând comparativ modelele neuronului real (biologic) şi neuronului artificial se pot face următoarele observaţii [40]:

- Din punct de vedere al implementării este practic imposibil şi chiar ineficient ca modelul artificial al neuronului să copieze exact comportamentul şi structura celui biologic.
- RNA sunt proiectate pentru rezolvarea unor probleme specifice şi deci arhitectura şi trăsăturile RNA depind de problema pe care trebuie să o rezolve.
- Un neuron real produce la ieşire o secvenţă de impulsuri şi nu o anumită valoare cum este cazul celui artificial. Reprezentarea ratei de activare printr-un singur număr (y_k) ignoră informaţia care ar putea fi conţinută de exemplu în faza impulsurilor.
- Unele celule nervoase biologice efectuează o însumare neliniară a intrărilor. Pot exista chiar operaţii logice (ŞI, SAU, NU) efectuate la nivelul dendritelor.
- Ieşirile neuronilor nu se modifică în mod sincron şi nu toţi au acelaşi tip de întârziere.
- Cantitatea de substanţă transmiţătoare (mediator chimic) eliberată la nivelul sinapsei poate să varieze într-un mod imprevizibil. Fenomenul este aproximat grosier prin intermediul funcţiei de activare.

4.1.2 Arhitecturi ale RNA

Se pot distinge două mari categorii în modul de structurare al unei RNA:

- **RNA feedforward** (cu propagare înainte).

Sunt caracterizate de prezenţa unui strat de neuroni de intrare, un număr de straturi ascunse (posibil şi fără) şi un strat de neuroni de ieşire.

În fig.4.1.2.1 este prezentată arhitectura unei RNA feedforward cu un singur strat ascuns. Definitoriu pentru acest tip de RNA este faptul că un neuron primeşte

semnale doar de la neuroni aflați în stratul/straturi precedent/precedente. Se spune despre o RNA că este **total conectată** dacă fiecare nod din fiecare strat este conectat la fiecare neuron din stratul precedent (fig.4.1.2.1).

Dacă anumite conexiuni sinaptice lipsesc, se spune că RNA este **parțial conectată** (fig.4.1.2.2). RNA total conectate au un caracter general, în sensul în care pot fi folosite într-o gamă largă de probleme, dar rezultatele nu sunt întotdeauna cele mai bune [41]. RNA parțial conectate introduc anumite restrângeri, care reprezintă tocmai cunoștințe apriorice despre problema de rezolvat şi care reduc gradul de generalitate al unei RNA. Prin restrângerea câmpului de recepție al neuronilor se efectuează o extragere a trăsăturilor locale iar în straturile ce urmează acestea sunt combinate pentru a se forma trăsături de ordin superior. Astfel RNA parțial conectate pot da rezultate mai bune decât RNA total conectate în rezolvarea anumitor probleme specifice, cu condiția exploatării cunoștințelor apriorice despre problema dată.

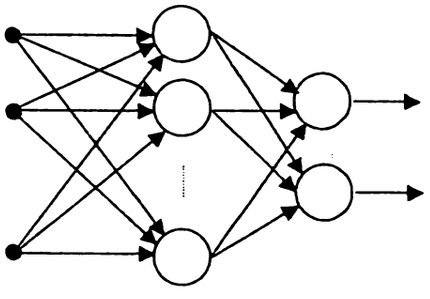


Fig. 4.1.2.1 RNA feedforward total conectată, prezentată de autorul tezei în [39].

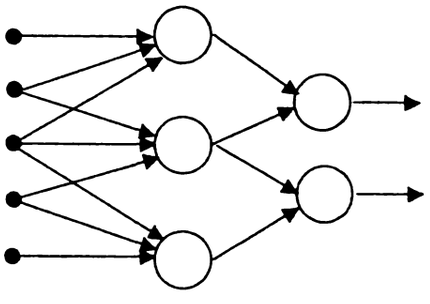


Fig.4.1.2.2 RNA feedforward parțial conectată, prezentată de autorul tezei în [39].

- **RNA recurente** (feedback, cu propagare înapoi).

RNA recurente se individualizează prin existența unui semnal de reacție, din partea neuronilor de ordin superior, pentru cei de ordin inferior sau chiar pentru propriile lor intrări (fig.4.1.2.3).

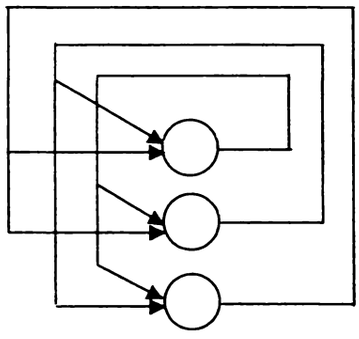


Fig.4.1.2.3 RNA recurentă, prezentată de autorul tezei în [39].

4.1.3 Tipuri şi algoritmi de instruire

RNA achiziționează cunoştințele prin instruire (învățare). Învățarea presupune adaptarea parametrilor liberi ai RNA ca urmare a stimulilor mediului în care se găseşte rețeaua.

Vectorii de instruire sunt prezentați RNA în mod secvențial iar ponderile sinaptice, care memorează practic cunoştințele rețelei, sunt adaptate pentru a extrage informația pe care aceşti vectori o conțin.

Tipul de învățăre este determinat de maniera în care sunt ajustați parametrii liberi ai RNA.

În literatura RNA [38], [42], [43] există o mare diversitate de opinii în ceea ce priveşte modul de clasificare al algoritmilor şi tipurilor de învățăre; fig.4.1.3.1 prezintă opinia autorului tezei, prezentată în [13], în ceea ce priveşte această clasificare.

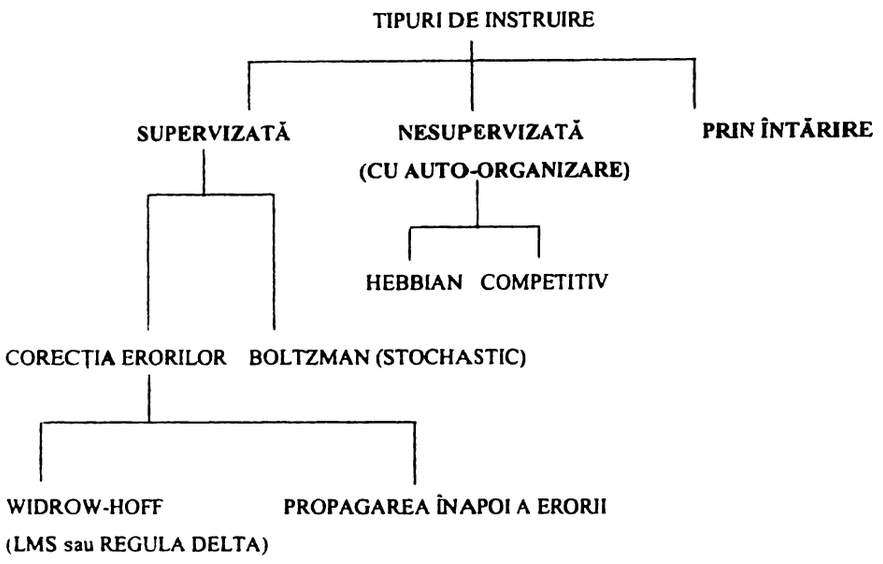


Fig.4.1.3.1 Tipuri şi algoritmi de instruire, cf. opiniei autorului tezei [13].

Descrieri detaliate ale paradigmelor învățării şi ale tipurilor de algoritmi de învățare pot fi găsite în [13], [37] sau [38].

4.2 Învățarea RNA văzută ca o strategie de optimizare

În continuare vor fi prezentați algoritmii de învățare folosiți pentru instruirea RNA-MLP implicate în experimentele din cadrul acestei lucrări, pe baza celor propuse de către autorul tezei [44], autorul tezei şi Petropoulakis [45] şi Moller în [46].

Necesitatea unor astfel de algoritmi de antrenament sofisticați rezidă din complexitatea problemelor aferente recunoaşterii faciale. Mai precis, manipularea unor vectori de intrare de dimensiuni mari ($nx10^2$...$nx10^3$ componente) necesită algoritmi care să fie capabili să conveargă într-un timp cât mai scurt şi cu cerințe de memorie rezonabile.

Este recunoscut faptul că algoritmii bazați pe metoda gradientului descendent prezintă o rată de convergență slabă şi depind de anumiți parametri care trebuie

specificați de utilizator, neexistând o bază teoretică pentru alegerea acestora. Un exemplu în acest sens îl constituie algoritmul cu propagare înapoi a erorii (BP, backpropagation) în versiunile standard sau cu moment.

Cele mai multe metode folosite pentru minimizarea funcțiilor sunt bazate pe aceeași strategie:

1) Alegerea unui vector pondere inițial $\mathbf{w}_1$ și k=1.

2) Determinarea unei direcții de căutare $\mathbf{p}_k$ și a unui pas α_k astfel încât funcția eroare să îndeplinească următoarea inegalitate:

$E(\mathbf{w}_k + \alpha_k \mathbf{p}_k) < E(\mathbf{w}_k)$

3) Modificarea vectorului pondere:

$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \mathbf{p}_k$

4) Dacă $E'(\mathbf{w}_k) \neq 0$, k = k+1 și salt la 2; altfel întoarce $\mathbf{w}_{k+1}$ ca fiind vectorul pentru care se obține minimul dorit.

Se constată că determinarea punctului curent următor reprezintă un proces iterativ care implică două faze: în primul rând găsirea unei ***direcții*** în spațiul ponderilor de căutare iar în al doilea rând găsirea unui ***pas*** care să specifice cât este de mare deplasarea după direcția găsită anterior.

4.2.1 Accelerarea vitezei de convergență a algoritmului BP prin reglarea fuzzy a ratei de învățare

Dacă direcția de căutare $\mathbf{p}_k$ menționată mai sus se consideră a fi negativul gradientului funcției de cost, adică $-E'(\mathbf{w})$ și α_k = ct., se obține algoritmul gradientului descendent. Se observă că minimizarea prin metoda gradientului descendent se bazează pe aproximarea liniară $E(\mathbf{x} + \mathbf{y}) \approx E(\mathbf{w}) + E'(\mathbf{w})^T\mathbf{y}$.

Autorul tezei [44] și autorul tezei și Petropoulakis [45] propun o metodă bazată pe reglarea fuzzy a ratei de învățare, în scopul eliminării dezavantajelor mai sus menționate. Tiponuț și autorul tezei [13] descriu în detaliu principiile și modul concret

de implementare al acestei metode:

Este cunoscut faptul că rata de învățare η trebuie să fie suficient de mică, de regulă subunitară, pentru a preveni apariția oscilațiilor și a asigura convergența algoritmului de antrenament. Însă o rată prea mică de învățare poate încetini considerabil procesul de învățare al unei RNA. Timpul de antrenament poate fi considerabil redus prin folosirea unei rate de învățare adaptive (variabile) care să se mențină la o valoare cât mai ridicată, dar care să nu ducă la un proces de învățare instabil. Cvasitotalitatea metodelor fuzzy [47], [48], [49] sau clasice [50] de reglare a ratei de învățare se bazează pe analiza valorii absolute a erorii sau a gradientului din epoca curentă și eventual precedentă. Se propune în cele ce urmează o metodă fuzzy de adaptare a η, bazată pe valoarea relativă a criteriului de performanță specific unei RNA date. O astfel de abordare este justificată atât în cazul metodelor de adaptare clasice dar mai ales în cazul metodelor fuzzy deoarece valoarea numerică a erorii este dependentă de tipul funcției de activare a neuronilor. Spre exemplu, considerând o aceeași arhitectură a unei RNA, în cazul unei funcții de activare a neuronilor de tip liniar, gama de variație a erorii poate fi de 10^3 ori mai mare decât în cazul unei funcții de activare de tip sigmoidal. Astfel, considerarea valorii numerice absolute a criteriului de performanță (de regulă eroarea pătratică medie) drept mărime de intrare a unui controler fuzzy va presupune implicit introducerea unor limitări care afectează în mod negativ performanțele algoritmului de reglare a ratei de învățare.

În concluzie, se propune folosirea drept mărimi de intrare a controlerului fuzzy a ***valorii relative a criteriului de performanță*** ales:

$$p(n) = \frac{perf(n)}{perf(n-1)} \tag{4.2.1.1}$$

și a ***valoarii variației relative a criteriului de performanță***:

$$\Delta p(n) = \frac{perf(n)}{perf(n-1)} - \frac{perf(n-1)}{perf(n-2)} \tag{4.2.1.2}$$

Prin urmare cele două variabile lingvistice enunțate mai sus vor reprezenta mărimile de intrare ale controlerului fuzzy iar mărimea de ieșire va reprezenta un

coeficient "c" de modificare a ratei de învățare:

$$\eta(n) = c \cdot \eta(n-1) \tag{4.2.1.3}$$

Arhitectura acestui sistem este prezentată în fig.4.2.1.1.

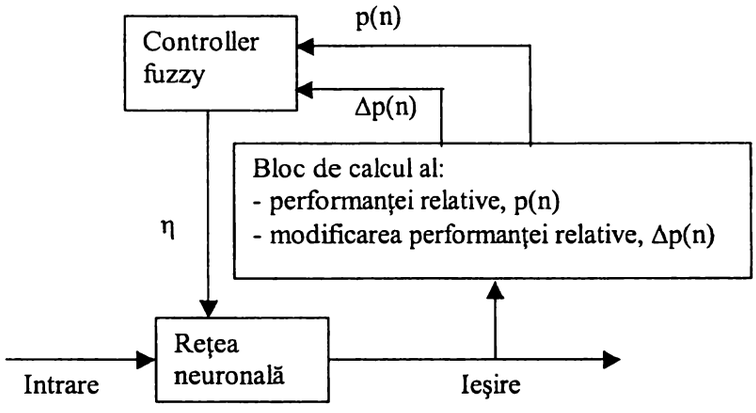


Fig.4.2.1.1 Sistem fuzzy pentru adaptarea ratei de învățare a unei RNA feedforward

Algoritmul de reglare a ratei de învățare funcționează conform bazei de reguli prezentate în tab.4.2.1.1 şi este inspirat din metoda clasică de modificare a ratei de învățare prezentată de către Vogl şi Mangis în [51]. Pentru claritatea expunerii se prezintă succint acestă metodă, care se găseşte implementat şi în mediul MATLAB v.5.3, Neural Network Toolbox v.3.0., sub forma funcției "*traingdx*":

$$\eta(k) = \begin{cases} c_1 \cdot \eta(k-1), \; daca \; perf(w(k)) < perf(w(k-1) \\ c_2 \cdot \eta(k-1), \; daca \; perf(w(k)) \geq c_3 \cdot perf(w(k-1) \\ \eta(k-1), \; altfel \end{cases} \tag{4.2.1.4}$$

Valorile tipice pentru aceşti coeficienți sunt: $c_1 = 1{,}05$, $c_2 = 0.7$, $c_3 = 1{,}04$.

Dezavantajul algoritmului mai sus menționat constă în valoarea coeficienților c_1 şi c_2 care sunt menținuți constanți pe tot parcursul procesului de antrenamet, împiedicând variația rapidă a ratei de învățare. Algoritmul fuzzy propus de către autorul tezei nuanțează acești coeficienți în funcție de valorile p(n) şi Δp(n) şi oferă o viteză de convergență sporită.

Pentru implementarea controlerului fuzzy se vor folosi facilitățile oferite de toolboxul MATLAB 5.3 specializat în dezvoltarea sistemelor fuzzy, adică Fuzzy Logic Toolbox v.2.0.1 Comanda MATLAB "fuzzy" apelează editorul FIS (Fuzzy Inference System editor) care, prin intermediul interfeței grafice utilizator (vezi fig.4.2.1.2) permite specificarea interactivă a:

- arhitecturii controlerului (număr de intrări şi ieşiri);
- tipului controlerului (Mamdami sau Sugeno);
- funcțiilor de apartenență;
- bazei de reguli;
- metodei de defuzzyficare;
- vizualizarii suprafaței de control etc.

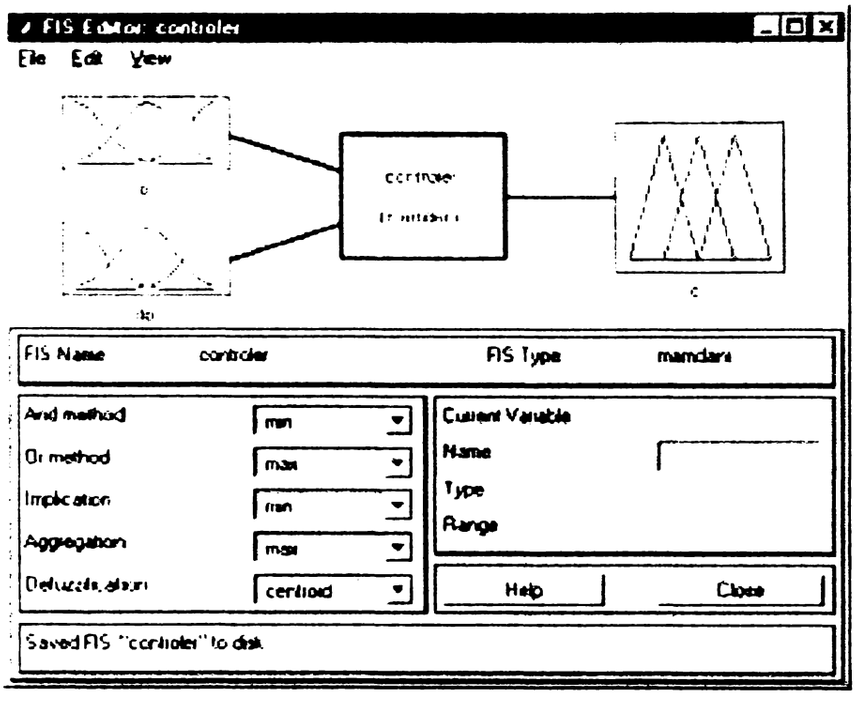
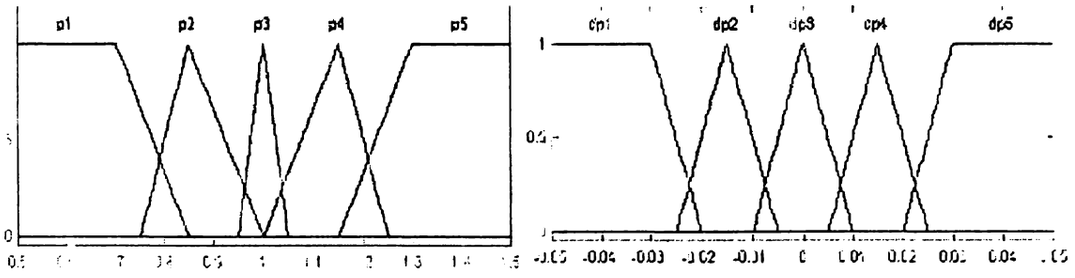


Fig.4.2.1.2 Arhitectura controlerului fuzzy de tip Mamdami pentru modificarea adaptivă a ratei de învățare, prezentată de autorul tezei în [13].

Funcţiile de apartenenţă pentru cele două variabile lingvistice de intrare sunt prezentate în fig.4.2.1.3 iar valorile lingvistice pentru variabila lingvistică asociată ieşirii se pot urmări în fig.4.2.1.4. Suprafaţa de reglare a controlerului fuzzy este înfăţisată în fig.4.2.1.5.



a) p(n) b) Δp(n)

Fig. 4.2.1.3 Funcţiile de apartenenţă pentru valoare relativă a criteriului de performanţă p(n) şi variaţie a valoarii relative a criteriului de performanţă Δp(n), prezentate de autorul tezei în [45].

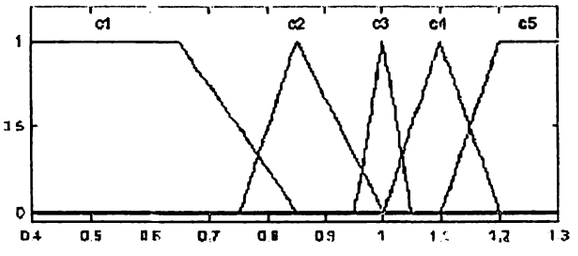


Fig. 4.2.1.4 Funcţiile de apartenenţă pentru variabila lingvistică "c", prezentate de autorul tezei în [45].

dp \ p	p1 (caz cel mai bun)	p2	p3	p4	p5 (caz cel mai rău)
dp1 (caz cel mai bun)	c5	c5	c4	c3	c2
dp2	c5	c4	c4	c2	c2
dp3	c4	c4	c3	c2	c1
dp4	c3	c2	c2	c2	c1
dp5 (caz cel mai rău)	c2	c2	c1	c1	c1

Tab.4.2.1.1 Baza de reguli a controlerului fuzzy, prezentată de autorul tezei în [45].

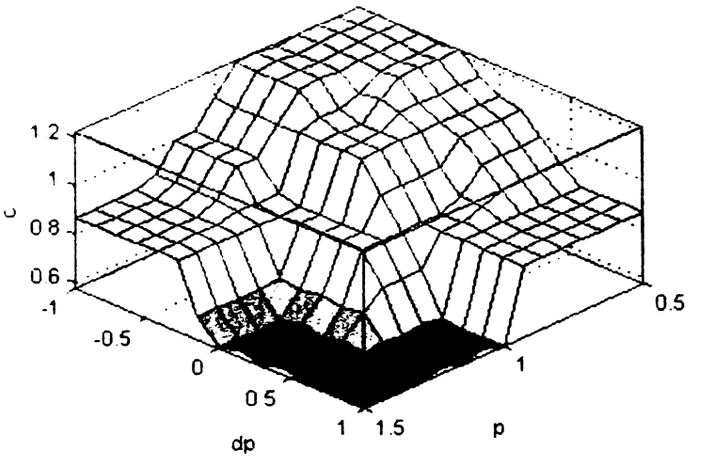


Fig. 4.2.1.5 Suprafaţa de reglare a controlerului fuzzy, cf. autorului tezei [13].

Odată definit controlerul fuzzy, va trebui scrisă o funcţie de antrenament a reţelei neuronale denumită "*trainfuzzy*" care va realiza interfaţarea algoritmului gradientului descendent cu algoritmul de reglare adaptivă a ratei de învăţare. Totodată acestă funcţie trebuie să respecte convenţiile MATLAB, ea fiiind practic o "metodă" specifică unui obiect de tip "net". În ANEXA 1 este redat codul sursă al noii funcţii de antrenament. Pentru evaluarea performanţelor algoritmului de antrenament descris anterior s-au considerat următoarele tipuri de probleme:

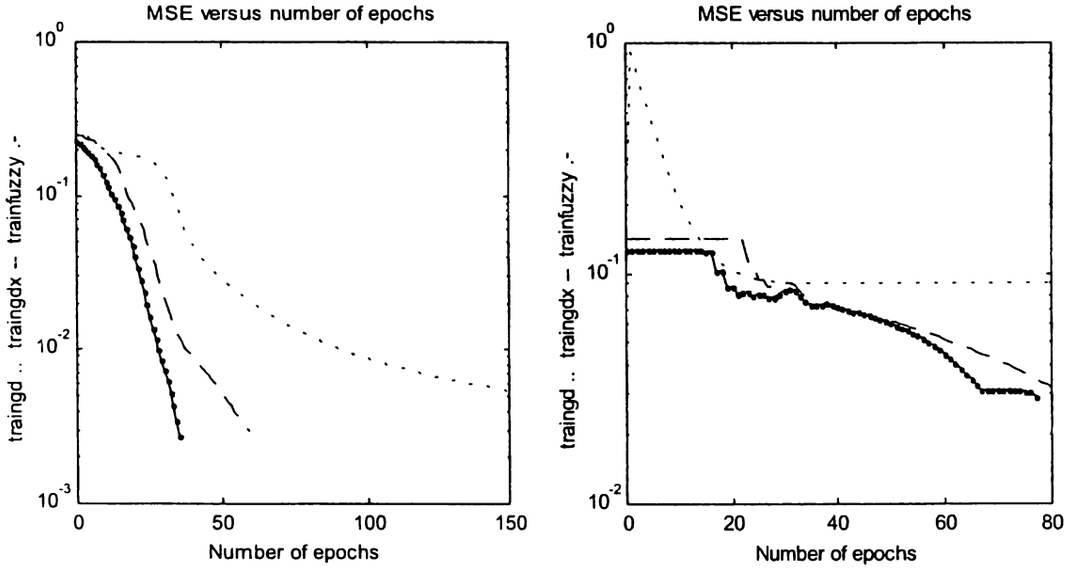
- ***Problema sonarului, Mine versus Stânci***; Acest set de date a fost utilizat de către Gorman şi Sejnowski [52] în studierea semnalelor emise de sonar, folosind reţele neuronale. Problema constă în discriminarea între ecoul emis de către cilindri de metal şi ecoul emis de către stânci. Setul de date este în formatul standard CMU (Carnegie Mellon University) [53], şi este intens folosit în literatura de specialitate pentru compararea performanţelor algoritmilor de antrenament ai RNA. Există 104 tipare de antrenament şi 104 tipare de test formate din vectori de intrare continuali 60 – dimensionali şi vectori de ieşire unidimensionali de tip enumerativ.

- ***Problema aproximării unei funcții***; Reprezintă un test des utilizat în compararea performanțelor RNA [54] şi îşi propune construirea unei rețele care să aproximeze funcția:

$$f(x) = 0{,}2 + 0{,}8 \cdot (x + 0{,}7 \cdot \sin(2 \cdot \pi \cdot x))$$

presupunând $0 \leq x \leq 1$. Datele de antrenament sunt luate cu un pas de 0,1 deci există 11 puncte de antrenament. Se utilizează 101 date de test, considerate deci în intervale de 0,01. Datele de test vor fi folosite la verificarea capacității de interpolare a RNA.

- ***Problema recunoaşterii tiparelor***; Problema constă în recunoaşterea a 100 de imagini faciale (10 subiecți x 10 imagini/persoană). Setul de date este împărțit în 50 de imagini de antrenament şi 50 de imagini de test alese din baza de date AT&T (fosta ORL) [55]. Imaginile sunt în format 56x46 pixeli deci dimensiunea RNA-MLP va fi: 2576 neuroni în stratul de intrare, 50 neuroni ascunşi şi 10 neuroni de ieşire



a) Problema aproximării unei funcții b) Problema clasificării tiparelor

Fig. 4.2.1.6 Rezultate experimentale obținute de către autorul tezei [44].

Evoluția erorii medii pătratice de-a lungul epocilor de antrenament indică drept cea mai bună metodă funcția de antrenament trainfuzzy, atât pentru cazul aproximării funcțiilor cât şi pentru cazul recunoaşterii tiparelor

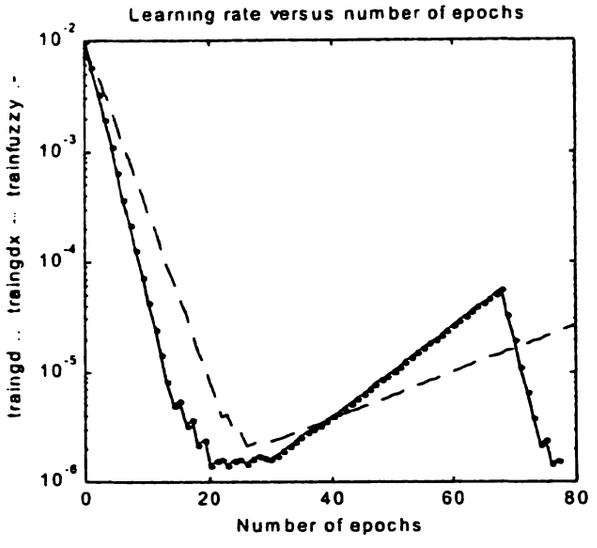


Fig. 4.2.1.7 Variaţia ratei de învăţare în cazul metodei fuzzy este mai rapidă decât versiunea clasică, ceea ce justifică şi performanţele mai bune ale funcţiei "trainfuzzy" faţă de "traingdx", cf. autorului tezei [44].

Metodă de antrenament	# unităţi ascunse	# iteraţii	Timp de antrenament [s]	Rată de antrenament iniţială	Eroare set antrenament [%]	Eroare set test [%]
Problema sonarului (Mines versus Rocks)						
Trainlm	20	-	-	0.01	-	-
Traingdm	20	800	29.34	0.01 constant	11.5385	16.3462
Traingdx	20	800	27.85	0.01	2.8846	11.5385
Trainfuzzy	20	800	26.61	0.01	0.9615	5.6732
Aproximarea unei funcţii						
Trainlm	9	12	1.26	0.65	2.2501	1.4703
Traingdm	9	150	18.89	0.65 constant	3.5603	2.3570
Traingdx	9	59	5.50	0.65	2.7846	0.9310
Trainfuzzy	9	34	3.91	0.65	2.4817	0.0091
Problema recunoaşterii tiparelor faciale						
Trainlm	50	80	-	0.01	-	-
Traingdm	50	80	105.73	0.01 constant	90	90
Traingdx	50	80	90.63	0.01	4	10
Trainfuzzy	50	76	84.53	0.01	2	4

Tab.4.2.1.2 Rezultate comparative asupra a 3 tipuri de probleme: problema sonarului, aproximarea funcţiilor şi recunoaştere facială, cf. autorului tezei [44], [45]

trainlm = Metoda Levenberg-Marquardt;

traingdm = Metoda gradient descendent cu moment;

traingdx = Metoda cu rata de învăţare variabilă (versiunea clasică);

trainfuzzy = Metoda cu rata de învăţare variabilă (versiunea fuzzy)

Rezultatele experimentale (tab.4.2.1.2, fig.4.2.1.6, fig. 4.2.1.7) indică faptul că această metodă oferă un raport foarte bun ***complexitate algoritm / viteză de convergență***, fiind deosebit de util atunci când RNA primeşte la intrare ***vectori de dimensiuni mari ($nx10^2$...$nx10^3$)***, adică tocmai cazul întâlnit în recunoaşterea facială.

Sursele MATLAB aferente realizării acestor experimente se regăsesc în ANEXA 2.

4.2.2 Accelerarea vitezei de convergență pentru algoritmi bazați pe metoda gradientului conjugat

Metodele bazate pe direcția conjugată se înscriu în metodologia generală a strategiei de optimizare prezentată anterior, dar direcția de căutare şi pasul sunt modificate ținând cont de aproximația de ordinul II:

$$E(\mathbf{w}+\mathbf{y}) = E(\mathbf{w}) + E(\mathbf{w})^T\mathbf{y} + \tfrac{1}{2}\mathbf{y}^T E''(\mathbf{w})\mathbf{y} \qquad (4.2.2.1)$$

Pentru determinarea minimului funcției de cost, se anulează derivata ecuației precedente şi se obține:

$$E'_{qw}(\mathbf{y}) = E''(\mathbf{w})\mathbf{y} + E'(\mathbf{w}) = 0 \qquad (4.2.2.2)$$

Fie $\mathbf{p}_1, \ldots, \mathbf{p}_k$ o mulțime de vectori pondere în spațiul R^N. Mulțimea reprezintă un ***sistem conjugat*** în raport cu matricea nesingulară NxN, A, dacă, cf. Hestenes [56]:

$$\mathbf{p}_i^T A\mathbf{p}_j = 0 \quad (i \neq j, i = 1,\ldots,k) \qquad (4.2.2.3)$$

Dacă există un astfel de sistem conjugat, Johansson, Dowla şi Goodman [57] arată că:

$$\mathbf{y}_* - \mathbf{y}_1 = \sum_{i=1}^{N} \alpha_i \mathbf{p}_i \qquad (4.2.2.4)$$

în care $\mathbf{y}_1$ reprezintă punctul de start şi $\mathbf{y}_*$ punctul critic.

Înmulțind ec. (4.2.2.4) cu $\mathbf{p}_j^T E''(\mathbf{w})$ se obține:

$$\mathbf{p}_j^T(-E'(\mathbf{w}) - E''(\mathbf{w})\mathbf{y}_1) = \alpha_j \mathbf{p}_j^T E''(\mathbf{w})\mathbf{p}_j$$

$$\Rightarrow \quad \alpha_j = \mathbf{p}_j^T(-E'(\mathbf{w}) - E''(\mathbf{w})\mathbf{y}_1)/\mathbf{p}_j^T E''(\mathbf{w})\mathbf{p}_j = -\mathbf{p}_j^T E'_{qw}(\mathbf{y}_1)/\mathbf{p}_j^T E''(\mathbf{w})\mathbf{p}_j \qquad (4.2.2.5)$$

Punctul critic y^* poate fi determinat în N paşi iterativi folosind ecuaţiile (4.2.2.4), (4.2.2.5).

***Observaţie**: y^* nu este neapărat un minim (cu excepţia cazului când hessianul $E''(\mathbf{w})$ este pozitiv definit).*

Punctele intermediare $\mathbf{y}_{k+1} = \mathbf{y}_k + \alpha_k \mathbf{p}_k$ generate pentru determinarea iterativă a lui y^* reprezintă minime ale $E_{qw}(\mathbf{y})$ restricţionate la planul π_k: $\mathbf{y} = \mathbf{y}_1 + \alpha_1 \mathbf{p}_1 + \ldots + \alpha_k \mathbf{p}_k$. Cum se determină aceste puncte recursiv se arată în următoarea teoremă (Hestenes [56]):

TEOREMA 1. *Fie $p_1, \ldots, p_N$ un sistem conjugat şi y_1 un punct în spaţiul ponderilor. Fie punctele $y_2, \ldots, y_{N+1}$ definite recursiv de:*

$$y_{k+1} = y_k + \alpha_k p_k$$

în care $\alpha_k = \mu_k / \delta_k$, $\mu_k = -\mathbf{p}_j^T E'_{qw}(\mathbf{y}_k)$, $\delta_k = \mathbf{p}_k^T E''(\mathbf{w}) \mathbf{p}_k$

În acest caz y_{k+1} minimizează E_{qw} în planul π_k.

Algoritmul direcţiei conjugate presupune existenţa unui sistem conjugat. Dar, în realitate nu este necesară cunoaşterea apriorică a vectorilor $\mathbf{p}_1, \ldots, \mathbf{p}_N$ pentru că ei pot fi determinaţi recursiv, cf. (Hestenes [56]):

TEOREMA 2. *Fie y_1 un punct în spaţiul ponderilor şi p_1 şi r_1 egali cu valoarea gradientului descendent – $E'_{qw}(y_1)$. Se defineşte p_{k+1} recursiv prin:*

$$p_{k+1} = r_{k+1} + \beta_k p_k$$

în care $r_{k+1} = E'_{qw}(y_{k+1})$, $\beta_k = (|r_{k+1}|^2) - r^T_{k+1} r_k) / p^T_k r_k$ *şi y_{k+1} reprezintă punctul dat de TEOREMA 1. Atunci p_{k+1} reprezintă vectorul descendent pentru E_{qw} în planul π_{N-k}.*

În aceste condiţii se poate trece la prezentarea **algoritmului gradient conjugat standard**:

1. Se alege vectorul pondere iniţial $\mathbf{w}_1$.

   $\mathbf{p}_1 = \mathbf{r}_1 = -E'(\mathbf{w}_1)$, $k = 1$.

2. Se calculează informaţia de ordinul II:

$s_k = E''(\mathbf{w})\mathbf{p}_k$

$\delta_k = \mathbf{p}^T_k s_k$

3. Se calculează mărimea pasului de căutare:

$\mu_k = \mathbf{p}^T_k \mathbf{r}_k$

$\alpha_k = \mu_k / \delta_k$

4. Modificarea vectorului pondere:

$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \mathbf{p}_k$

$\mathbf{r}_{k+1} = E'_{qw}(\mathbf{w}_{k+1})$

5. Dacă k mod N = 0 repornește algoritmul: $\mathbf{p}_{k+1} = \mathbf{r}_{k+1}$

Altfel creează o nouă direcție conjugată:

$\beta_k = (|\mathbf{r}_{k+1}|^2 - \mathbf{r}^T_{k+1}\mathbf{r}_k) / \mu_k$

$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$

6. Dacă direcția descendentă $\mathbf{r}_k \neq \mathbf{0}$ atunci k = k + 1 și salt la 2.

Altfel, algoritm terminat, și întoarce $\mathbf{w}_{k+1}$ ca minimul dorit.

Deoarece hessianul $E''(\mathbf{w}_k)$ care trebuie calculat la fiecare iterație e dificil de obținut și implică cerințe mari de memorie și putere de calcul se preferă determinarea α_k ca fiind minimul lui E de-alungul liniei $\mathbf{w}_k + \alpha_k \mathbf{p}_k$ ("line search").

Este însă posibil să fie folosit, în estimarea pasului de căutare, și alt principiu, diferit de cel al căutării după o linie ("line search"). Acesta se bazează pe aproximarea (Hestenes [56]):

$$s_k = E''(\mathbf{w}_k)\mathbf{p}_k \approx \frac{E'(\mathbf{w}_k + \sigma_k \mathbf{p}_k) - E'(\mathbf{w}_k)}{\sigma_k},\ 0 < \sigma_k << 1 \qquad (4.2.2.6)$$

În practică însă nici această metodă nu oferă rezultate satisfăcătoare, aproximația pătratică (4.2.2.1) nefiind suficient de precisă.

Aceste dezavantaje sunt surmontate de către **algoritmul folosit în cadrul acestei lucrări pentru cazul în care vectorii de intrare ai RNA au dimensiuni mici și medii ($<10^2$ componenete)**, și anume algoritmul **gradientului conjugat scalat modificat**. Acesta folosește o formă ușor modificată a (4.2.2.6) și anume cea de la

(4.2.2.7), şi elimină nedeterminările hessianului funcţiei E în anumite puncte ale spaţiului ponderilor:

$$s_k = \frac{E'(\mathbf{w}_k + \sigma_k \mathbf{p}_k) - E'(\mathbf{w}_k)}{\sigma_k} + \lambda_k \mathbf{p}_k \qquad (4.2.2.7)$$

în care λ_k este modificat la fiecare iteraţie în funcţie de semnul lui δ_k care informează dacă hessianul nu este pozitiv definit, adică aproximarea (4.2.2.1) nu e precisă.

Având în vedere aceste modificări, algoritmul gradientului conjugat scalat modificat arată în următorul mod:

1. Se alege vectorul pondere iniţial $\mathbf{w}_1$ şi mărimile scalare:

   $0 < \sigma < 10^{-4}$, $0 < \lambda_1 < 10^{-6}$, $\bar{\lambda}_1 = 0$

   $\mathbf{p}_1 = \mathbf{r}_1 = -E'(\mathbf{w}_1)$, k = 1, SUCCES = ADEVARAT

2. Dacă SUCCES = ADEVARAT, se calculează informaţia de ordinul 2:

   $\sigma_k = \sigma / |\mathbf{p}_k|$

   $$s_k = \frac{E'(\mathbf{w}_k + \sigma_k \mathbf{p}_k) - E'(\mathbf{w}_k)}{\sigma_k}$$

   $\delta_k = \mathbf{p}^T{}_k \mathbf{s}_k$

3. Se scalează δ_k: $\delta_k = \delta_k + (\lambda_k - \bar{\lambda}_k) |\mathbf{p}_k|^2$

4. Dacă $\delta_k < 0$ atunci hessianul va fi făcut pozitiv definit:

   $\bar{\lambda}_k = 2(\lambda_k - \delta_k / |\mathbf{p}_k|^2)$

   $\delta_k = -\delta_k + \lambda_k |\mathbf{p}_k|^2$

   $\lambda_k = \bar{\lambda}_k$

5. Se calculează mărimea pasului de căutare:

   $\mu_k = \mathbf{p}^T{}_k \mathbf{r}_k$

   $\alpha_k = \mu_k / \delta_k$

6. Calcularea parametrului de comparaţie:

   $\Delta_k = 2\delta_k [E(\mathbf{w}_k) - E(\mathbf{w}_k + \alpha_k \mathbf{p}_k)] / \mu_k^2$

7. Dacă $\Delta_k \geq 0$:

$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \mathbf{p}_k$

$\mathbf{r}_{k+1} = E'_{qw}(\mathbf{w}_{k+1})$

$\bar{\lambda}_k = 0$, SUCCES = ADEVĂRAT

Dacă k mod N = 0 repornește algoritmul:

$\mathbf{p}_{k+1} = \mathbf{r}_{k+1}$

Altfel:

$\beta_k = (|\mathbf{r}_{k+1}|^2 - \mathbf{r}^T_{k+1}\mathbf{r}_k) / \mu_k$

$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$

Dacă $\Delta_k \geq 0.75$ se reduce parametrul de scalare:

$\lambda_k = (1/4)\lambda_k$

Altfel:

$\bar{\lambda}_k = \lambda_k$

SUCCES = FALS

8. Dacă $\Delta_k < 0.25$ se mărește parametrul de scalare:

$\lambda_k = \lambda_k + (\delta_k(1 - \Delta_k) / |\mathbf{p}_k|^2)$

9. Dacă direcția descendentă $\mathbf{r}_k \neq \mathbf{0}$ atunci k = k + 1 și salt la 2.

Altfel, algoritm terminat, și întoarce $\mathbf{w}_{k+1}$ ca minimul dorit.

Algoritmul de mai sus a fost implemenat în mediul MATLAB 5.3, Neural Networks Toolbox v.3.0.1 sub forma funcției de antrenament "trainscg".

4.3 Concluzii

În cadrul acestui capitol au fost analizate probleme de neurodinamică. O importanță deosebită a fost acordată algoritmilor de antrenament pentru RNA feedforward bazați pe corecția erorii.

Autorul tezei a analizat performanțele acestora oferind și soluții de rezolvare pentru problemele specifice ce apar în abordarea cu RNA a aplicațiilor gen recunoaștere

facială [44], [45], [58]. Principalele concluzii sunt enunţate în cele ce urmează.

Algoritmii tip gradient descendent au o serie de dezavantaje (viteză de convergenţă redusă, parametri a căror valoare optimă nu poate fi specificată apriori etc.) care îi fac inutilizabili în aplicaţii ca recunoaşterea facială. În concluzie, trebuie utilizaţi algoritmi mai sofisticaţi, ca şi cei enumeraţi în continuare, care să permită manipularea corespunzătoare a unui volum ridicat de date.

Algoritmii de tip Newton, incluzând aici şi metoda Levenberg-Marquardt [59], prezintă dezavantajul complexităţii extrem de ridicate a calculelor ce trebuie efectuate la fiecare iteraţie: calculul derivatelor parţiale de ordinul I şi II pentru funcţia de cost şi o inversă a hessianului acesteia. Mai mult, în abordarea Newton, este posibil ca această inversă să nu existe. Aceste afirmaţii pot fi susţinute şi pe baza rezultatelor din tab.4.2.1.2, în care funcţia de antrenament "trainlm", ce implementează metoda Levenberg-Marquardt, nu a reuşit să rezolve problemele ce presupuneau vectori de intrare cu dimensionalitate medie şi mare.

Algoritmii de tip cvasi-Newton (BFGS, Broyden-Fletcher-Goldfarb-Shanno) [60] şi cei bazaţi pe metoda gradientului conjugat (Polak-Ribiere, Fletcher-Reeves) [61] deşi au cerniţe ceva mai reduse în ceea ce priveşte puterea de calcul faţă de clasa de metode referită anterior, sunt bazaţi pe principiul căutării după o direcţie. Aceasta înseamnă că trebuie calculată, pentru fiecare iteraţie, o funcţie eroare globală care să conducă la o mărime corespunzătoare a pasului de căutare.

Experimentele, prezentate detaliat în subcapitolul 4.2, au confirmat drept cele mai potrivite metode de antrenament:

a) pentru cazul în care nu sunt extrase trăsături, deci vectori de intrare de mari dimensiuni, funcţia de antrenament "trainfuzzy" (vezi paragraful 4.2.1 referitor la accelerarea vitezei de convergenţă a algoritmului BP prin reglarea fuzzy a ratei de învăţare);

b) pentru cazul în care datele de intrare sunt preprocesate, funcţia "trainscg" (vezi paragraful 4.2.2 privind accelerarea vitezei de convergenţă pentru algoritmi bazaţi pe metoda gradientului conjugat) care implementează metoda gradientului conjugat scalat.

CAPITOLUL 5

Metode folosite în recunoaşterea facială

Prezentul capitol este dedicat metodelor folosite în tehnica recunoaşterii faciale.

Se începe, în subcapitolul 5.1, prin prezentarea modului biologic de procesare a informaţiei vizuale, aspect deosebit de important având în vedere capacitatea sistemului vizual uman de a recunoaşte mii de obiecte şi, în particular, feţe.

Subcapitolul 5.2 prezintă metode de recunoaştere facială clasice (convenţionale, bazate pe statistica tiparelor) iar subcapitolul 5.3 este dedicat metodelor de recunoaştere facială bazate pe reţele neuronale.

Sunt subliniate avantajele şi dezavantajele acestor metode; pe baza acestor observaţii sunt desprinse concluzii (subcapitolul 5.4) referitoare la arhitectura optimă a unui sistem pentru recunoaştere facială.

5.1 Procesarea biologică a informaţiei vizuale

Studiul modului de procesare al informaţiei vizuale reprezintă un aspect important al succesului oricărei aplicaţii care îşi propune să rezolve problema recunoaşterii faciale. Este cunoscută ***abilitatea deosebită*** a sistemului vizual uman de a recunoaşte obiecte, în particular feţe, în condiţii dintre cele mai diverse în ceea ce priveşte nivelul iluminării sau poziţia obiectelor în scenă.

Diversele structuri, cf. Gupta şi Knopf [62], ce se găsesc de-a lungul căii vizuale sunt prezentate în fig. 5.1.1.

- **Retina.** Lumina reflectată de diversele obiecte este focalizată pe un strat de celule fotoreceptoare. Receptorii retinieni includ aproximativ 120 de miloane de celule cu bastonaşe şi 6 milioane de celule cu conuri. Conurile sunt sensibile la lumina puternică, iar bastonaşele la lumina slabă. De aceea, celulele cu bastonaşe servesc vederii nocturne, iar cele cu conuri vederii diurne, color. O densitate mare a celulelor cu conuri

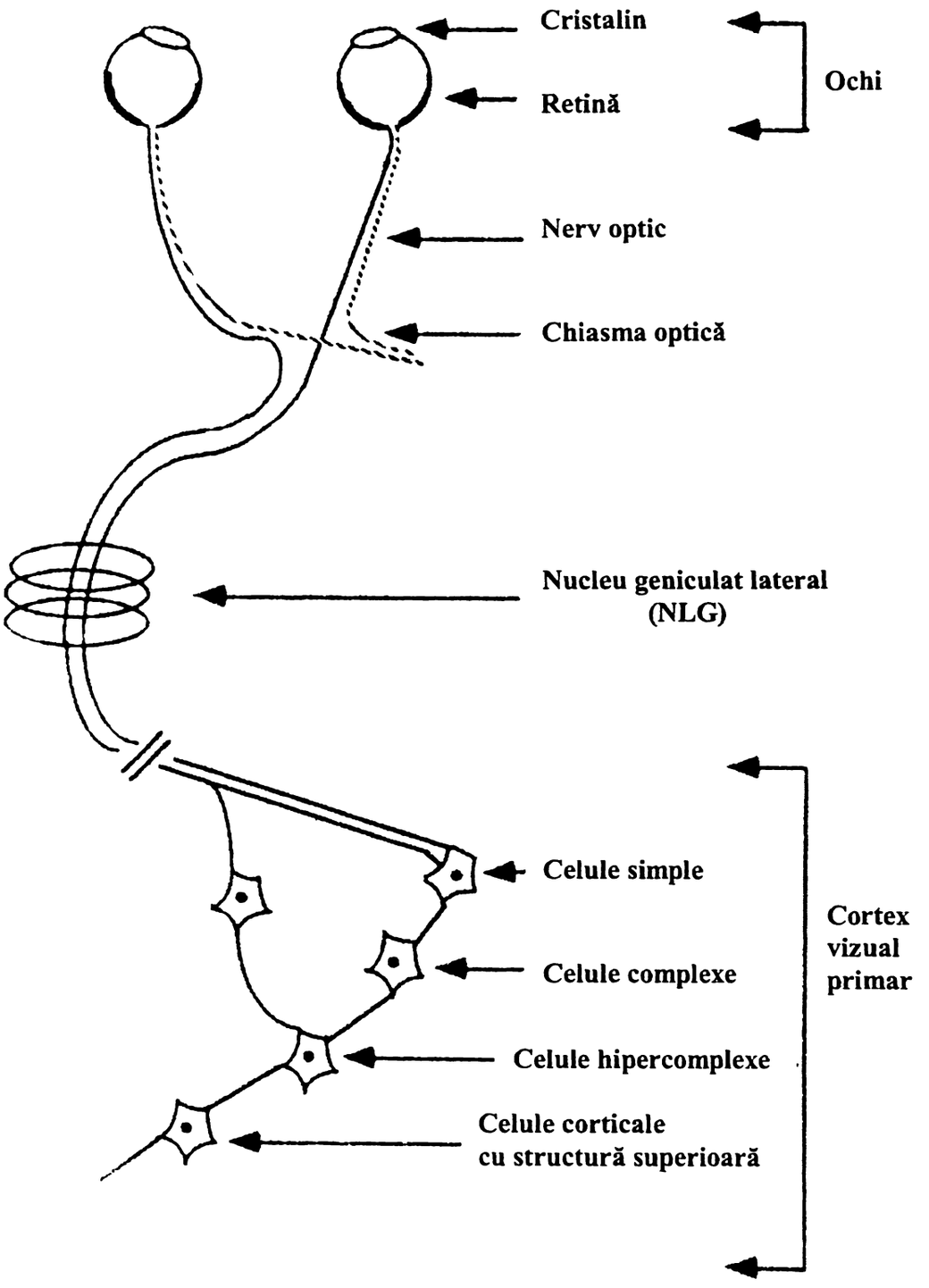


Fig.5.1.1 Structura căii vizuale biologice, prezentată de autorul tezei în [29].

este întâlnită în partea centrală a retinei, numită fovee, iar bastonașele domină regiunea periferică a retinei.

În termeni de procesare a informației, fotoreceptorii din retină se comportă ca niște **traductori** care convertesc energia luminoasă într-un semnal electric echivalent. Aceste semnale sunt mai apoi convertite într-un **tren de potențiale** de acțiune de către celulele ganglionare, aceste potențiale fiind transmise nervului optic. Deci celulele ganglionare funcționează ca un **codor** pentru transmiterea semnalelor vizuale de-a lungul unei linii de comunicație numită nerv optic. Deși există aproximativ 126 de milioane de fotoreceptori în retină, numărul axonilor ce pleacă din ganglioni și transmit toată informația vizuală cortexului, nu depășește 1 milion. În concluzie, retina efectuează o procesare de nivel scăzut a semnalului și o **compresie cu rata 126:1** a informației vizuale.

- **Nucleul lateral geniculat (NLG).** Înainte ca informația de la retină să ajungă la cortexul vizual primar, axonii celulelor ganglionare se împart în două grupuri. Jumătate din fibre, provenind din câmpul vizual stâng, trec printr-un NLG, iar cealaltă jumătate, provenind din câmpul vizual drept, prin cel de al-II-lea NLG.

Rolul NLG în procesarea informației vizuale nu este pe deplin elucidat; se pare că au un comportament asemănător cu cel al unei **stații releu** pentru redirecționarea informației în diferite părți ale cortexului. Unii cercetători presupun că celulele NLG sunt intens folosite în vederea color.

- **Cortexul vizual primar.** Semnalele transmise de NLG ajung în cortexul vizual primar (denumit și cortex vizual striat) situat în partea din spate a creierului (lobul occipital). Morfologia neuronală a acestuia se constituie sub forma unor structuri în formă de coloane, numite hipercoloane. Astfel cortexul poate fi modelat printr-o matrice 3D.

Aceste coloane neuronale validează anumiți neuroni pentru a răspunde selectiv la orientări specifice, mișcări și direcții de mișcare a stimulilor vizuali. Neuronii fiecărei coloane sunt interconectați cu neuronii altor coloane, ca și cu alte părți din creier.

Sunt întâlnite trei tipuri de celule nervoase selective la orientare. Primul tip de neuron, denumit **celulă corticală simplă,** răspunde cel mai bine la bare de lumină proiectate pe suprafaţa retiniană. Al II-lea tip, numit **celulă corticală complexă** are cel mai puternic răspuns la bare orientate de lumină care se mişcă în întreg câmpul receptiv. Al III-lea tip de celule, **celulele corticale hipercomplexe,** răspund la stimuli vizuali constituiţi din linii sau unghiuri mişcătoare de lungime specificată. Lungimea stimulilor nu are nici un efect asupra răspunsului celulelor complexe, dar are efect direct asupra celor hipercomplexe, astfel încât dacă stimulul este prea lung (sau prea scurt) celula nervoasă nu va fi activată.

- **Regiuni corticale înalte.** Informaţia vizuală din cortexul primar trece prin aproximativ 20 de structuri paralele (arii vizuale) ce se găsesc în creier. Unii neurofiziologi susţin că există două magistrale informaţionale care diverg din cortexul vizual primar. O cale transferă informaţia spre ariile temporale inferioare ale creierului. Se presupune că aceste regiuni sunt responsabile pentru **analiza de culoare şi formă,** necesară recunoaşterii obiectelor din câmpul vizual. O a II-a cale transferă informaţia către ariile parietale inferioare, responsabile **de analiza relaţiilor spaţiale** dintre obiecte şi de modul în care mişcarea obiectelor schimbă aceste relaţii. Mai mult, există căi spre lobii temporali şi frontali validând asocierea informaţiei vizuale cu informaţia provenită de la alte simţuri: auditiv, olfactiv, tactil.

La nivelurile cele mai avansate ale căii vizuale se pare că există neuroni corticali de nivel foarte înalt care se activează numai la stimuli foarte specifici. De exemplu, un anume neuron (**grandmother cell**) va răspunde pozitiv la o anumită imagine facială. Aceste celule par a fi situate la cel mai înalt nivel al reţelei neuronale, ele fiind probabil o parte dintr-o structură de calcul care răspunde, colectiv, de o anume entitate.

5.2 Metode convenţionale folosite în recunoaşterea facială

5.2.1 Metode bazate pe trăsături geometrice

Una din primele metode folosite pentru recunoaşterea facială a fost propusă de Kanade [63]. Aceasta presupunea extragerea unui set de trăsături faciale, calcularea unui vector cu 16 dimensiuni pe baza unor rapoarte de distanţe (şi arii) aferente trăsăturilor faciale şi compararea a două feţe pe baza unei sume de distanţe. Pentru o bază de date conţinând 20 de imagini faciale, Kanade a obţinut o rată de recunoaştere între 45% şi 75%, folosind un algoritm de extragere automată a trăsăturilor faciale.

Brunelli şi Poggio [64] compară metodele bazate pe trăsături geometrice cu cele bazate pe şabloane. Etapele pe care autorii le propun, în cazul primei metode, sunt:

a) Normalizarea

Este efectuată în scopul obţinerii unei invarianţe la poziţia, scara sau rotaţia feţei în planul imaginii. Dependenţa de translaţie poate fi eliminată odată ce originea coordonatelor este aleasă într-un punct care poate fi detectat cu o precizie bună în fiecare imagine.

Normalizarea presupune calcularea unui coeficient de intercorelaţie:

$$C_N(y) = \frac{< I_T T > - < I_T >< T >}{\sigma(I_T)\sigma(T)} \qquad (5.2.1.1)$$

în care I_T este o fereastră a imaginii I care trebuie comparată cu şablonul ("template") T, $< >$ reprezintă operatorul medie, $I_T T$ înseamnă produsul pixel cu pixel, iar σ este deviaţia standard pentru ariile care sunt comparate.

b) Extragerea trăsăturilor

În acest scop este calculată proiecţia integrală a imaginii $I(x, y)$, în dreptunghiul de coordonate $[x_1, x_2]$ x $[y_1, y_2]$:

$$\begin{cases} V(x) = \sum_{y=y_1}^{y_2} I(x,y) \\ H(y) = \sum_{x=x_1}^{x_2} I(x,y) \end{cases} \quad (5.2.1.2)$$

în care *V(x)* şi *H(y)* reprezintă proiecţia integrală verticală, respectiv orizontală. Proiecţiile astfel obţinute sunt extrem de eficiente în determinarea poziţiei trăsăturilor.

Cele 35 de trăsături geometrice extrase automat şi folosite ulterior în procesul de recunoaştere sunt următoarele:

- grosimea sprâncenelor şi poziţia lor verticală faţă de centrul ochilor;
- o descriere grosieră (11 coeficienţi) a arcadei;
- poziţia verticală a nasului precum şi lăţimea sa;
- poziţia orizontală a gurii, lăţimea sa, înălţimea buzelor;
- 11 raze care descriu forma bărbiei;
- lăţimea bigonială (lăţimea feţei măsurată la nivelul nasului);
- lăţimea zigomatică (lăţimea feţei măsurată la jumătatea distanţei dintre vârful nasului şi ochi).

c) Recunoaşterea

Este efectuată prin intermediul unui clasificator Bayes. Se presupune că vectorii de trăsături, pentru o singură persoană, sunt distribuiţi gaussian. Persoane diferite sunt caracterizate doar de valoarea medie, în timp ce distribuţia este comună. Acest lucru permite estimarea formei distribuţiei, adică a matricii de covarianţă Σ:

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N}\Sigma_i \quad (5.2.1.3)$$

în care Σ_i reprezintă estimarea matricii de covarianţă obţinută din datele pentru persoana "i". În consecinţă, probabilitatea unei măsurători poate fi asociată cu distanţa:

$$\Delta_j(\mathbf{x}) = (\mathbf{x} - \mathbf{m}_j)^T \Sigma^{-1} (\mathbf{x} - \mathbf{m}_j) \quad (5.2.1.4)$$

unde $\mathbf{m}_j$ este vectorul median care reprezintă persoana "j". Un vector necunoscut va fi asociat cu cel mai apropiat vecin din baza de date.

Este raportată o rată de recunoaştere de 90% pentru o bază de date ce cuprinde

47 de persoane.

Cox şi colectivul [65] introduc ***tehnica distanței combinate***. Aceasta constă în extragerea unui vector 30-dimensional obținut prin localizarea ***manuală*** a 35 de puncte de interes şi măsurarea anumitor distanțe.

Presupunând că un model combinat n-dimensional M modelează elementele bazei de date Y=$\{y_i\}$ iar $\boldsymbol{q}$ reprezintă un vector care trebuie asociat unui y_i, se poate scrie:

$$P_r(q \mid y, M) = \frac{1}{P_r(y \mid M)} P_r(qy \mid M) = \frac{1}{P_r(y \mid M)} \sum_{k=1}^{n} P_r(qy \mid M_k) P_r(M_k) =$$

$$= \frac{1}{P_r(y \mid M)} \sum_{k=1}^{n} P_r(q \mid y, M_k) P_r(y \mid M_k) P_r(M_k) =$$

$$= \frac{1}{P_r(y \mid M)} \sum_{k=1}^{n} P_r(q - y \mid \overline{M}_k) P_r(y \mid M_k) P_r(M_k) =$$

$$= \sum_{k=1}^{n} P_r(q - y \mid \overline{M}_k) P_r(M_k \mid y) \qquad (5.2.1.5)$$

în care $P_r(q \mid y, M_k) = P_r(q - y \mid \overline{M}_k)$ şi $P_r(M_k \mid y) = \dfrac{P_r(y \mid M_k) P_r(M_k)}{\sum_{i=1}^{n} P_r(y \mid M_i) P_r(M_i)}$

În cazul în care se foloseşte un singur element combinat, n = 1, distanța combinată se reduce la distanța Mahalanobis de la $\boldsymbol{q}$ la prototipul μ.

Folosind această metodă s-a obținut o rată de recunoaştere de 95% în condițiile în care au existat 95 de imagini de test şi un total de 685 imagini.

Observații:

Metodele prezentate presupun o măsurare exactă a distanțelor aferente trăsăturilor şi prezintă rezultate bune doar în condițiile unor baze de date largi, care conțin imagini la care perspectiva şi iluminarea sunt relativ constante, de exemplu bazele de date ale poliției.

Performanțele unui astfel de sistem sunt strâns legate de acelea ale algoritmului care localizează trăsăturile. Algoritmii actuali de localizare automată a punctelor de interes nu furnizează un grad înalt de precizie şi au cerințe mari în ceea ce priveşte

capacitatea de calcul (Sutherland şi colectiv [66]). ***De asemenea astfel de sisteme nu sunt suficient de robuste în privinţa schimbărilor expresiei faciale.***

5.2.2 Metode bazate pe analiza componentelor principale

Prezentată pentru prima dată de către Turk şi Petland [67], metoda feţelor proprii ("eigenfaces") reprezintă un punct de referinţă pentru diversele abordări ulterioare.

Trăsăturile extrase, denumite feţe proprii, reprezintă un set de vectori bază ortonormali, calculaţi dintr-o anumită colecţie de imagini faciale. Aceşti vectori reprezintă o bază de dimensiune redusă pentru reprezentarea imaginilor faciale şi sunt optimi în sensul erorii pătratice medii [67].

Notând mulţimea de N imagini de antrenament prin $\{z_1, z_2, \ldots, z_N\}$, se aplică analiza componentelor principale (PCA, Principal Component Analysis) în scopul găsirii a N vectori proprii ai matricii de covarianţă $\frac{1}{N}\sum_{n=1}^{N}(z_n - \bar{z})(z_n - \bar{z})^T$ în care $\bar{z} = \frac{1}{N}\sum_{n=1}^{N} z_n$ reprezintă media ansamblului de imagini. Sunt calculate valorile proprii ale matricii de covarianţă.

Fie Φ_k vectorul propriu corespunzător celei de a "k"-a valoare proprie. Primii N' ($\leq N$) vectori ortonormali $\Phi_1, \ldots, \Phi_{N'}$ formează o bază în spaţiul feţelor proprii. În [67] se arată că $N' = 40$ este suficient pentru a descrie N=115 imagini faciale.

Clasificarea bazată pe feţe proprii este efectuată în două etape:

a) **Extragerea trăsăturilor proprii**. Fiecare imagine de antrenament z_n este proiectată în spaţiul feţelor proprii ca un punct $x_n = \Phi^T(z_n - \bar{z})$ cu $\Phi = [\Phi_1, \ldots, \Phi_{N'}]$, şi este folosită ca un punct prototip. Fiind dată o imagine necunoscută z, se calculează pentru aceasta $x = \Phi^T(z - \bar{z})$.

b) **Clasificarea bazată pe vectorii proprii de tip trăsătură**. Metoda de clasificare cea mai simplă este bazată pe distanţa euclidiană $d(x, x_n)$ folosind

criteriul vecinului cel mai apropiat. În [67] este folosit criteriul centrului cel mai apropiat, în care o clasă este reprezentată de către centrul tuturor x_n aparţinători acelei clase, iar clasificarea se bazează pe distanţa de la x la fiecare centru.

Turk şi Petland prezintă rezultatele asupra unei baze de date constituită din 16 subiecţi, la care variază orientarea capului, scara şi condiţiile de iluminare. Pentru variaţii ale iluminării, orientării şi scării sistemul obţine 96%, 85% respectiv 64% rata clasificărilor corecte.

În [68] Moghaddan şi Petland prezintă rezultate foarte bune asupra bazei de date FERET, doar o singură greşeală făcută în clasificarea a 150 de imagini frontale. Sistemul foloseşte o preprocesare extensivă în scopul localizării capului, detecţiei trăsăturilor şi normalizării imaginii.

În [69] Moghaddam, Wahid şi Petland introduc noţiunea de măsură probabilistică a similarităţii, bazată pe probabilitatea ca diferenţa intensităţilor imaginilor, $\Delta = I_1 - I_2$, să fie caracteristică variaţiilor tipice ale aceluiaşi obiect. De exemplu, pentru scopul recunoaşterii faciale, se pot defini două clase pentru variaţiile unei imagini faciale: **variaţii intrapersonale** Ω_I (corespunzătoare, spre exemplu, diverselor expresii faciale ale aceluiaşi individ) şi **variaţii extrapersonale** Ω_E (corespunzătoare variaţiilor dintre indivizi diferiţi). Măsura similarităţii poate fi exprimată atunci, ca fiind:

$$S(I_1, I_2) = P(\Delta \in \Omega_I) = P(\Omega_I \mid \Delta) \qquad (5.2.2.1)$$

în care $P(\Omega_I \mid \Delta)$ reprezintă o probabilitate a posteriori dată de regula lui Bayes, folosind estimări ale $P(\Delta \mid \Omega_I)$ şi $P(\Delta \mid \Omega_E)$.

Sistemul realizat după principiile enunţate mai sus s-a dovedit a fi cel mai bun în competiţia FERET.

Liu şi Wechsler propun metoda proiecţiilor optime [70]. Tehnica OPA (Optimal Projection Axes) integrează PCA, albizarea şi transformarea de rotaţie cu căutarea prin intermediul algoritmilor genetici. În primul rând PCA proiectează imaginile într-un spaţiu cu mai puţine dimensiuni, păstrând informaţia reprezentativă pentru imaginea

originală. Albizarea contrabalansează faptul că eroarea pătratică medie, aferentă PCA, acordă o pondere însemnată frecvenţelor joase. Prin albizare, normele (distanţele) nu sunt conservate. Se pot aştepta astfel performanţe îmbunătăţite de la o bază neortogonală. Urmează o rotaţie a vectorilor bază, ghidată de un algoritm evolutiv.

Pentru experimentare sunt folosite 1107 imagini faciale corespunzătoare la 369 subiecţi din baza de date FERET. Autorii arată obţinerea unor rezultate mai bune decât cele ale metodelor feţelor proprii sau MDF (Most Discriminant Features).

S.Z. Li şi I. Lu prezintă în [71] o nouă metodă de clasificare denumită metoda celei mai apropiate linii de trăsături. Oricare două puncte în spaţiul trăsăturilor sunt caracterizate de o dreaptă care trece prin ele. Trăsăturile sunt obţinute prin metoda feţelor proprii. Clasificarea este bazată pe determinarea distanţei minime de la punctul-trăsătură care necesită a fi clasificat la liniile de trăsături existente. Se raportează o performanţă record asupra bazei de date ORL şi anume o rată a erorii de 3.125%.

Observaţii:

Rezumând chestiunile prezentate în cadrul acestui paragraf, se poate afirma că metoda feţelor principale reprezintă un algoritm simplu şi rapid. Problema principală o reprezintă posibilele limitări ale aplicabilităţii acestui algoritm ca urmare a necesităţii existenţei unui grad înalt de corelaţie între intensităţile pixelilor imaginilor de test şi antrenament. Acest dezavantaj este de regulă surmontat prin preprocesarea extensivă în scopul normalizării imaginilor.

5.2.3 Metode bazate pe compararea şabloanelor

Compararea şabloanelor ("template matching") reprezintă o tehnică relativ simplă şi presupune, în general, parcurgerea următorilor paşi [64], [72]:

- **Normalizarea imaginii**. Se poate efectua, calculând I / < I > adică raportul dintre valoarea locală a intensităţii şi media intensităţilor unei vecinătăţi. Alte modalităţi constau în calculul intensităţii gradientului imaginii ($|\partial_x I| + |\partial_y I|$) sau a laplacianului intensităţii imaginii $|\partial_{xx} I| + |\partial_{yy} I|$.

- Fiecare persoană va fi reprezentată printr-o bază de date a cărei câmpuri constau din imaginea frontală şi un set de regiuni dreptunghiulare care cuprind ochii, nasul, gura şi o porţiune cuprinsă între arcade şi bărbie (fig. 5.2.3.1)
- Faza de recunoaştere, presupune compararea şabloanelor aferente persoanei recunoscute cu cele existente în baza de date. Se va returna un vector ce indică " potrivirea" dintre şabloane prin intermediul matricii de corelaţie.

Fig. 5.2.3.1 Regiuni selectate pentru metoda comparării şabloanelor (ochii, nasul, gura şi o porţiune cuprinsă între arcade şi bărbie)

Brunelli şi Paggio, comparând în [64] metodele bazate pe trăsături geometrice şi cele bazate pe compararea şabloanelor, constată rezultate mai bune în al II-lea caz, dar "*o generalizare a rezultatelor nu ar fi potrivită, pentru că acestea sunt, în mod clar, specifice tipului de problemă şi de implementare abordat*".

Observaţie:

Tehnica comparării şabloanelor poate fi folosită doar în cazul în care imaginile de test au aceeaşi scară, orientare şi iluminare cu imaginile din setul de antrenament (cf. Cox şi colectiv [65]).

5.2.4 Metode bazate pe compararea reţelelor elastice

Compararea reţelelor elastice (Elastic Graph Matching, EGM) introduce o reprezentare specifică a feţei, prin aceea că faţa este reprezentată printr-o mulţime de vectori trăsături poziţionaţi în nodurile unei reţele dreptunghiulare care acoperă imaginea facială.

Ca şi trăsături pot fi utilizate modulele răspunsurilor unor filtre Gabor pentru diverse orientări şi rezoluţii (S. Ben-Yacoub şi colectiv [73]) sau valorile locale ale spectrului de putere (M. Lades şi colectiv [74]).

Diferenţele calitative care apar la două imagini faciale care trebuie comparate pot fi evaluate folosind următoarea distanţă:

$$d(G,R) = \sum_{i=1}^{N_n} d_n(G_{ni}, R_{ni}) + \lambda \sum_{j=1}^{N_e} d_e(G_{ej}, R_{ej}) = \sum_{i=1}^{N_n} d_{ni} + \lambda \sum_{j=1}^{N_e} d_{ej} \qquad (5.2.4.1)$$

unde G_{ni} reprezintă nodul "*i*" al reţelei G, R_{ej} nodul "*j*" al reţelei R; N_n, N_e reprezintă numărul de noduri respectiv muchii iar λ un factor pondere. O reţea plastică, care nu opune rezistenţă la deformaţii se obţine pentru $\lambda = 0$, în timp ce o reţea rigidă se obţine pentru valori mari ale lui λ.

Observaţii:

Algoritmul EGM obţine rezultate bune în cazul variaţiilor poziţiei şi expresiei feţei.

Dezavantajul acestei metode constă în complexitatea calculelor necesare etapei de extragere a trăsăturilor şi mai ales celei de comparare a reţelelor elastice: în [74] ***se menţionează un timp de comparare de 25s a unei imagini cu 87 de alte imagini folosind o maşină cu 23 de transputere lucrând în paralel!***

5.3 Metode bazate pe rețelele neuronale folosite în recunoașterea facială

5.3.1 Metodă bazată pe folosirea rețelelor neuronale cu autoorganizare și convoluționale

În [75] Lawrence și colectiv prezintă un sistem de recunoaștere facială reprezentativ pentru metodele discutate în cadrul acestui subcapitol, în care atât etapa de extragere a trăsăturilor cât și recunoașterea propriu-zisă sunt implementate prin intermediul rețelelor neuronale.

Sistemul combină eșantionarea locală a imaginilor cu **rețele neuronale cu autoorganizare**, tip hartă de trăsături (SOFM, Self-Organizing Feature Map) și **rețele neuronale convoluționale** (CN, Convolutional Network). Arhitectura acestui sistem poate fi urmărită în fig. 5.3.1.1.

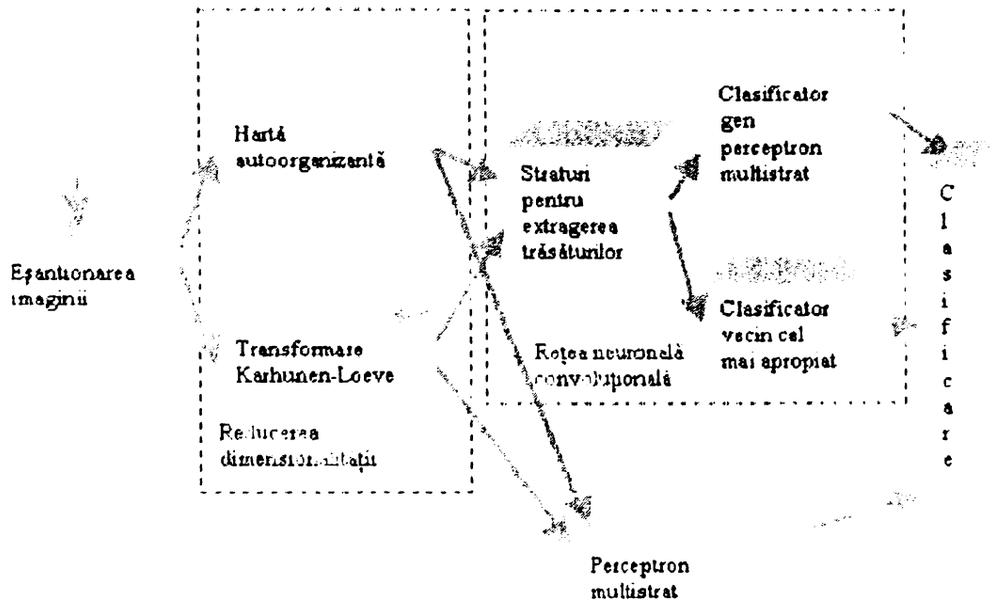


Fig.5.3.1.1 Arhitectura sistemului propus de Lawrence și colectiv [75].

Reţelele neuronale SOFM cuantifică eşantioane din imagine într-un spaţiu ordonat topologic, astfel încât la eşantioane asemănătoare aplicate la intrare se vor activa neuroni ai stratului de ieşire învecinaţi din punct de vedere topologic. Rezultă astfel, pe lângă o reducere a dimensionalităţii datelor de intrare şi o invarianţă parţială la translaţie, rotaţie, scară şi deformaţii. De asemenea sunt prezentate comparativ rezultatele obţinute folosind în loc de SOFM, transformarea Karhunen-Loeve, cu menţiunea că aceasta nu a oferit rezultate la fel de bune ca metoda de extragere a trăsăturilor bazată pe SOFM.

În scopul clasificării trăsăturilor extrase este folosită o reţea neuronală de tip convoluţional, caracterizată de folosirea câmpurilor receptive locale, folosirea în comun a ponderilor şi o eşantionare subspaţială. Sunt prezentate şi rezultate comparative obţinute pentru un clasificator tip reţea neuronală perceptron multistrat.

Rezultatele experimentale obţinute asupra bazei de date ORL (Olivetti Research Laboratory, Cambridge,UK) sunt prezentate sintetizat în tab. 5.3.1.1.

Metodă	**Rata erorii**
Feţe proprii	10,5%
Karhunen-Loeve + CN	7,5%
SOFM + CN	3.8%

Tab.5.3.1.1 Rezultatele experimentale obţinute asupra bazei de date ORL cf. [75].

Concluzia desprinsă de către autori este că soluţia SOFM+CN dă rezultatele cele mai bune într-un timp mai mic de 0,5s.

O metodă bazată pe o structură uşor modificată a unei reţele neuronale convoluţionale este prezentată în [76]. Este vorba despre o reţea neuronală de tip neocognitron care spre deosebire de modelul standard foloseşte pentru neuronii de tip S, modelul McCulloch-Pitts.

Pentru localizarea feţei este folosit un perceptron multistrat cu 10 neuroni

ascunși. Acesta va localiza imagini faciale de dimensiune fixă (32x32 pixeli) obținute analizând diferite rezoluții și orientări posibile ale feței în imagine.

Pentru testarea performanțelor rețelei s-au achiziționat 800 de imagini pentru persoana "A" și 1200 de imagini corespunzătoare la 18 persoane catalogate drept "non A". Se raportează obținerea unor performanțe de 4,5% respectiv 59,7% pentru nivelul erorii de test în cazul în care sunt impuse respectiv nu sunt impuse constrângeri (iluminare, scară, orientare) imaginilor.

Avantajul unei astfel de arhitecturi constă în eliminarea fazei de preprocesare a imaginii în scopul normalizării și extragerii trăsăturilor. Acest lucru este realizat implicit de către rețeaua neuronală convoluționanală, trăsături de complexitate crescândă fiind extrase succesiv, pe măsura procesării informației de către straturile ascunse de nivel ierarhic crescător.

5.3.2 Metoda bazată pe rețele neuronale cu decizie probabilistică

În [77] Lin și colectiv prezintă un sistem constituit, în principal, din trei module:

- un detector de fețe, capabil să localizeze o imagine facială conținută într-o anumită scenă;
- un modul care localizează poziția ochilor, în scopul generării vectorilor de trăsături. Regiunile feței pentru care sunt extrase trăsături (gradient pe direcțiile x și y, intensitatea regiunilor) cuprind sprâncenele, ochii și nasul, exclusiv gura;
- ultimul modul este destinat recunoașterii feței.

Se menționează că toate cele trei module sunt implementate cu ajutorul **rețelelor neuronale cu decizie probabilistică** (PDBNN, Probabilistic Decision-Based Neuronal Networks).

Două tipuri de trăsături sunt extrase din imaginile originale: intensități și muchii (imagini gradient). Astfel, acești vectori vor fi aplicați la intrările a două RNA-PBD. Intensitățile pixelilor și valorile muchiilor sunt normalizate, între 0 și 1, pentru a compensa schimbările în iluminare. În acest scop sunt aplicate filtrări și modificări ale

histogramei. În final, imaginile normalizate şi recondiţionate (de dimensiune 140x100) sunt reeşantionate ajungând la rezoluţii scăzute (14x10).

Avantajul acestei soluţii constă în structura modulară a PDBNN, fiecărei persoane alocându-i-se o subreţea. Autorii raportează o rată a erorii pentru baza de date ORL de 4%, un timp de antrenament de 20 min. şi un timp de recunoaştere sub 0,1 s folosind maşini SGI Indy.

5.3.3 Metode bazate pe arbori neuronali

În [78] Wilder propune un sistem bazat pe o metodă de extragere a trăsăturilor ce constă într-o transformată (Hadamard, Cosinus Discretă şi Karhunen-Loeve) a **proiecţiei scării de gri**. Proiecţia scării de gri se obţine efectuând suma valorilor intensităţilor pixelilor de-a lungul unei direcţii specifice. Se obţine astfel o serie unidimensională de semnături reprezentând date 2D. În lucrarea menţionată anterior sunt folosite proiecţiile pe direcţiile orizontală şi verticală dintr-o fereastră rectangulară ce încadrează bărbia, urechile şi fruntea. Apoi proiecţiile scării de gri sunt integrate în benzi suficient de înguste pentru a detecta trăsăturile principale şi, în acelaşi timp, suficient de largi pentru a ignora trăsăturile fine. În continuare trăsăturile sunt supuse unor transformări ortonormale. Se arată că transformarea cosinus discretă oferă performanţe superioare metodei Hadamard şi Karhunen-Loeve.

În etapa de clasificare este implicat un **arbore ierarhic neuronal**. Spre deosebire de MLP, în acest caz numărul de neuroni nu trebuie specificat apriori. Arborele neuronal se dezvoltă pe măsură ce învaţă şi are exact numărul de neuroni necesar clasificării. În plus, arborele neuronal nu se "înţepeneşte" în minime locale, convergând întotdeauna spre o soluţie.

În etapa experimentală s-au folosit 32 de subiecţi, 6 imagini/subiect din care 3 în faza de antrenament şi 3 în faza de test. S-a obţinut o rată globală a erorii de 3%.

5.3.4 Metode ce folosesc rețele neuronale bazate pe funcții radiale

A.J. Hovell şi H. Buxton [79] prezintă o combinație dintre o tehnică de extragere a trăsăturilor bazată pe funcții Gabor şi RNA-RBF. Sunt raportate rezultate referitoare la baza de date ORL: 5% eroare la datele de test pentru cazul 5 imagini de antrenament pentru o persoană.

Sato, Shah şi Aggarwal propun în [80] un sistem care foloseşte imagini parțiale ale feței (urechi, ochi şi nas). Fără a fi extrase nici un fel de trăsături, aceste subimagini reeşantionate în scopul obținerii unei rezoluții mai slabe sunt aplicate la intrarea unei **rețele neuronale bazate pe funcții radiale** (RBF, Radial Basis Function).

Pentru testarea performanțelor sistemului, sunt folosite 50 persoane x 3 imagini în etapa de antrenament şi 70 persoane x 6 imagini în etapa de test. Sunt raportate rate ale recunoaşterii şi de rejecție de 100%.

Observații:

1) ***Regiunile care includ urechile, ochii şi nasul sunt extrase manual.***
2) ***Pentru prelevarea imaginii sunt folosite două camere de luat vederi, una pentru imaginea frontală şi una pentru cea laterală. Acest lucru implică o aliniere precisă a camerelor şi distanțe fixe de la cameră la subiect.***
3) ***Nu sunt prezentate rezultate experimentale pentru cazul în care iluminarea, scara sau poziția feței sunt variabile.***

Ratele de acceptare şi rejecție se explică doar pe baza observațiilor (1) - (3).

5.3.5 Metode bazate pe rețele neuronale de tip perceptron multistrat (MLP) şi cu învățare prin cuantizare vectorială (LVQ)

Cabello, Sanchez şi Pastor compară în [81] performanțele a două tipuri de **rețele neuronale, LVQ şi MLP**. Sunt analizate cazurile în care se folosesc intensitățile imaginilor originale sau segmentate şi cazul în care sunt extrase trăsături geometrice bazate pe distanțele aferente unor puncte localizate ***manual***. Preprocesarea imaginilor

include reducerea rezoluţiei şi segmentarea imaginii. Pentru cazul trăsăturilor geometrice se aplică şi transformarea Karhunen-Loeve.

Pentru faza experimentală, s-a folosit o bază de date de 300 de persoane. Imaginile corespund la 30 de bărbaţi de aproximativ aceeaşi vârstă şi rasă.

Concluziile experimentelor arată că dacă sunt folosite imaginile în tonuri de gri drept vectori de intrare, RNA-LVQ oferă rezultate mai bune decât RNA-MLP (96,7% respectiv 83,3%). Prin aplicarea etapei de segmentare se constată o scădere drastică a ratei de recunoaştere.

Pentru cazul trăsăturilor geometrice situaţia este inversată: MLP oferă rezultate mai bune decât LVQ (93,3% faţă de 84,4%).

5.3.6 Metode bazate pe reţele neuronale tip grupuri de arbori cu toleranţă adaptivă

În [82], Zhang şi Fulcher introduc modelul RNA bazate pe **grupuri de arbori cu toleranţă adaptivă** (GAT, Group-based Adaptive tolerance Tree) pentru recunoaşterea facială invariantă la translaţie în vederea realizării unui sistem de securitate pentru aeroporturi.

În cazul unor astfel de arbori, fiecare nod corespunde unei categorii separate, iar decizia este luată într-o manieră descendentă, la nivelul fiecărui nod intermediar în funcţie de apartenenţa eşantionului de la intrare la o anumită subclasă. În general N niveluri sunt necesare pentru clasificarea a 2^N categorii. Sunt implementaţi arbori GAT separaţi pentru recunoaşterea feţelor cu ochelari sau barbă şi pentru clasificarea perspectivei feţei.

Pentru testarea performanţelor sistemului sunt utilizate 780 de imagini (10 perspective x 78 indivizi). 87 de imagini au fost utilizate în etapa de antrenament, iar 693 pentru test. Eroarea raportată este de 0,15% ceea ce corespunde unei singure imagini clasificate greşit.

5.3.7 Metode bazate pe rețele neuronale ART

Pessoa şi Leitao prezintă în [83] un sistem de recunoaştere facială bazat pe metode de **filtrare** inspirate din **modalitatea biologică de procesare a informației vizuale**. In **etapa de clasificare**, trăsăturile extrase pe baza răspunsului unor celule complexe, se aplică unei **rețele neuronale fuzzy de tip ART** (Adaptive Resonance Theory).

Prima etapă a filtrării imaginii constă în aplicarea unor ***filtre neorientate*** care corespund filtrării biologice efectuată la nivelul ganglionilor retinieni. Ieşirea acestor filtre e dată de ecuația:

$$R = \left[\frac{\beta E - \gamma I}{\alpha + E + I} \right]^{+} \tag{5.3.7.1}$$

în care [x]+ = max(x,0), β = 0,5, γ = 0,5, α = 10. Contribuțiile excitatorii (E) şi inhibitorii (I) sunt date de:

$$E = L*G_c \text{ şi } I = L*G_s \tag{5.3.7.2}$$

unde funcțiile de ponderare centru ("center") G_c şi înconjurător ("surround") G_s sunt gaussieni normalizați de forma:

$$G = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-i)^2 + (y-i)^2}{2\sigma^2} \right) \tag{5.3.7.3}$$

cu σ = 0,5 pentru G_c şi σ = 2 pentru G_s.

Aceste tipuri de filtre sunt sensibile la discontinuități ale luminanței şi răspunsul lor este proporțional cu raportul acestor luminanțe.

Etapa a II-a implică folosirea unor celule asemănătoare în funcționare cu **celulele corticale simple, senzitive la orientare şi contrast**:

$$s_k^{ld} = [R*S_k^{ld}] \text{ şi } s_k^{dl} = [R*S_k^{dl}] \tag{5.3.7.4}$$

în care cu s_k^{ld} respectiv cu s_k^{dl} s-a notat răspunsul celulelor simple la tranziții luminos-întunecat respectiv întunecat-luminos. Filtrele S sunt obținute ca diferențe de gaussieni de forma:

$$G = \frac{1}{\sigma_x \sigma_y \sqrt{2\pi}} \exp\left(-\frac{(x-i)^2}{\sigma_x^{\,2}} + \frac{(y-i)^2}{\sigma_y^{\,2}} \right) \qquad (5.3.7.5)$$

unde $\sigma_x = 0{,}5$ şi $\sigma_y = 1{,}5$.

Etapa a III-a presupune folosirea celulelor complexe, cu răspuns insenzitiv la contrast:

$$C = \sum_k s_k^{\,ld} + \sum_k s_k^{\,dl} \qquad (5.3.7.6)$$

Se menţionează că sunt considerate răspunsurile celulelor complexe pentru 4 rezoluţii diferite.

Reţeaua neuronală folosită la clasificare reprezintă o versiune simplificată a unei reţele fuzzy ART. Ea constă dintr-un strat de intrare, la care se aplică versiunile transformate ale imaginii, conectat printr-un set de ponderi sinaptice la stratul de ieşire.

Sunt prezentate rezultatele experimentale asupra bazei de date ORL. Cea mai bună performanţă obţinută la un anumit nivel al rezoluţiei, folosind 5 tipare de antrenament/clasă, este o eroare de 6,65%.

5.4 Concluzii

În acest capitol s-au prezentat şi analizat: sistemul vizual uman (subcapitolul 5.1) datorită performanţelor deosebite ale acestuia în recunoaştere şi diverse metode, tratate de autorul tezei în [84] şi [85], convenţionale (subcapitolul 5.2) sau neuronale (subcapitolul 5.3), dedicate recunoaşterii faciale.

Principala concluzie desprinsă din subcapitolul 5.1 se referă la prezenţa unei structuri ierarhizate pentru sistemul vizual uman (SVU) care efectuează operaţii de nivel scăzut adică preprocesarea datelor (retina, nucleul lateral geniculat) precum şi operaţii de nivel înalt (cortexul vizual primar şi regiuni corticale înalte) responsabile de analiza culorii , formelor şi a relaţiilor spaţiale dintre obiecte. La acest nivel se situează şi neuronii specializaţi în detecţia şi recunoaşterea facială.

În consecinţă, un sistem artificial de recunoaştere facială ar trebui să efectueze

aceleaşi tipuri de operaţii ca şi SVU.

Principalele concluzii desprinse din analiza metodelor convenţionale (subcapitolul 5.2) folosite în recunoaşterea facială sunt:

- Metode bazate pe trăsături geometrice. Prezintă rezultate bune în condiţiile unor baze de date largi dar la care iluminarea şi perspectiva sunt relativ constante. Algoritmii actuali de localizare automată a punctelor de interes nu oferă o acurateţe deosebită şi de aici poate să rezulte o depreciere a performanţelor globale.
- Metode bazate pe analiza componentelor principale. Dezavantajul major constă în necesitatea existenţei unui grad înalt de corelaţie între intensităţile pixelilor imaginilor de test şi de antrenament. Uneori acesta poate fi surmontat printr-o preprocesare extensivă a datelor.
- Metode bazate pe compararea şabloanelor. Pot fi folosite doar în cazul în care imaginile de test au aceeaşi scară, orientare şi iluminare cu imaginile din setul de antrenament.
- Metode bazate pe compararea reţelelor elastice. Avantajul principal este determinat de rezultatele bune obţinute în cazul în care există variaţii ale poziţiei şi expresiei faciale. Ca şi dezavantaj poate fi menţionat complexitatea calculelor necesare implementării metodei.

Principalele concluzii desprinse din analiza metodelor bazate pe reţele neuronale (subcapitolul 5.3) folosite în recunoaşterea facială sunt:

- Metodă bazată pe folosirea reţelelor neuronale cu autoorganizare şi convoluţionale. Reţelele neuronale cu autoorganizare se dovedesc a fi o soluţie viabilă pentru faza de extragere a trăsăturilor, fiind uşor superioare metodei clasice de extragere a trăsăturilor bazată pe analiza componentelor principale (PCA, Principal Component Analysis). Această afirmaţie e susţinută şi prin rezultatele experimentale prezentate de I.S. Lawrence şi colectiv în [75]. RNA tip convoluţional, datorită structurii intrinseci, rezolvă atât faza de extragere a

trăsăturilor cât şi cea de clasificare. Dezavantajele acestui tip de metode sunt legate de timpii de antrenament excesiv de lungi.

- Metoda bazată pe reţele neuronale cu decizie probabilistică. Remarcabil este faptul referitor la posibilitatea de implementare a operaţiilor de detecţie dar şi de recunoaştere facială cu un singur tip de RNA, cea cu decizie probabilistică. Modularitatea acestei soluţii se poate transforma însă şi în dezavantaj dacă numărul de clase creşte deoarece pentru fiecare clasă este alocată câte o reţea neuronală PDBNN.
- Metode bazate pe arbori neuronali. În acest cazul numărul neuronilor RNA nu trebuie specificat apriori, ceea ce constituie un avantaj faţă de majoritatea tipurilor de RNA. Arborele neuronal se dezvoltă pe măsură ce învaţă şi are exact numărul de neuroni necesar clasificării.
- Metode ce folosesc reţele neuronale bazate pe funcţii radiale. În cazurile prezentate în §5.3.4 sunt raportate rate excelente de clasificare pentru acest tip de RNA. Autorul tezei consideră însă că aceste rezultate se datorează în bună măsură constrângerilor aplicate imaginilor (în principal iluminare şi expresie facială) preprocesate şi modului extrem de riguros în care aceste imagini faciale au fost achiziţionate.
- Metode bazate pe reţele neuronale de tip perceptron multistrat (MLP) şi cu învăţare prin cuantizare vectorială (LVQ). RNA tip MLP au fost folosite extensiv în cvasitotalitatea problemelor ce se pretează la o rezolvare bazată pe calcul neuronal. Acest fapt se datorează robusteţii acestei soluţii şi existenţei unui număr mare de algoritmi de antrenament sofisticaţi care dau rezultate foarte bune. Opinia autorului tezei este că RNA-MLP se constituie într-o soluţie viabilă pentru problema recunoaşterii faciale dacă e folosită în conjuncţie cu un algoritm potrivit pentru extragerea trăsăturilor. RNA-LVQ, fiind derivată din reţelele neuronale de tip Kohonen, oferă performanţe bune şi fără o etapă anterioară de extragere de trăsături.
- Metode bazate pe reţele neuronale tip grupuri de arbori cu toleranţă adaptivă.

Oferă o acurateţe deosebită a clasificării dar având în vedere faptul că în cazul unor astfel de arbori fiecare nod corespunde la o categorie separată, complexitatea calculelor va creşte simţitor dacă numărul de clase creşte.

- Metode bazate pe reţele neuronale ART. RNA ART prezintă avantajul extrem de important al faptului că adăugarea unei clase nu implică reantrenarea RNA. În problema recunoaşterii faciale ele nu pot fi folosite însă decât în măsura în care vectorii destinaţi clasificării sunt furnizaţi de către un algoritm corespunzător de extragere a trăsăturilor.

Concluzia finală desprinsă din acest capitol este că soluţia optimă pentru un sistem de recunoaştere facială trebuie să cuprindă obligatoriu o etapă de extragere a trăsăturilor şi una de clasificare. S-a constatat că se obţin performanţe superioare dacă aceste etape sunt distincte deşi numeroase tipuri de RNA realizează implicit ambele operaţii. Autorul tezei consideră utilă folosirea unei metode convenţionale în partea de preprocesare şi extragere de trăsături datorită vitezei superioare soluţiilor neuronale. În aspectul luării deciziilor, al recunoaşterii propriu-zise, este recomandată folosirea unui sistem neuronal datorită performanţelor în clasificare superioare metodelor clasice şi datorită avantajelor discutate în subcapitolul 3.1.

CAPITOLUL 6

Propuneri privind realizarea unui sistem neuronal de recunoaştere facială

6.1 Propuneri privind arhitectura şi metodele de implementare pentru un sistem de recunoaştere facială

Pe baza analizei metodelor specifice recunoaşterii faciale (RF) prezentate în capitolul anterior se desprind următoarele etape (fig.6.1.1) aferente unui proces de recunoaştere facială:

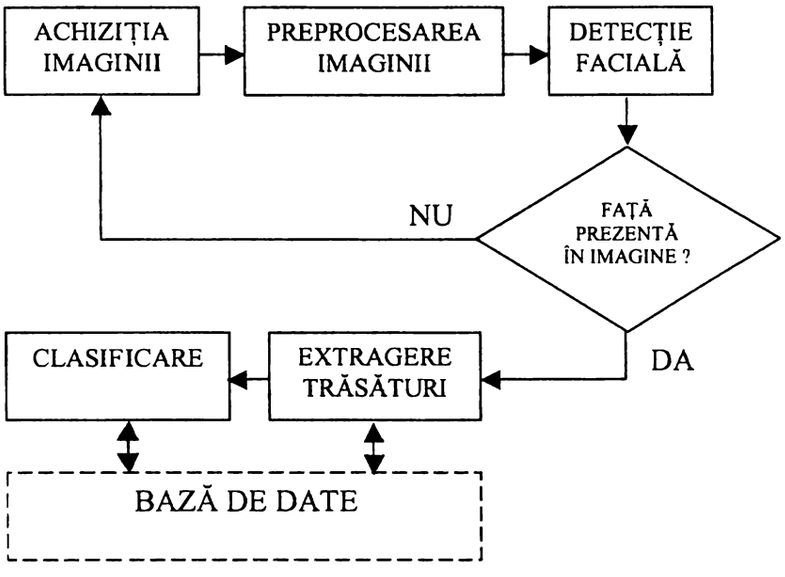


Fig. 6.1.1. O posibilă arhitectură pentru un sistem de recunoaştere facială, prezentată de autorul tezei în [85].

6.1.1 Achiziţia imaginii

Pentru prelevarea imaginilor în care se presupune că ar exista feţe umane se folosesc dispozitive videocaptoare cum ar fi camerele video însoţite de o placă videocaptoare, pentru aplicaţii care necesită rezoluţii şi rate de refresh ridicate sau aşa-numitele WEBCAM-uri folosite în cazul videoconferinţelor desfăşurate prin intermediul INTERNET-ului, care oferă rezoluţii şi rate de refresh mai scăzute, sunt conectate direct la porturile paralel sau USB şi se constituie într-o soluţie excelentă pentru aplicaţiile RF sub aspectul raportului preţ/performanţă.

Sunt menţionate în literatură (Hang şi Join [86]) şi cazuri în care RF este posibilă prin intermediul imaginilor în infraroşu.

Pentru cazul aplicaţiei concrete s-a folosit produsul firmei Creative, Video Blaster WEBCAM 3 USB (fig.6.1.1.1) care oferă posibilitatea de videocaptură a imaginilor "live" la o adâncime a culorii de 16,7 milioane, 30 cadre/s la rezoluţia de 352x288 pixeli sau 15 cadre/s la 640x480 pixeli.

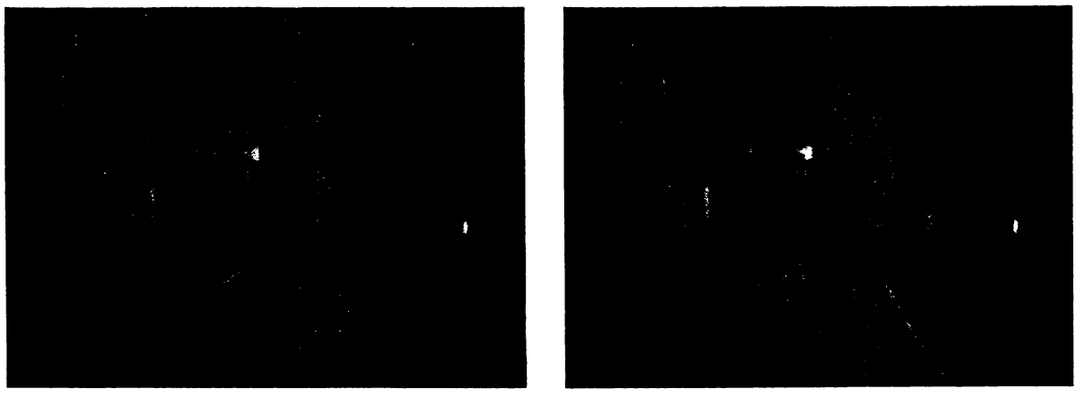
Fig.6.1.1.1 Video Blaster WEBCAM 3 USB folosită la implementarea sistemului neuronal de recunoaştere facială propus de autorul tezei.

6.1.2 Preprocesarea imaginii

Imaginea prelevată este supusă anumitor prelucrări în scopul îmbunătăţirii performanţelor blocurilor care succed etapei curente. În mod tipic sunt folosite următoarele tehnici de preprocesare:

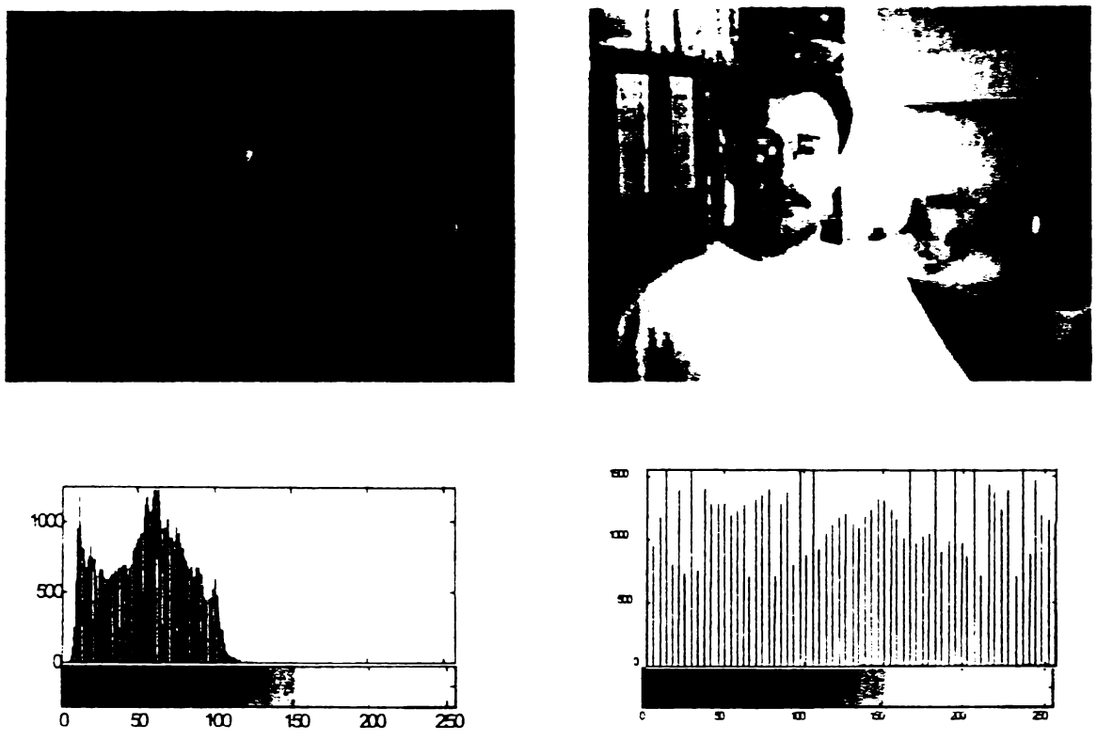
- pentru cazul unei iluminări slabe sau cu variaţii se foloseşte o compensare liniară care calculează media iluminării urmând ca aceasta să fie scăzută din elementele unei ferestre;
- egalizarea histogramei în scopul îmbunătăţirii contrastului imaginii;
- netezirea imaginii, în cazul în care aceasta este contaminată cu zgomot.

Pentru cazul aplicaţiei concrete imaginea originală în format 320x240 RGB a fost supusă transformării în tonuri de gri (fig.6.1.2.1). S-a constatat experimental, de către autorul tezei, că operaţia de egalizare a histogramei (fig.6.1.2.2) nu aduce avantaje majore în procesul de recunoaştere facială.



a) RGB, 16mil. culori b) 256 nivele de gri

Fig. 6.1.2.1. Conversia imaginii RGB – tonuri de gri



a) Imagine şi histogramă originală

b) Imagine şi histogramă modificată

Fig. 6.1.2.2 Egalizarea histogramei îmbunătăţeşte contrastul imaginii

6.1.3 Detecţia feţei

Reprezintă o etapă importantă în ansamblul operaţiilor dintr-un sistem de recunoaştere facială întrucât influenţează decisiv calitatea procesului de recunoaştere şi clasificare. Scopul acestei etape îl reprezintă identificarea regiunilor dintr-o anumită imagine care conţin feţe umane.

La implementarea detecţiei feţei s-au avut în vedere două metode, descrise în cele ce urmează:

- **Detecţia feţei bazată pe reţele neuronale**

Metoda prezentată în continuare a fost propusă de Rowley, Baluja şi Kanade

[87] şi constă într-o etapă preliminară de preprocesare a imaginii (vezi §6.1.2) aplicată unor ferestre de dimensiune 20x20 pixeli. Aceste ferestre preprocesate sunt ulterior aplicate unui grup de reţele neuronale (fig.6.1.3.1).

Se observă trei tipuri de neuroni ascunşi: 4 neuroni urmăresc subregiuni de dimensiuni 10x10 pixeli, 16 prelucrează subregiuni de 5x5 pixeli şi 6 urmăresc regiuni suprapuse de 20x5 pixeli. Fiecare dintre aceste tipuri au fost alese astfel încât să permită unităţilor ascunse să reprezinte trăsături localizate care ar putea prezenta importanţă în detecţia facială.

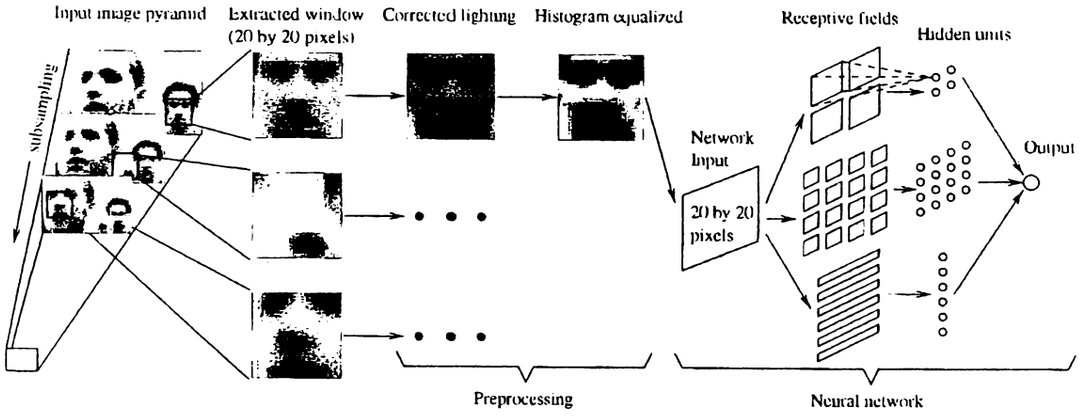


Fig. 6.1.3.1 Sistem neuronal de detecţie facială, cf. [87]

Merită subliniată procedura de antrenament folosită în cazul imaginilor care nu reprezintă feţe, dat fiind faptul că acoperirea tuturor imaginilor care nu reprezintă feţe este practic imposibilă. Autorii folosesc o modalitate adaptată după Sung şi Poggio [88]:

- Se creează un set de 1000 de imagini non-feţe cu valori aleatoare ale intensităţilor pixelilor.
- Se aplică metoda de preprocesare pentru fiecare dintre aceste imagini.
- Se antrenează reţeaua neuronală pentru a produce o ieşire egală cu "1" pentru imagini ce reprezintă feţe şi "-1" pentru exemplele ce reprezintă non-feţe.
- Reţeaua antrenată este folosită pentru explorarea unor imagini care nu conţin feţe umane. Se selectează toate subimaginile identificate în mod greşit drept feţe.

- Se selectează un număr de subimagini identificate greşit la pasul anterior, se aplică metodologia de preprocesare şi se adaugă în mulţimea imaginilor de antrenament care nu conţin feţe.

În fig. 6.1.3.2 este prezentat rezultatul algoritmului descris anterior.

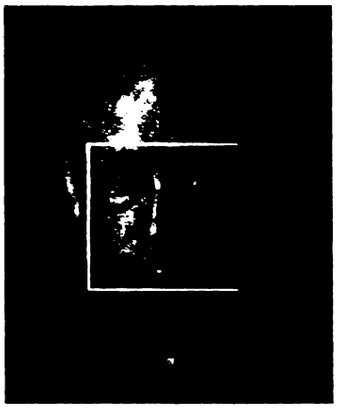


Fig.6.1.3.2 Rezultatul rulării algoritmului de detecţie facială bazat pe reţele neuronale.

În urma analizei funcţionării algoritmului s-au constatat următoarele *dezavantaje*:

- în primul rând ***timpul*** consacrat fazei de antrenament este considerabil, mai ales în cazul în care există un număr mare de imagini non-feţe.
- numărul de ***imagini non-feţe*** trebuie să fie foarte mare;
- în al treilea rând feţele nu sunt dreptunghiulare ci mai degrabă alungite şi deci o atare transformare conduce la anumite deformări ale imaginii;
- în ultimul rând dimensiunea de 20x20 pixeli a ferestrei care baleiază întreaga imagine e destul de mică şi deci faţa este greu identificabilă.

- **Detecţia facială bazată pe decompoziţie subspaţială şi principiul similarităţii maxime**

Metoda propusă de Moghaddam şi Petland [89] estimează distribuţia de probabilitate a unui obiect folosind decompoziţia (descompunerea) spaţiului imaginii bazată pe metoda vectorilor proprii. Densitatea dorită este descompusă în două componente, cea din subspaţiul principal şi cea din subspaţiul complementar ortogonal (fig. 6.1.3.3).

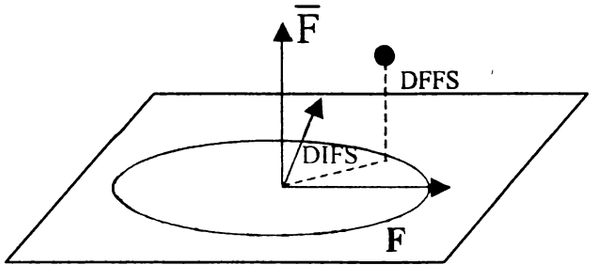


Fig. 6.1.3.3 Descompunerea în subspaţiul principal F şi cel complementar ortogonal $\bar{F}$ pentru o distribuţie gaussiană, cf. [89].

Concret, fiind dată o mulţime de imagini de antrenament $\{\mathbf{x}'\}_{t=1}^{N_T}$ aparţinătoare clasei obiectului Ω se doreşte estimarea apartenenţei clasei sau a funcţiei de similaritate ("likehood function"), adică $P(\mathbf{x}|\ \Omega)$. Pentru aceasta se parcurg următoarele etape:

a) Descompunerea în vectori proprii

Problema valorilor proprii poate fi ilustrată prin ecuaţia:

$$\mathbf{\Lambda} = \mathbf{\Phi}^T \mathbf{\Sigma} \mathbf{\Phi} \qquad (6.1.3.1)$$

în care $\mathbf{\Sigma}$ reprezintă matricea de covarianţă, $\mathbf{\Phi}$ matricea vectorilor proprii şi $\mathbf{\Lambda}$ matricea diagonală a valorilor proprii. În analiza componentelor principale (PCA) se efectuează o transformare Karhunen – Loeve (KLT) care identifică vectorii proprii cu valorile proprii cele mai mari şi se obţin vectorii trăsătură $\mathbf{y} = \mathbf{\Phi}_M^T \tilde{\mathbf{x}}$, unde $\tilde{\mathbf{x}} = \mathbf{x} - \bar{\mathbf{x}}$ este vectorul imagine normalizat, iar $\mathbf{\Phi}_M$ o submatrice a $\mathbf{\Phi}$ conţinând vectorii proprii principali.

Prin selectarea primelor M componente principale se descompune ortogonal spaţiul vectorilor R^N în două subspaţii mutual exclusive şi complementare $\mathbf{F} = \{\mathbf{\Phi}_i\}_{i=1}^{M}$

şi $\overline{\mathbf{F}} = \{\Phi_i\}_{i=M+1}^{N}$.

Se poate defini eroarea de reconstrucţie reziduală:

$$\varepsilon^2(\mathbf{x}) = \sum_{i=M+1}^{N} y_i^2 = \|\tilde{\mathbf{x}}\|^2 - \sum_{i=1}^{M} y_i^2 \qquad (6.1.3.2)$$

În acest caz componenta din spaţiul ortogonal $\overline{\mathbf{F}}$ va fi asimilată cu o *distanţă de la spaţiul trăsăturilor* (DFFS, "Distance From Feature Space") şi este echivalentă cu eroarea reziduală $\varepsilon^2(\mathbf{x})$. Componenta lui **x** aparţinătoare spaţiului trăsăturilor **F** va fi referită prin *distanţa în spaţiul trăsăturilor* (DIFS, "Distance In Feature Space").

b) Cazul distribuţiei gaussiene

În acest caz similaritatea unui tipar **x** este dată de:

$$P(\mathbf{x}\mid\Omega) = \frac{\exp[-\frac{1}{2}(\mathbf{x}-\bar{\mathbf{x}})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\bar{\mathbf{x}})]}{(2\pi)^{N/2}\mid\boldsymbol{\Sigma}\mid^{1/2}} \qquad (6.1.3.3)$$

Pentru caracterizarea acestei similarităţi se foloseşte distanţa Mahalanobis:

$$d(\mathbf{x}) = \tilde{\mathbf{x}}^T\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{x}} \qquad (6.1.3.4)$$

care se mai poate scrie:

$$d(\mathbf{x}) = \tilde{\mathbf{x}}^T[\boldsymbol{\Phi}\boldsymbol{\Lambda}^{-1}\boldsymbol{\Phi}^T]\tilde{\mathbf{x}} = \mathbf{y}^T\boldsymbol{\Lambda}^{-1}\mathbf{y} \qquad (6.1.3.5)$$

Evaluarea (6.1.3.5) este neeficientă şi se încearcă *estimarea* $d(\mathbf{x})$ folosind doar M proiecţii:

$$d(\mathbf{x}) = \sum_{i=1}^{M}\frac{y_i^2}{\lambda_i} + \sum_{i=M+1}^{N}\frac{y_i^2}{\lambda_i} = \sum_{i=1}^{M}\frac{y_i^2}{\lambda_i} + \frac{1}{\rho}\left[\sum_{i=M+1}^{N} y_i^2\right] = \sum_{i=1}^{M}\frac{y_i^2}{\lambda_i} + \frac{\varepsilon^2(\mathbf{x})}{\rho} \qquad (6.1.3.6)$$

În consecintă se poate scrie forma ***similarităţii estimate*** bazată pe $\hat{d}(\mathbf{x})$ ca fiind produsul unor densităţi gaussiene independente:

$$\hat{P}(\mathbf{x}\mid\Omega) = \left[\frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{M}\frac{y_i^2}{\lambda_i}\right)}{(2\pi)^{M/2}\prod_{i=1}^{M}\lambda_i^{1/2}}\right]\cdot\left[\frac{\exp\left(-\frac{\varepsilon^2(\mathbf{x})}{2\rho}\right)}{(2\pi\rho)^{(N-M)/2}}\right] = P_F(\mathbf{x}\mid\Omega)\hat{P}_{\bar{F}}(\mathbf{x}\mid\Omega) \qquad (6.1.3.7)$$

Valoarea optimă a valorii medii a valorilor proprii nefolosite se determină prin

minimizarea unei funcţii de cost $J(\rho)$:

$$J(\rho) = \int P(\mathbf{x} \mid \Omega) \log \frac{P(\mathbf{x} \mid \Omega)}{\hat{P}(\mathbf{x} \mid \Omega)} d\mathbf{x} = E\left[\log \frac{P(\mathbf{x} \mid \Omega)}{\hat{P}(\mathbf{x} \mid \Omega)} \right]$$

$$= \frac{1}{2} \sum_{i=M+1}^{N} \left[\frac{\lambda_i}{\rho} - 1 + \log \frac{\rho}{\lambda_i} \right] \qquad (6.1.3.8)$$

Valoarea optimă ρ* se obţine rezolvând ecuaţia $\frac{\partial J}{\partial \rho} = 0$. Rezultă:

$$\rho^* = \frac{1}{N - M} \sum_{i=M+1}^{N} \lambda_i \qquad (6.1.3.9)$$

c) Cazul unei distribuţii multimodale

În cazul în care mulţimea datelor de antrenament reprezintă imagini sau obiecte multiple în condiţii de iluminare variabilă distribuţia imaginilor în spaţiul **F** nu mai este unimodală (fig.6.1.3.4).

Densităţile complexe se pot modela folosind o combinaţie de gaussieni:

$$P(y \mid \Theta) = \sum_{i=1}^{N_c} \pi_i g(y; \mu_i, \Sigma_i) \qquad (6.1.3.10)$$

în care $g(y; \mu_i, \Sigma_i)$ reprezintă o densitate gaussiană M-dimensională cu media μ şi covarianţa Σ.

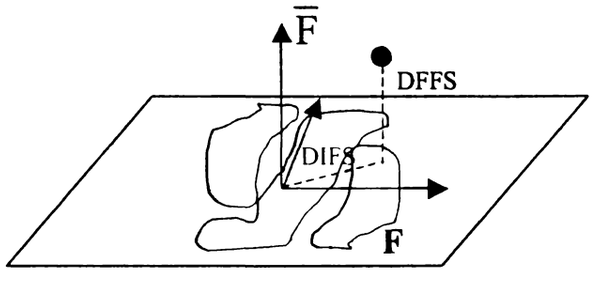


Fig. 6.1.3.4 Descompunerea în subspaţiul principal **F** şi cel complementar ortogonal $\bar{F}$ pentru o distribuţie arbitrară, cf. [89].

Combinaţia este specificată complet de către parametrul $\Theta = \{\pi_i, \mu_i, \Sigma_i\}_{i=1}^{N_c}$.

Pe baza detecţiei similarităţii maxime autorii recomandă folosirea acestei metode nu doar în cazul detecţiei faciale ci şi în cazul detecţiei şi recunoaşterii obiectelor în general.

În urma implementării de către autorul tezei a metodei descrise anterior se constată performanţe foarte bune în acurateţea detecţiei feţei umane în imagine (fig. 6.1.3.5).

Fig.6.1.3.5 Detecţie facială pe baza principiului similarităţii maxime

Totuşi, s-au constatat şi anumite ***dezavantaje***:

- algoritmul nu foloseşte în nici un fel ***proprietăţile geometrice ale feţei*** (simetrie, poziţia relativă a ochilor, nasului etc.);
- algoritmul nu poate detecta feţe ***mai mici decât faţa medie*** decât cu preţul scăderii drastice a vitezei;
- performanţa algoritmului depinde de ***raportul de aspect*** al feţelor prezente în imagine;
- în cele mai multe cazuri ***DFFS*** conduce la îmbunătăţirea performanţelor, dar ***nu întotdeauna.***

Comparând performanţele (acurateţe şi viteză) metodei neuronale de detecţie a feţei cu cele ale metodei cu descompunerea subspaţială s-a optat pentru folosirea celei de-a doua metode în implementarea sistemului de detecţie şi recunoaştere facială prezentat în cadrul acestei teze.

6.1.4 Extragerea trăsăturilor

Reprezintă o etapă deosebit de importantă în cadrul unui sistem de recunoaştere facială întrucât prin intermediul acesteia se ***extrage informaţia relevantă***, caracteristică pentru o anumită persoană. Totodată această etapă asigură şi o ***reducere a dimensionalităţii*** datelor de intrare aplicate ulterior clasificatorului.

Aşa cum s-a constatat din capitolul anterior, există numeroase metode de extragere a trăsăturilor, atât convenţionale (analiza componentelor principale, filtrare Gabor) cât şi neuronale (reţele neuronale cu extractori ierarhici de trăsături, RNA cu autoorganizare tip hartă de trăsături, RNA cu autoasociere, filtre inspirate din modelul biologic al căii vizuale).

Pentru cazul concret al implementării sistemului de recunoaştere facială s-a ales o modalitate convenţională de extragere a trăsăturilor, ***metoda operatorului de interes***, inspirată din cea prezentată de către Moravec în [90] pentru cazul navigaţiei roboţilor autonomi. Ideea a fost preluată ulterior de către Nasrabadi şi Choo şi pentru cazul vederii stereoscopice în conjuncţie cu RNA de tip Hopfield [91] sau, de către Wang şi colectiv, pentru detecţia automată a ţintelor [92].

S-a arătat de către autorul tezei [93] faptul că această metodă poate fi folosită cu succes şi în ***cazul recunoaşterii faciale*** fiind după cunoştinţa autorului tezei, şi ***unica abordare*** de acest fel prezentată în literatura recunoaşterii faciale.

Procedura constă în extragerea unor variante direcţionale (orizontal, vertical şi ambele diagonale) pentru fiecare bloc dintr-o imagine facială. Aceste variante arată mărimea activităţii locale dintr-un bloc al imaginii. Regiunile cu varianţă ridicată individualizează cu succes tipologiile prezente în imaginile faciale.

Ecuaţiile pentru calculul varianţei direcţionale sunt următoarele:

$$\sigma^2 = \frac{1}{P \times Q} \sum_{y=0}^{P-1} \sum_{x=0}^{Q-1} [p(x,y) - \mu]^2 \tag{6.1.4.1}$$

$$\sigma^2{}_H = \frac{1}{P \times Q} \sum_{y=0}^{P-1} \sum_{x=0}^{Q-1} [p(y,x+1) - p(y,x)]^2 \tag{6.1.4.2}$$

$$\sigma^2{}_V = \frac{1}{P \times Q} \sum_{y=0}^{P-1} \sum_{x=0}^{Q-1} [p(y+1,x) - p(y,x)]^2 \tag{6.1.4.3}$$

$$\sigma^2{}_{135} = \frac{1}{P \times Q} \sum_{y=0}^{P-1} \sum_{x=0}^{Q-1} [p(y+1,x+1) - p(y,x)]^2 \tag{6.1.4.4}$$

$$\sigma^2{}_{45} = \frac{1}{P \times Q} \sum_{y=0}^{P-1} \sum_{x=0}^{Q-1} [p(y+1,x) - p(y,x+1)]^2 \tag{6.1.4.5}$$

în care μ reprezintă media datelor şi este calculată cu formula:

$$\mu = \frac{1}{P \times Q} \sum_{y=0}^{P-1} \sum_{x=0}^{Q-1} p(y,x) \tag{6.1.4.6}$$

şi $\{p(y,x),\ 0 \le y \le P\text{-}1,\ 0 \le x \le Q\text{-}1\}$ reprezintă intensitatea pixelilor într-un bloc P x Q, cf. fig.6.1.4.1.

Se vor obţine astfel pentru fiecare bloc dintr-o regiune a imaginii un set de cinci mărimi: $\theta_i = \{\sigma, \sigma_H, \sigma_V, \sigma_{45}, \sigma_{135}\}_i$, i = 1...L, L fiind numărul total de blocuri distincte în care se împarte o regiune a unei imagini.

Considerând imaginea originală de dimensiune N x M şi fereastra P x Q din care se extrage vectorul trăsătură se poate deduce valoarea reducerii dimensionalităţii (RD) datelor aplicate ulterior clasificatorului:

$$RD \cong \frac{N \times M}{5 \times \frac{N}{P} \times \frac{M}{Q}} = \frac{P \times Q}{5} \tag{6.1.4.7}$$

Deşi acest factor depinde de dimensiunea ferestrei din care se extrag trăsăturile, pentru cazul concret descris în această lucrare (P=Q=8) se obţine o ***reducere de 12 ori*** a volumului de date care urmează a fi prelucrată de clasificatorul neuronal descris în paragraful următor, reţinându-se totodată şi elementele caracteristice dintr-o anumită

imagine. Efectul aplicării operatorului de interes asupra unei imagini poate fi urmărit în fig.6.1.4.2.

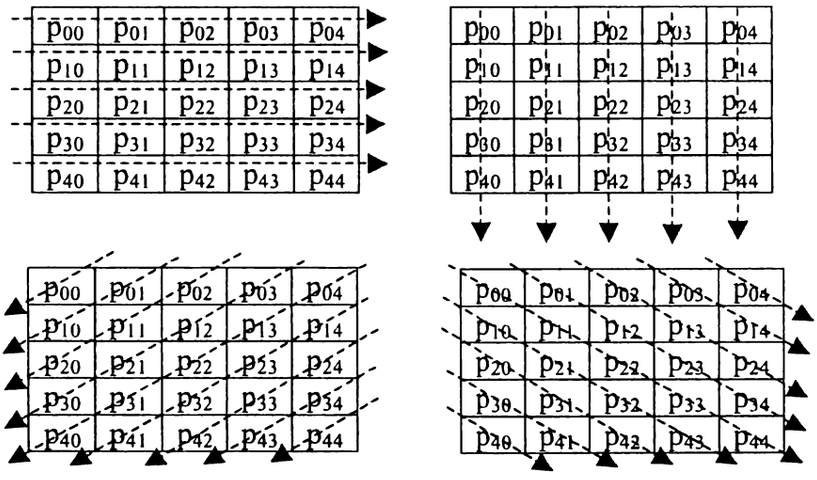
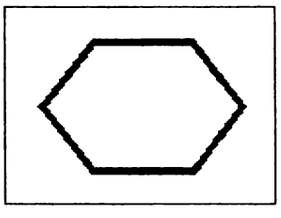


Fig.6.1.4.1 Extragerea variantelor direcţionale dintr-o imagine pe baza meteodei operatorului de interes



a) Imaginea originală

b) σ c) σ_H d) σ_V e) σ_{135} f) σ_{45}

Fig.6.1.4.2 Efectul extragerii variantelor direcţionale dintr-o imagine pe baza metodei operatorului de interes

6.1.5 Clasificarea

Reprezintă cea mai importantă etapă a procesului de recunoaştere facială şi se concretizează prin atribuirea unei clase (nume, etichetă) pentru fiecare imagine facială reprezentată de către vectorul trăsăturilor.

La implementarea acestei etape s-a apelat la o soluţie neuronală, într-o arhitectură bazată pe o RNA tip perceptron multistrat, total conectată (fig.6.1.5.1), cu un singur strat ascuns.

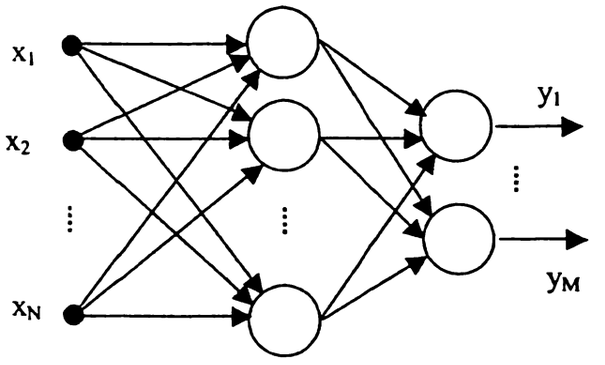
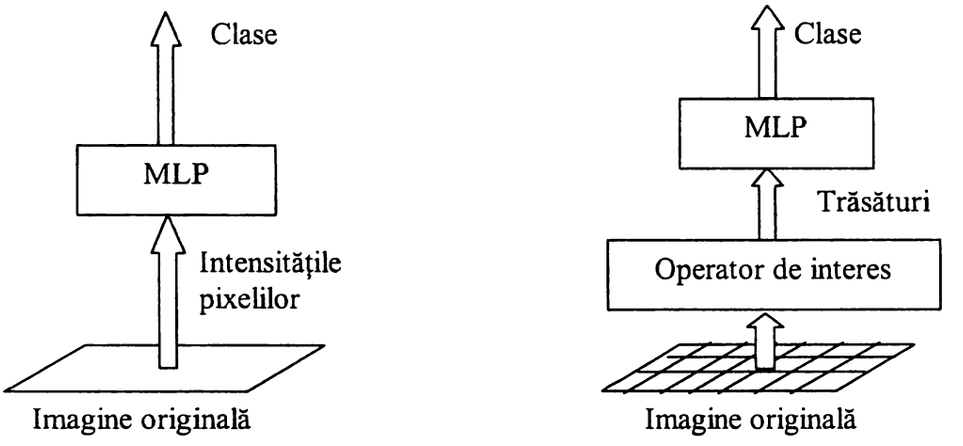


Fig.6.1.5.1 RNA tip perceptron multistrat, cu un singur strat ascuns, total conectată

Argumentele principale care au stat la baza acestei alegeri , discutate şi în capitolul anterior, au fost în primul rând viteza superioară oferită de către acest tip de clasificator - condiţie obligatorie pentru realizarea unui sistem care să funcţioneze în timp real. La aceasta se alătură şi acurateţea clasificării oferită de către acest tip de RNA.

Contribuţia originală a autorului s-a manifestat pe de-o parte în domeniul soluţiei arhitecturale alese pentru ansamblul extragere trăsături + clasificare dar şi în ceea ce priveşte algoritmii de antrenament pentru RNA-MLP. În mod concret, au fost propuşi 2 algoritmi de antrenament, prezentaţi de autorul tezei în [44] ***şi*** [45] ***şi 4 arhitecturi ale ansamblului extragere trăsături + clasificator prezentate de autorul tezei în*** [94] ***şi*** [95].

Primele două arhitecturi folosesc o singură RNA-MLP, la intrările acesteia aplicându-se în primul caz direct intensităţile pixelilor imaginii , iar în al doilea caz, trăsăturile extrase prin metoda operatorului de interes prezentată detaliat în paragraful anterior. Cele două arhitecturi pot fi urmărite în fig.6.1.5.2.a şi b.

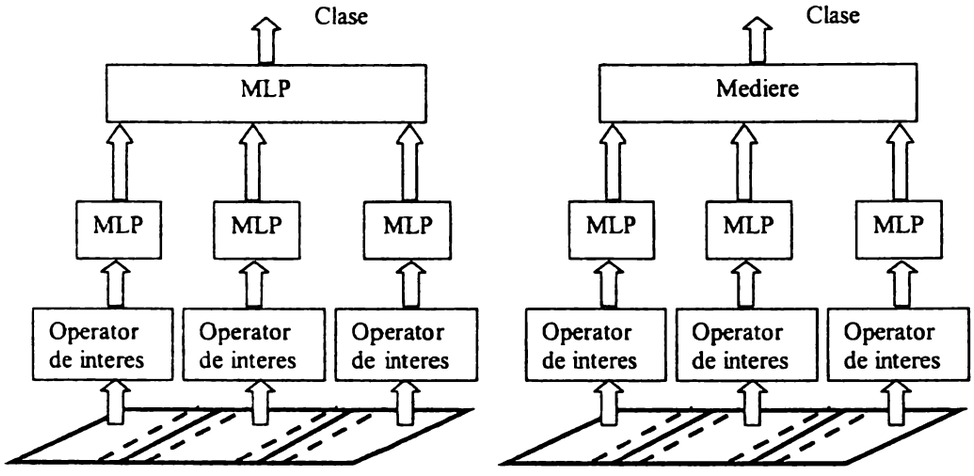


a) Caz I. Intensităţile pixelilor aplicate direct RNA-MLP

b) Caz II. Trăsături extrase prin metoda operatorului de interes

Fig.6.1.5.2 Arhitectură de sistem pentru RF cu un singur clasificator

Următoarele două abordări folosesc o arhitectură paralelă de RNA-MLP la a căror intrări se aplică trăsături extrase prin metoda operatorului de interes din regiuni parţial/total suprapuse sau nesuprapuse (fig.6.1.5.3 a şi b). Diferenţa dintre aceste ultime două cazuri constă în modalitatea de luare a deciziei în ceea ce priveşte clasificarea finală a imaginii faciale prezentate. În cazul III răspunsurile RNA-MLP sunt prelucrate de o RNA-MLP ierarhic superioară, iar în cazul IV clasificarea este efectuată pe baza medierii răspunsurilor RNA-MLP ierarhic inferioare.



a) Caz III. Decizia finală este luată de către o RNA-MLP ierarhic superioară

b) Caz IV. Decizia finală luată pe baza medierii răspunsurilor RNA ierarhic inferioare

Fig.6.1.5.3 Arhitectură de sistem pentru RF cu structură ierarhică şi paralelă de clasificatori.

6.2 Rezultate experimentale

6.2.1 Descrierea bazei de date

Experimentele au fost efectuate asupra bazei de date ORL (**O**livetti **R**esearch **L**aboratory, actual în proprietatea AT&T U.K.) care conține un set de imagini faciale prelevate între Aprilie 1992 şi Aprilie 1994 de către ORL în Cambridge, UK [55]. Baza de date constă din 10 imagini diferite pentru 40 de subiecți distincți, deci un total de 400 de imagini. Imaginile au fost luate în intervale de timp diferite, în condiții de variație a iluminării, expresiei faciale (ochi deschişi/închişi, zâmbet, cu/fără ochelari). Toate imaginile conțin un fundal omogen, întunecat. Mărimea imaginilor este de 92x112 pixeli, 256 niveluri de gri (fig. 6.2.1.1).

În toate cazurile în care nu apare în mod explicit altfel, numărul de imagini/persoană corespunzătoare etapei de antrenament este egal cu 5 şi deci numărul de imagini/persoană consacrat etapei de test va fi tot 5. Numărul de clase (persoane distincte) este egal cu 40, în cazul în care nu se specifică altfel. Rezultă deci un număr de 200 de imagini alese în mod aleator, la începutul fiecărui experiment, corespunzătoare etapei de antrenament şi 200 de imagini aleatoare de test, cele două mulțimi fiind disjuncte.

6.2.2 Influența folosirii tehnicii de extragere a trăsăturilor de tip operator de interes asupra procesului de recunoaştere facială

În tab.6.2.2.1 sunt prezentate comparativ performanțele celor două arhitecturi prezentate în fig.6.1.5.2 a şi b. Practic, folosind un singur clasificator tip RNA-MLP, se evaluează cazurile în care imaginea este aplicată direct, fără preprocesare, respectiv cazul în care la intrarea RNA-MLP se aplică un vector de trăsături obținut din imaginea originală prin metoda operatorului de interes. Experimentele au fost efectuate conform metodologiei descrise în §6.2.1; **faza de antrenament s-a considerat încheiată** în

momentul în care eroarea pătratică medie (EPM) a atins valoarea de 0,01%, ceea ce corespunde unei erori nule pentru datele de antrenament .

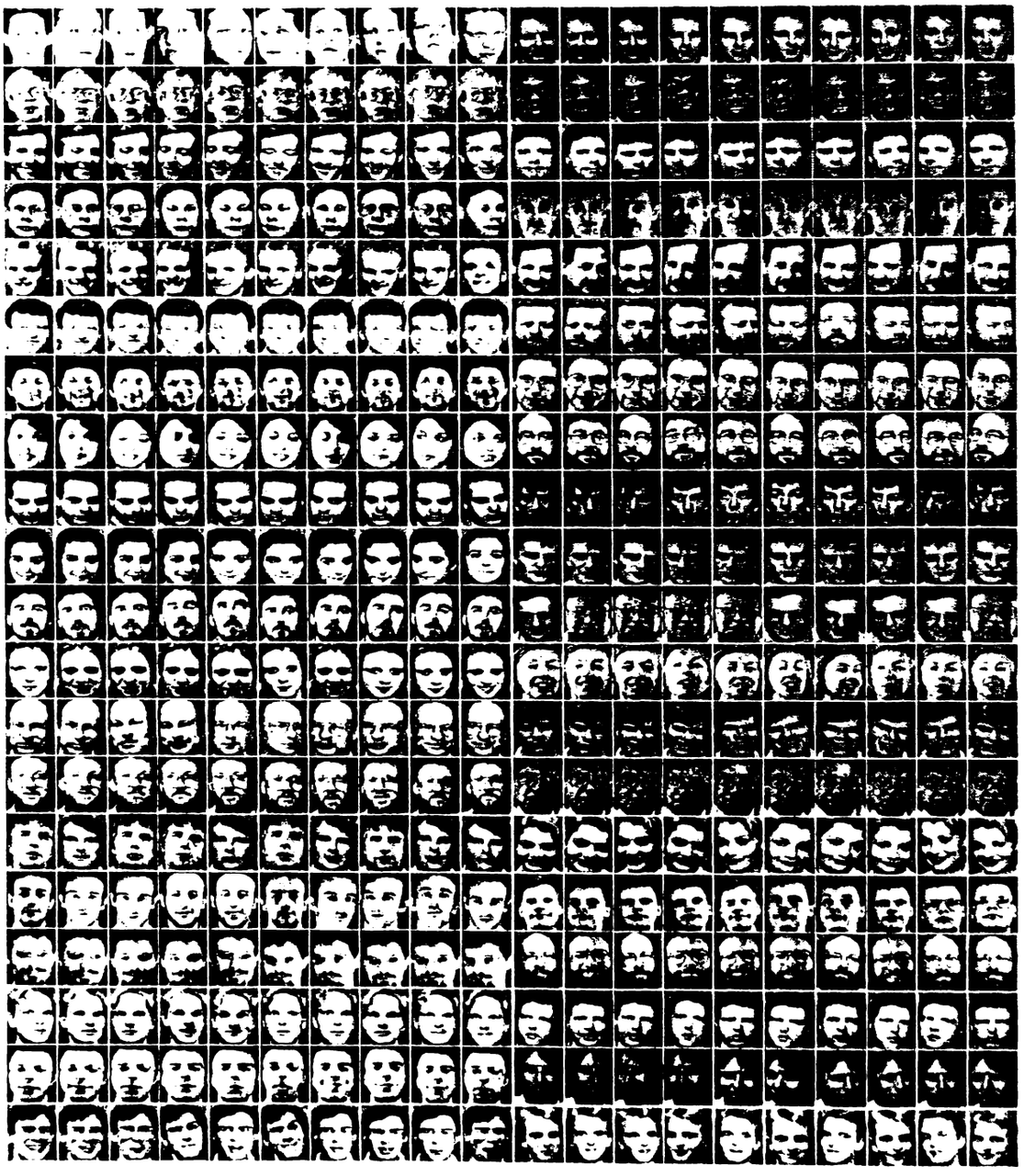


Fig.6.2.1.1 Baza de date folosită pentru testarea performanţelor sistemului de recunoaştere facială [55].

Sistem	Număr neuroni intrare-ascunşi-ieşire	EPM %	1 img./pers.		3 img./pers.		5 img./pers.	
			Eroare test %	Eroare totală %	Eroare test %	Eroare totală %	Eroare test %	Eroare totală %
			Durată Experiment [min.]		Durată Experiment [min.]		Durată Experiment [min.]	
Fig.6.1.5.2 a	2576 – 80 – 40	0.01	55,94	50,35	23,92	16,75	14,80	7,40
			3,56 min.		10,14 min.		16,66 min.	
Fig.6.1.5.2 b	210 – 80 – 40	0.01	41,16	37,05	14,50	10,15	8,80	4,40
			2,25 min.		3,27 min.		4,58 min.	

Tab.6.2.2.1 Rezultate medii pentru 5 experimente consecutive efectuate cu sistemele din fig.6.1.5.2 a şi b, adică un singur clasificator neuronal cu/fără extragere de trăsături prin metoda operatorului de interes. Imaginile originale au fost reeşantionate cu un factor de ½ deci sunt de dimensiune 56x46 pixeli, iar dimensiunea ferestrei din care se extrag trăsăturile este 8x8 pixeli. Mulţimile de antrenament şi test au fost obţinute cf. specificaţiilor din § 6.2.1.

Rezultatele experimentale prezentate în tab.6.2.2.1 reprezintă valori medii calculate în urma efectuării unui număr de 5 experimente. Timpii medii de efectuare a experimentelor rezultă în urma efectuării operaţiilor de generare aleatoare a ordinii persoanelor şi imaginilor aferente ce constituie setul de date de antrenament (40 persoane x 5 imagini = 200 de imagini), extragere trăsături pentru întreaga bază de date, antrenare RNA-MLP, clasificare set de date de antrenament şi test.

Pentru implementarea algoritmilor s-a folosit mediul MATLAB v.5.3, 1999, rulând pe hardware PC INTEL CELERON 533 MHz.

În urma analizei rezultatelor experimentale se desprind următoarele concluzii:

- în urma folosirii sistemului cu extragere de trăsături se constată o ***creştere a performanţei clasificatorului cu aproximativ 40%*** faţă de cazul fără extragere de trăsături (fig.6.2.2.1);
- se constată ***reducerea dimensiunii datelor*** aplicate la intrarea RNA-MLP de aproximativ ***12 ori*** pentru cazul 6.1.5.2 b faţă de 6.1.5.2 a;
- ***timpul mediu/experiment este de 4 ori mai mic*** în cazul al doilea; spre deosebire de primul caz, timpul mediu/experiment pentru sistemul cu extragere de trăsături nu e direct proporţional cu numărul de tipare de antrenament;

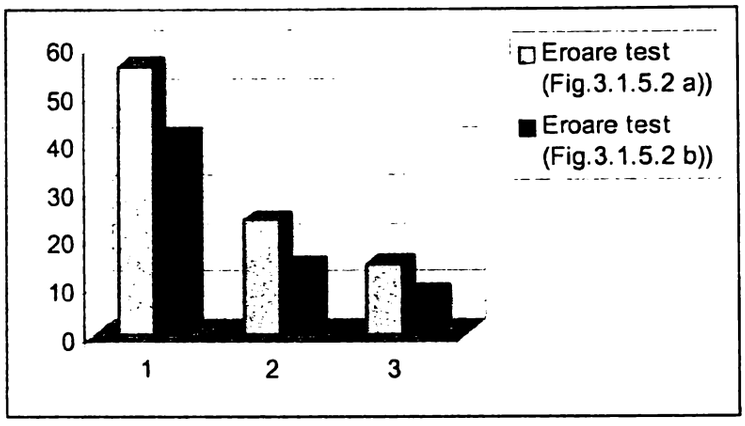


Fig.6.2.2.1 Rezultate comparative pentru cazul sistemelor de recunoaştere facială din fig.6.1.5.2 a şi b.

Concluziile precedente ne îndreptăţesc să considerăm metoda de extragere a trăsăturilor propusă drept deosebit de eficientă, având în vedere rezultatele obţinute în condiţii similare, asupra aceleiaşi baze de date, de către alte sisteme de recunoaştere facială prezentate în literatură [96].

Fişierul sursă aferent realizării acestor experimente este prezentat în ANEXA 3.

6.2.3 Influenţa rezoluţiei imaginii faciale şi a dimensiunilor ferestrei de extragere a trăsăturilor asupra performanţelor procesului de recunoaştere facială

Aşa cum s-a mai menţionat, imaginile bazei de date au dimensiunea de 112x92 pixeli. În cele ce urmează se va urmări evoluţia performanţelor ansamblului extragere trăsături + clasificator neuronal pentru cazurile în care imaginea originală va fi reeşantionată la 75%, 50% respectiv 25% din dimensiunile iniţiale (tab.6.2.3.1) precum şi influenţa dimensiunii ferestrei de extragere a trăsăturilor (tab.6.2.3.2).

Factor de reeşantionare (dimensiune în pixeli)	25% (5x5 pixeli)	50% (56x46 pixeli)	75% (84x69 pixeli)	100% (112x92 pixeli)
Eroare test [%] (dimensiune fereastră)	12,90 (5x5 pixeli)	8,80 (8x8 pixeli)	10,20 (7x10 pixeli)	10,30 (8x8 pixeli)

Tab.6.2.3.1 Influenţa rezoluţiei imaginii asupra performanţelor ansamblului extragere trăsături + clasificator neuronal

Dimensiunea ferestrei, în pixeli	4 x 6	7 x 6	8 x 8	14 x 12
Eroare test [%]	9,30	8,20	8,80	9,80

Tab.6.2.3.2 Influenţa dimensiunii ferestrei de extragere a trăsăturilor asupra performanţelor ansamblului extragere trăsături + clasificator neuronal, dimensiunea imaginii 56x46

Din analiza rezultatelor experimentale prezentate în tab.6.2.3.1 şi 6.2.3.2 se constată următoarele:

- ***performanţele sistemului de recunoaştere facială cresc pe măsură ce rezoluţia imaginilor descreşte, ating un maxim pentru un factor de reeşantionare de 1/2 iar dacă rezoluţia scade în continuare (factor de reeşantionare 1/4) performanţele se înrăutăţesc.*** Explicaţia probabilă a acestui fapt constă în aceea că imaginile la rezoluţie ridicată conţin şi multă informaţie irelevantă pentru discriminarea facială, iar cele cu rezoluţie foarte scăzută nu oferă suficientă informaţie pentru a putea individualiza imaginile faciale.
- la imaginile care folosesc jumătate din rezoluţia iniţială se obţin cele mai bune performanţe vizavi de rata clasificării corecte (8,80% eroare la datele de test, cu fereastră 8x8), dar acestea pot fi în continuare îmbunătăţite prin optimizarea dimensiunii ferestrei de extragere a trăsăturilor. Astfel cea mai scăzută rată a erorii pentru datele de test, 8,20%, se obţine pentru o dimensiune a ferestrei de 7x6 pixeli.

Fişierul codului sursă MATLAB aferent experimentelor prezentate în cadrul

acestui paragraf este prezentat în ANEXA 3.

6.2.4 Influenţa folosirii unor arhitecturi paralele şi ierarhice de RNA-MLP asupra procesului de recunoaştere facială

Se arată de către Căleanu ([10], [11]) că performanţele sistemului de recunoaştere facială pot fi în continuare îmbunătăţite prin folosirea unei arhitecturi paralele ale ansamblului extractor de trăsături + clasificator neuronal.

Rezultatele experimentale pentru cazurile în care agregarea deciziei este efectuată de către o RNA-MLP ierarhic superioară (fig.6.1.5.3.a) şi pentru cazul în care decizia finală se bazează pe medierea rezultatelor parţiale (fig.6.1.5.3.b), sunt prezentate în tab.6.2.4.1.

Din analiza rezultatelor prezentate în tab.6.2.4.1 se constată următoarele:

- ambele arhitecturi paralele oferă ***rezultate superioare*** faţă de sistemul prezentat în paragraful anterior care nu beneficia de acest tip de arhitectură;
- cele mai bune performanţe sunt obţinute în cazurile în care trăsăturile sunt extrase din regiuni ***distincte*** ale imaginii;
- agregarea deciziei prin intermediul ***operatorului medie*** oferă rezultate cu 30...40% mai bune decât sistemul cu agregarea deciziei prin intermediul RNA-MLP şi cu 30...60% mai bune decât sistemul format dintr-un singur ansamblu extractor de trăsături + clasificator neuronal (fig.6.1.5.2 b). ***Diferenţele dintre sisteme sunt uşor amplificate pe măsură ce numărul de imagini de antrenament/ clasă creşte.***
- prin folosirea arhitecturilor paralele ***scade dramatic dimensiunea*** RNA-MLP implicate, ceea ce conduce la o economie de memorie şi timp de calcul.

Codul sursă MATLAB care permite rularea experimentelor prezentate în cadrul acestui paragraf este prezentat în ANEXA 4.

			1 imagine/persoană	3 imagini/persoană	5 imagini/persoană
Tip Sistem	Dimensiunea regiunilor	EPM %	Eroare test [%] Eroare totală [%] Durată experiment [min]	Eroare test [%] Eroare totală [%] Durată experiment [min]	Eroare test [%] Eroare totală [%] Durată experiment [min]
Fig.6.1.5.3 a) (3RNA + RNA)	3 regiuni nesuprapuse: 14x46 28x46 14x46	0,01	40,61 36,55 5,4	14,00 9,80 12,8	6,40 3,20 22,4
	3 regiuni parțial suprapuse 28x46 28x46 28x46	0,01	43,94 39,55 6,2	17,28 12,10 13,8	12,00 6,00 17,8
	3 regiuni complet suprapuse 56x46 56x46 56x46	0,01	39,27 35,35 10,4	15,21 10,65 19,2	9,3 4,65 27,4
Fig.6.1.5.3 b) (3RNA + MEDIE)	3 regiuni nesuprapuse: 14x46 28x46 14x46	0,01	31,27 28,15 4,4	10,28 7,20 10,4	3,70 1,85 18,3
	3 regiuni parțial suprapuse 28x46 28x46 28x46	0,01	37,22 33,50 5,2	12,85 9,00 11,1	8,70 4,35 13,1
	3 regiuni complet suprapuse 56x46 56x46 56x46	0,01	31,33 28,20 9,4	10,57 7,40 19,2	6,30 3,15 23,9

Tab.6.2.4.1 Rezultatele medii pentru 5 experimente consecutive efectuate pentru cazurile în care agregarea deciziei este efectuată de către o RNA-MLP ierarhic superioară (fig.6.1.5.3.a) şi pentru cazul în care decizia finală se bazează pe medierea rezultatelor parţiale (fig.6.1.5.3.b). Imaginile originale au fost reeşantionate cu un factor de ½ deci sunt de dimensiune 56x46 pixeli, iar dimensiunea ferestrei din care se extrag trăsăturile este 7x6 pixeli. Mulţimile de antrenament şi test au fost obţinute cf. specificaţiilor din §6.2.1.

6.3 Concluzii

În subcapitolul 6.1 autorul tezei propune o arhitectură [85] pentru un sistem de detecţie şi recunoaştere facială precum şi modalităţi concrete de implementare a etapelor ce le implică un astfel de sistem [93],[94], [95].

În subcapitolul 6.2 sunt prezentate rezultatele experimentale ce vin să susţină validitatea propunerilor din 6.1.

Principalele concluzii desprinse din analiza subcapitolului 6.1 sunt următoarele:

- în etapa de achiziţie a imaginii pot fi folosite cu succes camere video tip WEBCAM care deşi prezintă un preţ redus oferă rezoluţii satisfăcătoare;
- doar în cazuri speciale (prezenţa zgomotului sau variaţii ale iluminării) se impun măsuri de preprocesare a imaginii achiziţionate;
- în ceea ce priveşte detecţia feţei, autorul tezei a implementat două tipuri de algoritmi, unul bazat pe RNA şi altul clasic, bazat pe decompoziţie subspaţială. În primul caz s-au constatat următoarele dezavantaje: timpul consacrat fazei de antrenament este considerabil iar pentru o acurateţe a detecţiei e nevoie de un număr ridicat de exemple din categoria "non-feţe". Şi în cazul algoritmului bazat pe decompoziţie subspaţială au fost identificate anumite dezavantaje: nu se folosesc în nici un fel proprietăţile geometrice ale feţei iar performanţele depind de raportul de aspect al feţelor prezente în imagine. În final s-a optat pentru folosirea celui de al doilea algoritm întrucât viteza de procesare a imaginilor a fost superioară primului.
- pentru etapa de extragere a trăsăturilor autorul tezei a propus o nouă metodă în sensul în care aceasta nu a mai fost aplicată în problema recunoaşterii faciale: metoda operatorului de interes. Justeţea propunerii este ilustrată de rezultatele experimentale prezentate în subcapitolul 6.2
- având în vedere avantaje (vezi § 3.1, § 5.4) ca viteza şi acurateţea clasificării se preferă folosirea, în etapa de clasificare, unui clasificator neuronal, într-o structură

paralelă şi ierarhică, prezentată de autorul tezei în [95].

Principalele concluzii desprinse din analiza subcapitolului 6.2 sunt următoarele:

- în ceea ce priveşte influenţa folosirii tehnicii de extragere a trăsăturilor de tip operator de interes asupra procesului de recunoaştere facială s-a constatat o creştere a performanţei clasificatorului cu 40% , o reducere a dimensiunii datelor la intrarea RNA de 12 ori şi reducerea timpului de procesare cu 75% faţă de cazul fără extragere de trăsături;
- în ceea ce priveşte influenţa rezoluţiei imaginii faciale şi a dimensiunilor ferestrei de extragere a trăsăturilor asupra performanţelor procesului de recunoaştere facială s-a constatat faptul că performanţele sistemului de recunoaştere facială cresc pe măsură ce rezoluţia imaginilor descreşte, ating un maxim pentru un factor de reeşantionare de 1/2 iar dacă rezoluţia scade în continuare (factor de reeşantionare 1/4) performanţele se înrăutăţesc. De subliniat şi concluzia referitoare la existenţa unui optim, în ceea ce priveşte dimensiunea (7x6 pixeli, pentru cazul particular al bazei de date ORL) ferestrei de extragere a trăsăturilor.
- examinând performanţele arhitecturii paralele şi ierarhice de RNA-MLP propusă de autorul tezei se constată o îmbunătăţire a acestora faţă de cazul în care s-a folosit doar un singur clasificator RNA-MLP global. În ceea ce priveşte agregarea răspunsurilor RNA ierarhic inferioare s-a constatat că operatorul medie oferă rezultate mai bune decât o RNA ierarhic superioară. Rezultatele cele mai bune asupra bazei de date ORL (pentru cazul 5 imagini/persoană, eroare date antrenament 0%, eroare date test 3,7%, eroare globală 1,85%) s-au obţinut în cazul în care câmpul receptiv al clasificatorilor neuronali paraleli a fost disjunct.

Comparând performanţele obţinute de către autorul tezei cu cele prezentate în literatura recunoaşterii faciale (pentru o sinteză a rezultatelor vezi [79]), referitoare la aceeaşi bază de date şi în aceleaşi condiţii de experimentare, se poate afirma faptul că

soluţia aleasă de autorul tezei se situează la doar 0,7% în urma celei mai bune performanţe raportate în literatură [71].

CAPITOLUL 7

Implementarea unui sistem de recunoaştere facială

7.1 Introducere

Pe baza specificațiilor prezentate în capitolul anterior se urmăreşte în cele ce urmează dezvoltarea unei aplicații capabile să efectueze detecția şi recunoaşterea imaginilor faciale în timp real.

O astfel de aplicație trebuie să îndeplinească anumite condiții:

- Posibilitatea **interfațării aplicației cu sursa de semnal video** folosită la captura imaginilor faciale. Aceasta presupune pornirea/opririrea fluxului video precum şi captura anumitor cadre din acesta.
- Posibilitatea de **selecție şi/sau detecție automată a prezenței unei imagini faciale** într-o imagine.
- Posibilități de **manipulare** (creare, adăugare, ştergere) a unei **baze de date** care să cuprindă imagini faciale. Este recomandată şi posibilitatea de vizualizare a bazei de date.
- Posibilitatea lansării unui **proces de recunoaştere** a imaginii faciale selectate sau detectate din fluxul video prin compararea cu cele existente în baza de date. Aplicația trebuie să divizeze imaginea facială extrasă din scenă în **subimagini**, conform celor discutate anterior. Fiecare dintre aceste subimagini este supusă procedurii de **extragere a trăsăturilor** prin metoda operatorului de interes. **Rețelele neuronale** aferente fiecărei subimagini, în prealabil antrenate pentru recunoaşterea bazei de date existente la un moment dat, vor lua independent decizii în ceea ce priveşte apartenența imaginii faciale prezentate la una din clasele deja existente. Decizia finală va fi constituită pe baza agregării

răspunsurilor RNA individuale. Este recomandată existenţa posibilităţii de modificarea a unui aşa-numit prag de recunoaştere. Prin aceasta se poate modifica nivelul de încredere cu care RNA semnalizează existenţa unei persoane în baza de date.

Soluţiile de implementare pentru un sistem de recunoaştere facială care să îndeplinească condiţiile mai sus menţionate pot fi atât **software** cât şi **hardware**. Opţiunea se face în funcţie de destinaţia finală (supraveghere aeroporturi, stadioane, acces clădiri etc.), modul de operare (asistat sau autonom) şi resurse disponibile (camere video, calculatoare, sisteme cu procesor de semnal).

În cele ce urmează s-a optat pentru implementarea software a sistemului de recunoaştere facială (subcapitolul 7.2) dar sunt prezentate soluţii de implementare şi pentru un sistem hardware (subcapitolul 7.3). Argumentele care stau la baza acestei alegeri au în vedere în primul rând versatilitatea, flexibilitatea soluţiei soft. Prin contrast, soluţia hardware deşi este mai rapidă, e utilă doar în momentul în care se operează cu o bază de date stabilă, pentru care procesul de antrenament al RNA nu trebuie să fie reluat.

7.2 Implementarea software

Deşi nu este cea mai rapidă metodă de implementare, oferă avantajul flexibilităţii, în sensul în care pot fi operate cu uşurinţă modificări necesare în special în faza de antrenare, testare şi optimizare a sistemului de recunoaştere facială.

Există numeroase medii şi limbaje de programare, dar potrivite pentru dezvoltarea unor aplicaţii cu RNA sunt **C/C++** şi **MATLAB**. Atât în C++ cât şi în MATLAB se poate aplica noţiunea de obiect în construcţia RNA. Deosebirea dintre cele două limbaje este aceea că C este **compilat** pe când scriptul **MATLAB** e interpretat. De aici ar rezulta o diferenţă de viteză favorabilă C-ului. În realitate, experimentele descrise în continuare arată o diferenţă de viteză de 15% între implementări identice C şi

MATLAB ale unei RNA, care nu reuşeşte să surmonteze anumite avantaje pe care le oferă mediul MATLAB şi care sunt discutate în cele ce urmează.

7.2.1 Mediul MATLAB v.5.3

În cadrul acestei lucrări toate referinţele au în vedere **MATLAB versiunea 5.3.0** (R11), 21.01.1999.

MATLAB (**MAT**rix **LAB**oratory) [96], [97], [98], [99] reprezintă un mediu de programare destinat calculului numeric şi reprezentărilor grafice în domeniul ştiinţei şi ingineriei. Elementul de bază cu care operează MATLAB este **matricea**. **Este uşor de extins**, prin faptul că orice utilizator poate adăuga propriile funcţii sau le poate modifica pe cele existente. Folosind funcţiile predefinite ale MATLAB-ului se obţine o **importantă economie de timp** în crearea de noi aplicaţii. Deasemenea pachetul software include un set de funcţii specifice (denumite "toolbox") anumitor domenii, ca de exemplu: reţele neuronale, logică fuzzy, prelucrări de imagine etc. .

Există posibilitatea modelării, analizei şi simulării **sistemelor dinamice**, prin descrierea acestora **la nivel de schemă bloc** prin intermediul mediului Simulink. În acestă categorie pot fi incluse, de exemplu, sisteme cu procesoare numerice de semnal ("DSP Blockset") sau sisteme bazate pe circuite electronice de putere ("Power System Blockset").

Adăugând la cele de mai sus şi numeroasele posibilităţi de reprezentare grafică 2 sau 3D a datelor şi **posibilitatea interfaţării codului MATLAB cu cel scris în "C" sau limbaj de asamblare** pentru procesoarele de semnal, avem argumentele necesare care să justifice implementarea aplicaţiilor cu reţele neuronale prin intermediul MATLAB.

Aceste posibilităţi sunt sintetizate în fig.7.2.1.1.

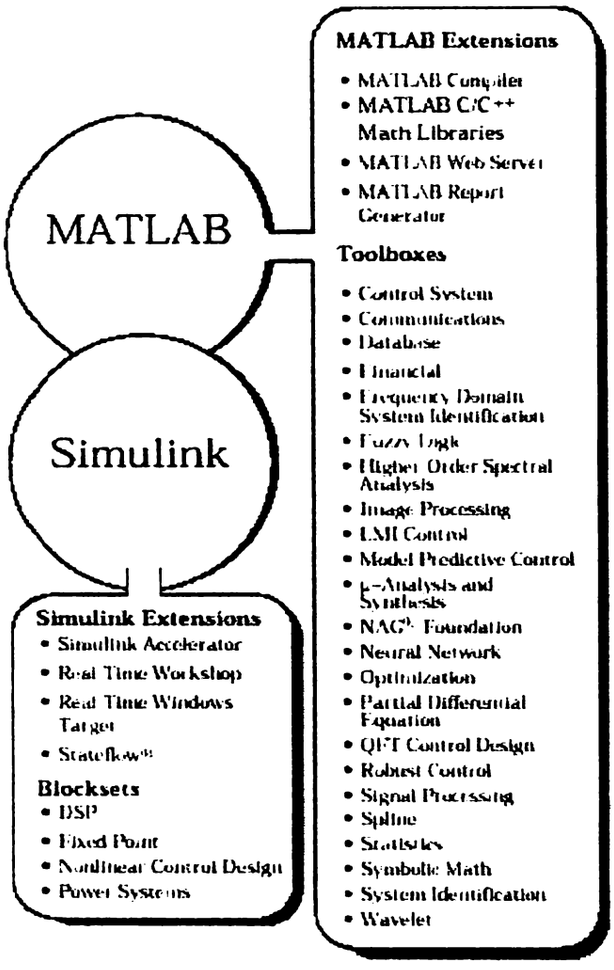


Fig. 7.2.1.1 Posibilităţile oferite de mediul MATLAB v.5.3 [96].

Multitudinea posibilităţilor de implementare pentru RNA oferită de setul specific de funcţii (toolbox) dedicat acestui domeniu [100], ***posibilitatea interfaţării codului MATLAB cu cel din C sau ASM, facilităţile oferite în dezvoltarea interfeţei grafice utilizator sunt argumente pentru alegerea acestui mediu în dezvoltarea aplicaţiei pentru recunoaşterea facială.***

7.2.2 Dezvoltarea interfeței grafice utilizator

Interfața grafică utilizator, (GUI – **G**raphical **U**ser **I**nterface) are rolul de a mijloci comunicarea între program şi cel care îl utilizează. Deşi, în aparență, ar putea reprezenta un aspect secundar în realizarea unei aplicații, de multe ori ***succesul acesteia depinde de calitatea designului GUI.***

S-a considerat utilă formularea unor principii după care se va efectua proiectarea interfeței grafice:

- ***Considerații privind aspectele statice ale GUI***

Principiile pe care trebuie să le îndeplinească GUI sunt, cf. [101]: simplitatea, consistența şi familiaritatea (fig.7.2.2.1).

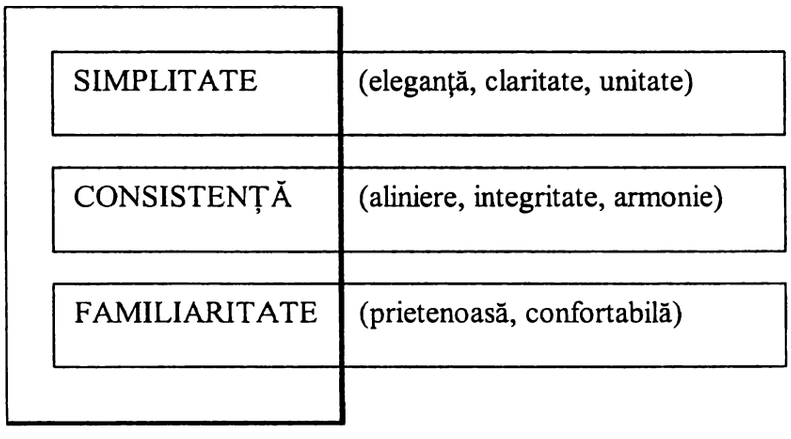


Fig.7.2.2.1 Principiile după care trebuie efectuat designul unei GUI [101].

Simplitatea. Reprezintă scopul principal în proiectarea GUI. O interfață grafică simplă sporeşte claritatea prezentării şi dă un sens unitar acesteia. ***Reprezentarea calitativă***, adică sublinierea formei, poate fi mult mai importantă decât ***reprezentarea cantitativă***, numerică.

Tot în acest context se recomandă o ***arie minimă de interacțiune*** între utilizator şi program. Se preferă păstrarea unui număr redus de figuri pe ecran. ***Introducerea***

datelor se preferă a fi făcută ***grafic*** şi nu numeric.

Consistenţa. Serveşte la eliminarea confuziei şi lipsei orientării utilizatorului în momentul în care interacţionează cu programul.

De exemplu, în fig. 7.2.2.2 sunt prezentate două programe MATLAB demonstrative total diferite în fondul problemei. Una se referă la capabilităţile MATLAB de analiză a sunetului iar cealaltă la posibilităţile de reprezentare grafică 3D.

Se observă însă acelaşi amplasament în pagină a controalelor (în partea dreaptă a figurilor, cu chenar) şi chiar unele controale comune ("Info", "Close"). Acest aspect înlesneşte utilizarea unor aplicaţii diferite ca fond dar care îmbracă aceeaşi formă.

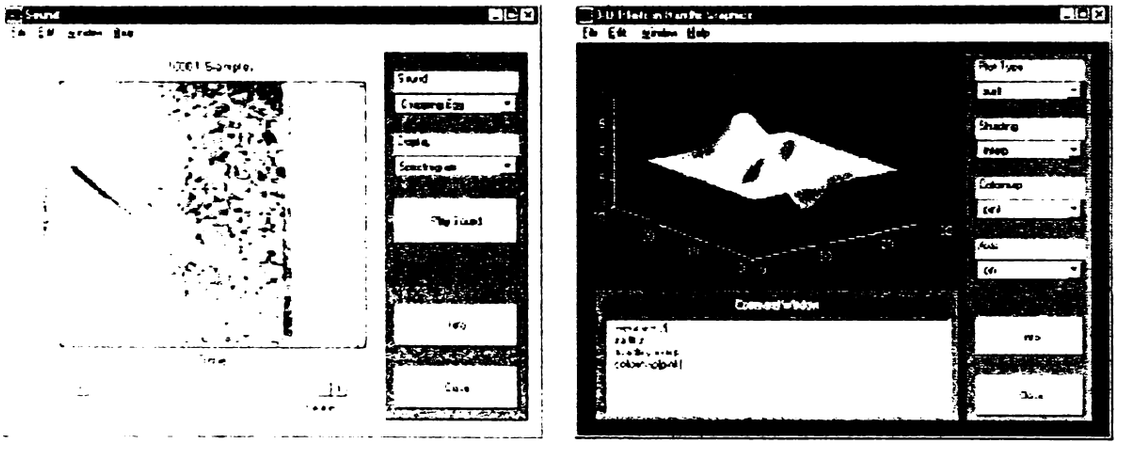


Fig.7.2.2.2 Deşi cele două aplicaţii MATLAB tratează aspecte cu totul diferite, amplasamentul în pagină este identic. Acest fapt elimină confuzia şi dezorientarea utilizatorului.

Familiaritatea. Dacă interfaţa grafică este într-un anumit sens familiară utilizatorilor, aceştia vor învăţa mai repede să o utilizeze. GUI poate să facă apel la experienţa utilizatorului pentru a grăbi înţelegerea acesteia. De exemplu, în fig. 7.2.2.3 se prezintă forma de undă (sus) şi analiza spectrală (jos) a tonurilor generate de

formarea unui număr de telefon. Pentru generarea tonurilor se foloseşte chiar reprezentarea grafică a tastaturii acestuia.

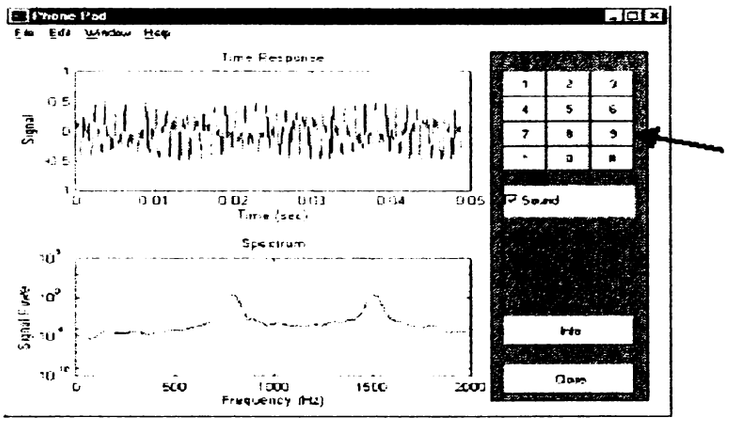


Fig. 7.2.2.3. GUI trebuie să introducă elemente familiare utilizatorului.

- ***Considerații privind aspectele dinamice la GUI***

În momentul în care se interacționează cu GUI, acțiunile trebuie să fie: ***imediate***, ***continue*** şi ***reversibile*** (fig.7.2.2.4).

IMEDIAT	(direct)
CONTINUU	(lin)
REVERSIBIL	(prietenos, încurajator)

Fig.7.2.2.4 Principii ale aspectului dinamic al GUI [101].

În acest sens este recomandată manipularea directă a datelor, ca şi când acestea ar reprezenta un obiect solid. Având în vedere conceptele de acţiune imediată şi continuă, se recomandă, dacă timpul de calcul o permite, afişarea instantanee a rezultatului obţinut asupra datelor selectate. Dacă timpul de calcul necesar unei acţiuni este semnificativ, se recomandă folosirea unui buton pentru invocarea acesteia.

În exemplul din fig. 7.2.2.5 utilizatorul are posibilitatea selecţiei directe a formei de undă (sus) după care transformata Fourier corespunzătoare acesteia va fi afişată instantaneu (jos). Se poate ilustra astfel cu uşurinţă relaţia dintre semnal şi transformata sa Fourier.

În final, reversibilitatea se referă la capacitatea de a anula efectul uneia sau mai multor acţiuni deja efectuate. Este uneori mai greu de implementat, dar întotdeauna apreciată de utilizator. Capacitatea de anulare încurajează experimentarea diverselor acţiuni şi dă un confort sporit în utilizarea GUI.

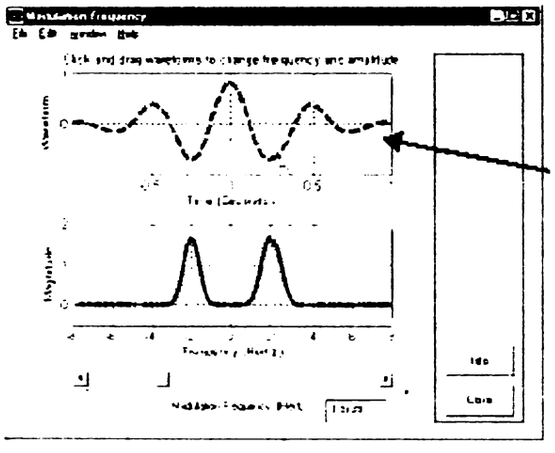


Fig.7.2.2.5 Exemplu de manipulare directă a datelor, cu afişarea instantanee a rezultatului acţiunii.

Se poate afirma în concluzie că pe baza principiilor mai sus enunţate, crearea unei interfeţe grafice utilizator, în general, sau folosind mediul MATLAB, în particular, presupune existenţa a două faze: proiectare şi implementare (fig.7.2.2.6).

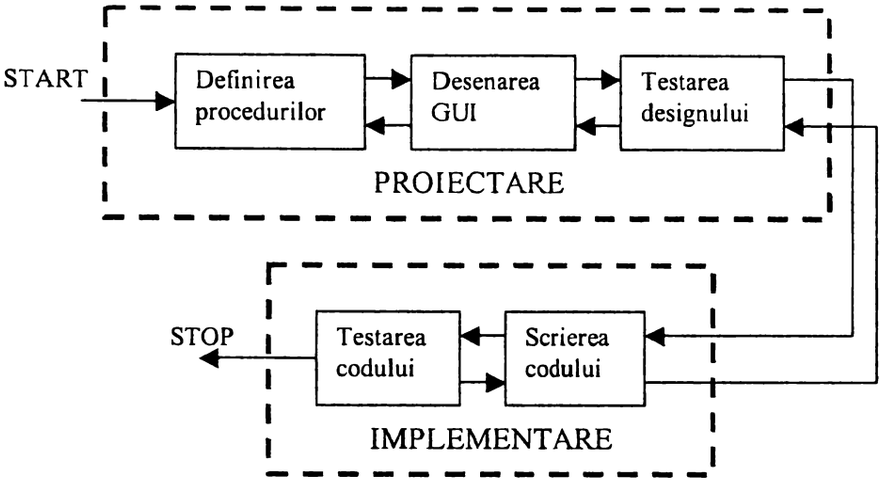


Fig. 7.2.2.6 Etapele realizării unei interfețe grafice utilizator.

Ținând cont de aspectele precizate anterior se va trece la realizarea programului destinat detecției şi recunoaşterii faciale folosind mediul MATLAB.

7.2.3 Recunoaştere facială bazată pe rețele neuronale – implementare MATLAB

Schema bloc a programului MATLAB este prezentată în fig.7.2.3.1. Conform acesteia, paşii necesar a fi parcurşi sunt următorii:

1. **Inițializarea**. În cadrul acestei etape se inițializează anumite variabile, se încarcă baza de date ce conține imagini faciale, se lansează în execuție interfața grafică utilizator şi funcția care permite afişarea bazei de date.
2. **Mod de operare**. La nivelul GUI se analizează în primul rând starea logică a modului de operare ("Continuu" sau "Secvențial), implicit setată ca "adevărat" pentru modul "Secvențial". În acest caz succesiunea temporală a operațiilor este lăsată la latitudinea utilizatorului, fiind condiționată uneori doar ordinea acestora. În modul "Continuu" se vor efectua în mod automat şi ciclic operațiile: "Start video", "Captură", "Detecție", "Recunoaştere", descrise în cele ce

urmează.

3. **Start video**. Funcţia permite pornirea unei surse video (fig.7.2.3.2). Deoarece MATLAB nu furnizează această facilitate s-a folosit standardul Microsoft Video for Windows, clasa AVICap, care permite manipularea surselor video, prin crearea unei biblioteci încărcate dinamic (*.dll) apelabilă din MATLAB.
4. **Captură**. În momentul în care este apelată permite selectarea cadrului curent din fluxul video (fig.7.2.3.3).
5. **Selecţie faţă**. Permite selectarea manuală, prin intermediul mouse-ului, a unui dreptunghi care să cuprindă o faţă umană, din scena capturată anterior.
6. **Detecţie faţă**. Permite izolarea automată a unei feţe umane conţinută în imaginea obţinută la punctul 4 (fig.7.2.3.4).. Deoarece algoritmul procesează imaginea capturată în multiple rezoluţii, operaţiunea de detecţie automată poate să dureze un timp îndelungat, dependent de parametrii algoritmului şi viteza procesorului pe care rulează aplicaţia (ex.: 2 min. pentru INTEL CELERON 533Mhz).
7. **Recunoaştere**. Presupune extragerea trăsăturilor şi antrenarea reţelelor neuronale pentru baza de date curentă, apoi extragerea trăsăturilor din imaginea facială obţinută la pct. 5 sau 6 şi compararea acesteia cu cele existente în baza de date. Dacă răspunsul oferit se situează sub un anumit prag, setabil din "Prag recunoaştere", nu este evidenţiată nici o imagine din baza de date. Altfel, se va evidenţia imaginea recunoscută.

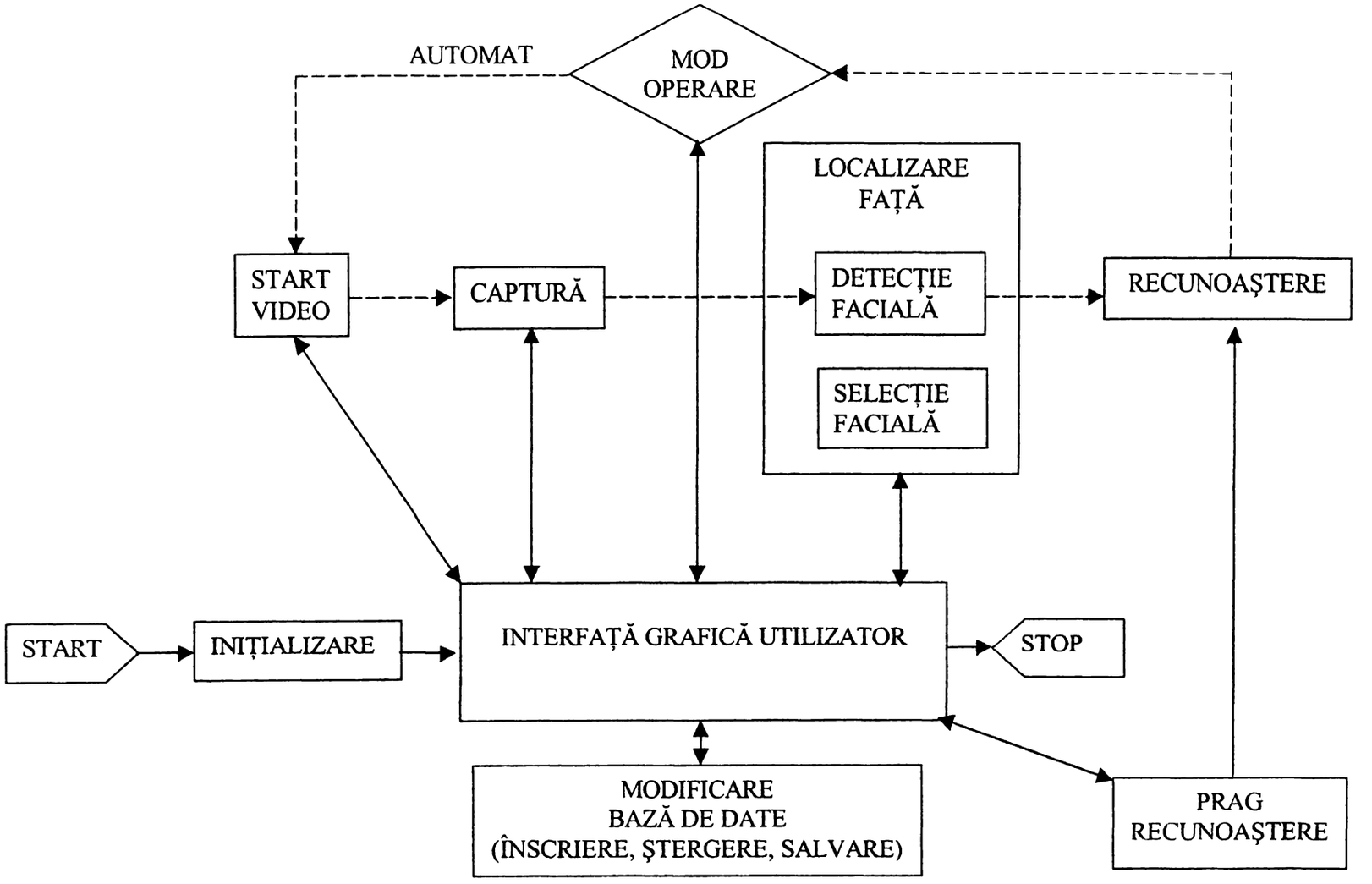


Fig.7.2.3.1 Schema bloc a programului de recunoaştere facială.

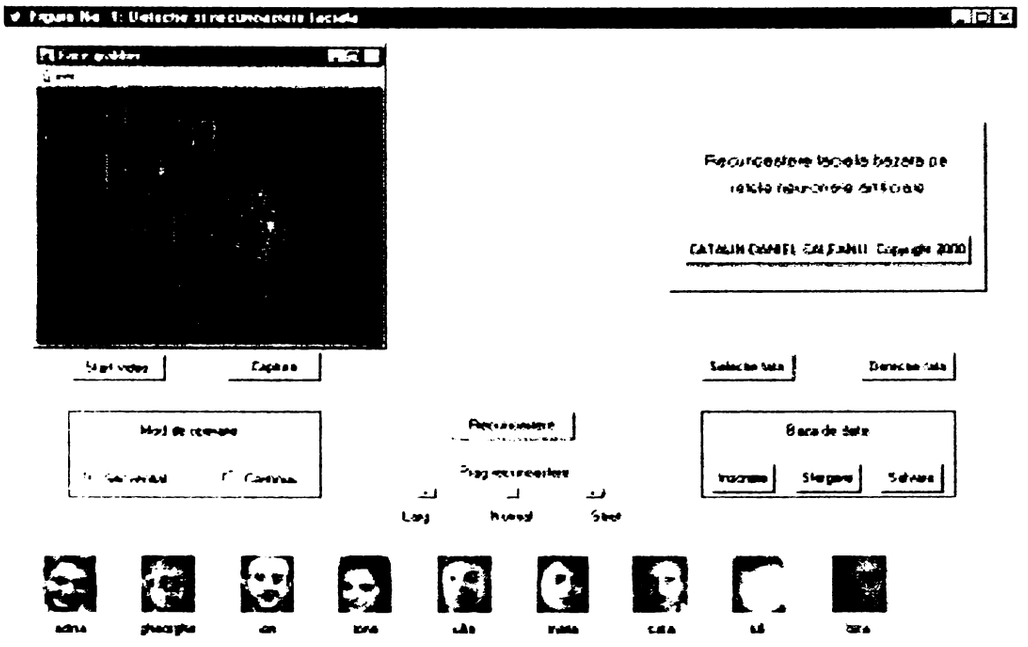


Fig.7.2.3.2 Interfaţa grafică utilizator permite vizualizarea unui flux video provenit de la o cameră video, în acest caz de tip WEBCAM.

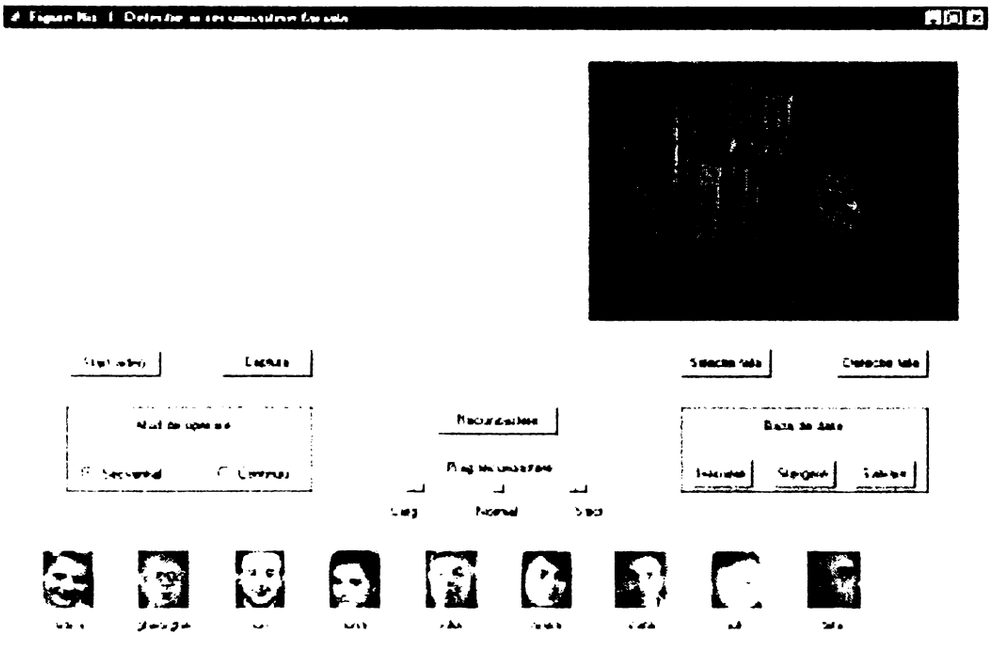


Fig.7.2.3.3 Prin funcţia "Captură" se achiziţionează cadrul curent din fluxul video.

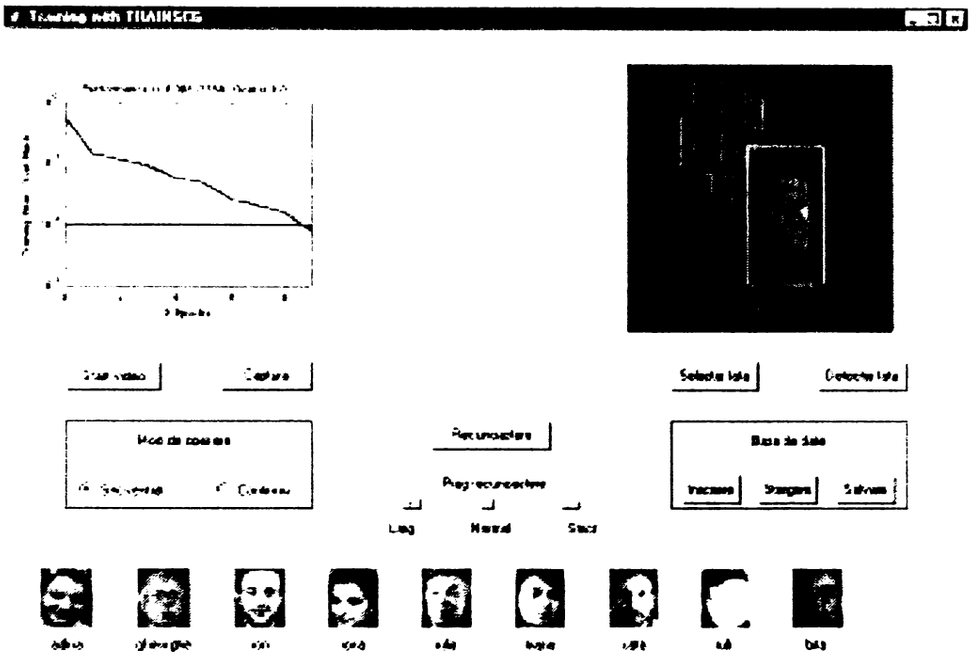


Fig.7.2.3.4 Se observă imaginea facială detectată automat în imagine ("Detecţie faţă) şi evoluţia procesului de antrenament pentru recunoaşterea identităţii ("Recunoaştere").

7.2.4 Mediu de dezvoltare pentru aplicaţii MLP – implementare "C/C++"

BPNNSim v.1.0 (fig.7.2.4.1) reprezintă o aplicaţie ***Windows 95/98*** scrisă în ***Microsoft Visual C++ 5.0***, dezvoltată de către autorul tezei [84] în scopul implementării unei reţele neuronale artificiale (RNA) total conectată de tip feedforward (***Multilayer Perceptron, MLP***) adică un complex de neuroni artificiali organizaţi pe nivele (straturi) şi între care există conexiuni ponderate astfel încât un neuron nu poate fi conectat decât cu neuroni din stratul imediat superior. În acest mod s-a reuşit evaluarea comparativă a implementărilor MATLAB şi C/C++ pentru RNA.

Algoritmul de învăţare este de tip supervizat cu propagarea înapoi a erorii (***Backpropagation, BP***).

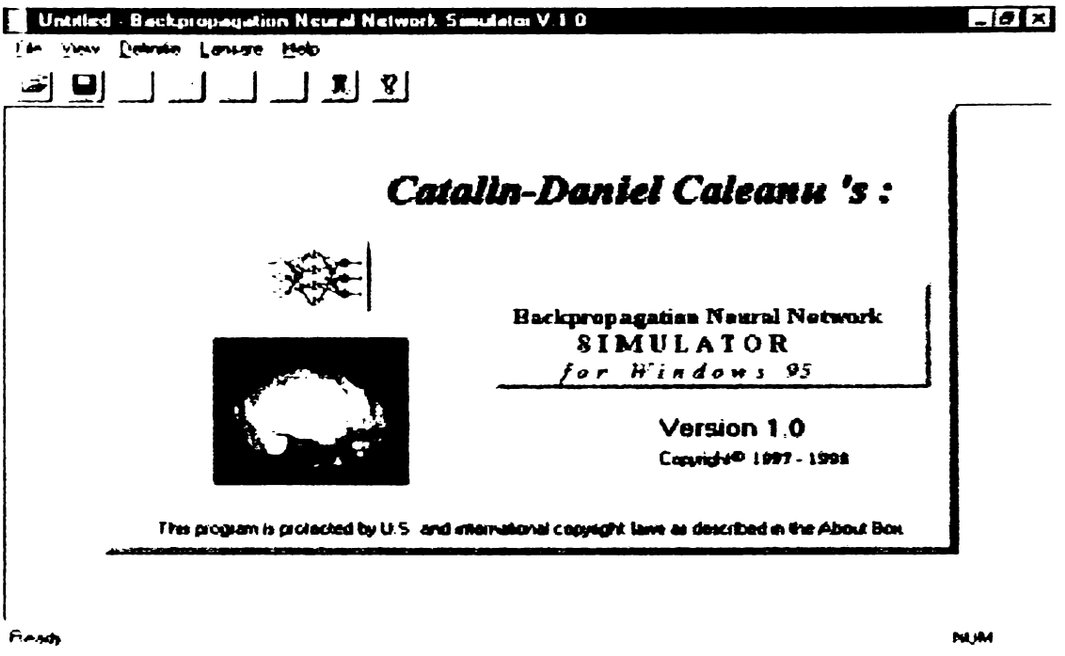


Fig.7.2.4.1 Mediul de dezvoltare C/C++ pentru RNA-MLP

Pentru a creea o aplicație cu RNA folosind BPNNSim se impune parcurgerea următoarelor etape [84]:

1. DESCRIEREA REȚELEI; se face prin deschiderea meniului "Definiție".

a) Submeniul "Număr neuroni" (fig 7.2.4.2) permite specificarea:

-numărului de straturi ascunse ale rețelei;

-numărului de neuroni din stratul de intrare, ieşire sau ascuns.

b) Submeniul "Constante rețea" (fig.7.2.4.3) permite modificarea:

-ratei de învățare inițiale, eta0; reglează viteza de învățare a rețelei.

-tau; rata de învățare se modifică după expresia:

eta[nr_epoci] = eta0/[1+nr_epoci/tau];

-momentum; adaugă o anumită inerție în modificarea ponderilor.

-constanta funcției de activare; fiecare neuron are o funcție de activare sigmoidală.

-eroarea medie mimimă după care procesul de învățare este oprit.

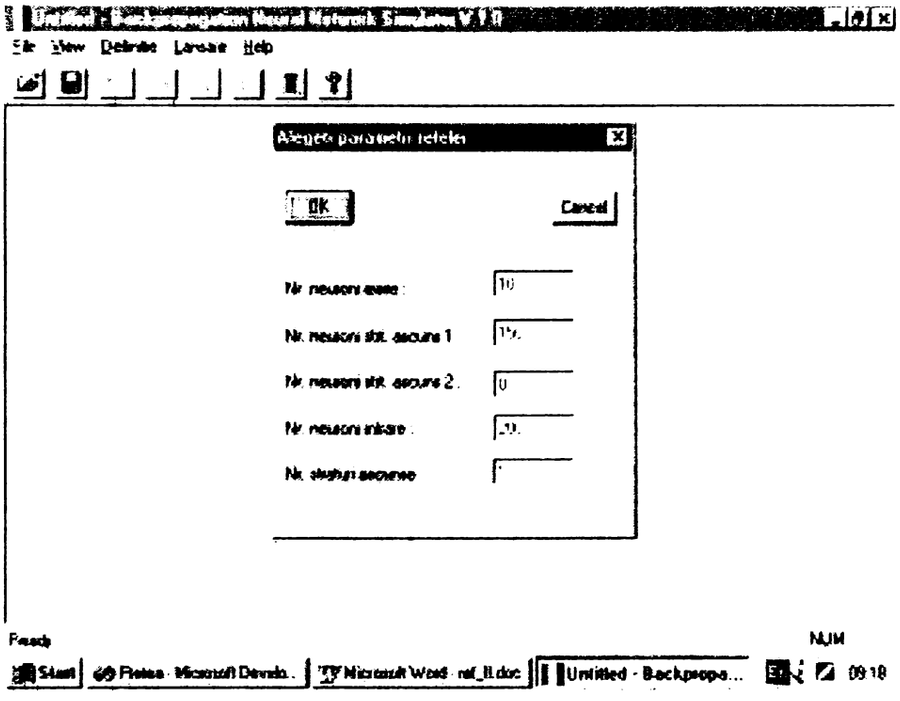


Fig. 7.2.4.2 Configurarea interactivă a RNA, submeniul "Număr neuroni".

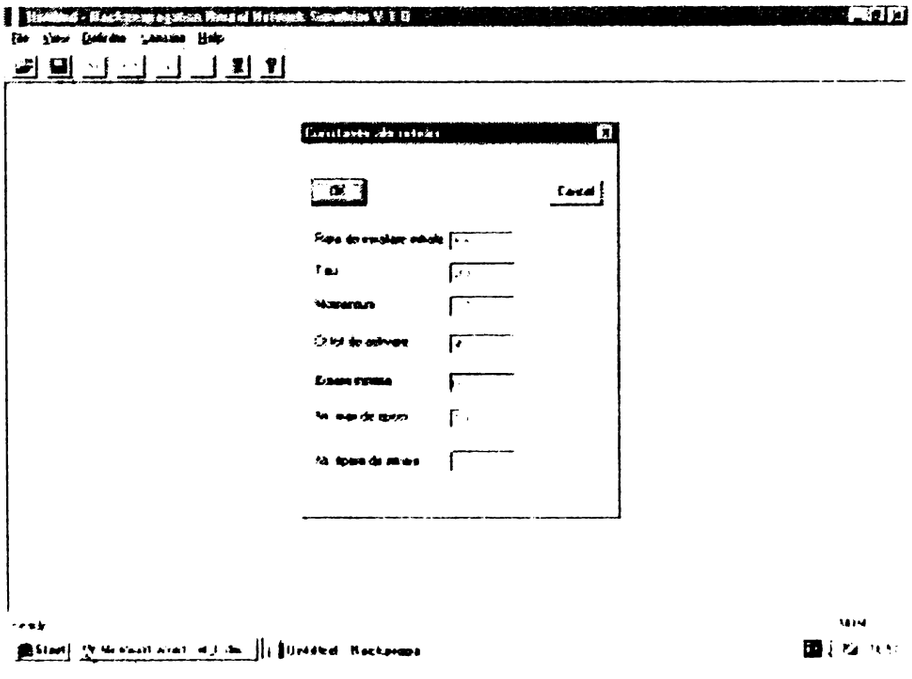


Fig. 7.2.4.3 Configurarea interactivă a RNA, submeniul "Constante rețea".

-numărul maxim de epoci după care procesul de învăţare este oprit.

-numărul de tipare aplicate la intrarea reţelei.

2. LANSAREA PROCESULUI DE ANTRENAMENT; se face prin deschiderea submeniului "Antrenament" din meniul "Lansare" (fig. 7.2.4.4). În această etapă reţeaua "învaţă" un număr de tipare specificat la 1b), fiecare având dimensiunea stratului de intrare.

Dacă eroarea a scăzut suficient de mult se pot salva cunoştintele acumulate ale reţelei sub forma ponderilor şi deplasamentelor neuronilor.

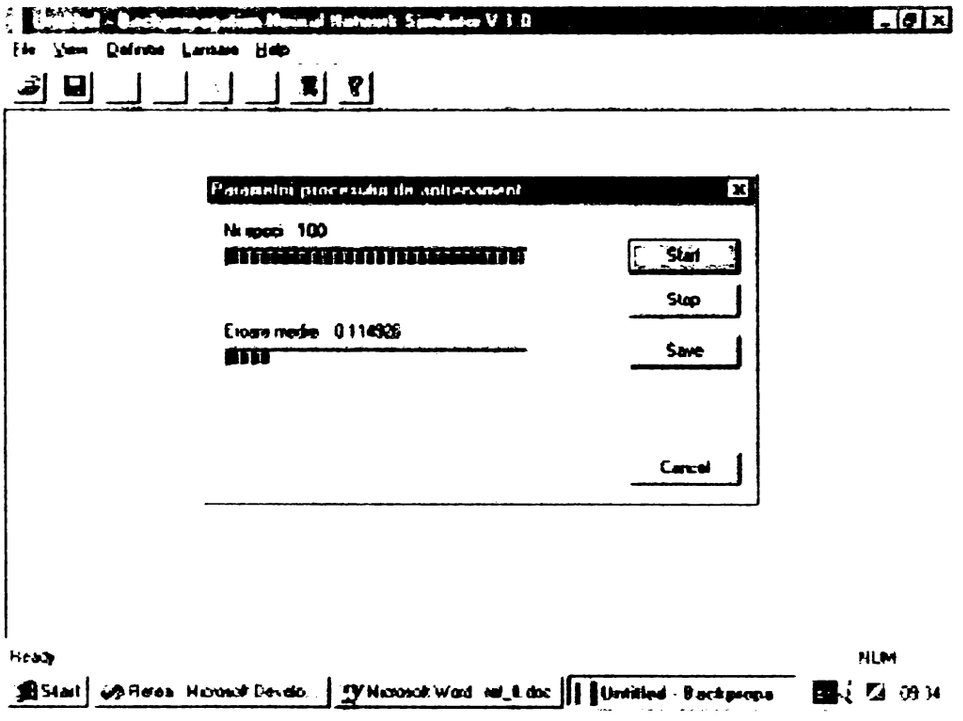


Fig. 7.2.4.4 Desfăşurarea procesului de antrenament

3. LANSAREA PROCESULUI DE UTILIZARE; se selectează fişierul de intrare (în cazul din fig. 7.2.4.5 este vorba despre 0.raw care reprezintă datele de antrenament referitoare la clasa "0") pentru care se doreşte efectuarea clasificării ("Deschide" din meniul "Fişier").

Se apelează "Utilizare" din meniul "Lansare". După inițializarea rețelei (încărcarea cunoştintelor) este afişată starea de activare a fiecărui neuron din stratul de ieşire (se observă faptul că tiparul aplicat la intrare a fost clasificat corect, ieşirea neuronului răspunzător pentru clasa "0" fiind cea activă). Se poate repeta aplicarea de tipare la intrare după acelaşi procedeu.

4. OBSERVAȚII

4.1 Numărul de octeți citiți din fişierele de intrare (*.raw) este identic cu cel al numărului de neuroni din stratul de intrare. Fiecare neuron din stratul de intrare primeşte o valoare cuprinsă între 0 si 255.

4.2 Deoarece rețeaua este folosită ca un clasificator, ieşirea neuronilor din stratul de ieşire ia valori în intervalul 0...1. Tiparul aplicat la ieşirea rețelei (*.out) este format din caractere 0 sau 1.

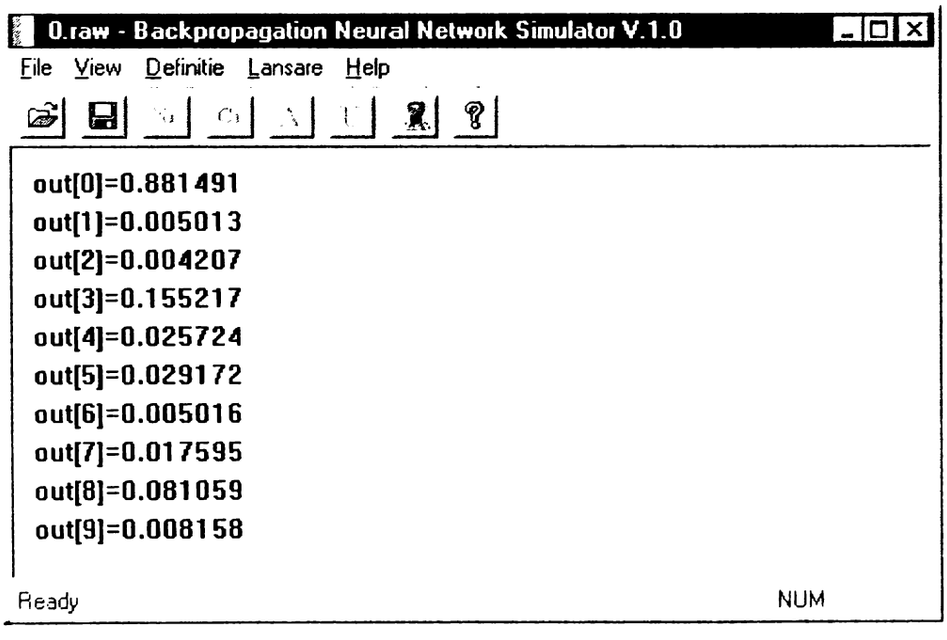


Fig. 7.2.4.5 Afişarea stării neuronilor stratului de ieşire.

Evaluând comparativ performanțele a două RNA identice implementate în MATLAB respectiv în C/C++ folosind BPNNSim se constată următoarele:

- viteza de execuţie a versiunii MATLAB e în medie cu 15% mai lentă decât cea C/C++;
- uşurinţa implementării RNA în MATLAB este net superioară celei "C/C++". Spre exemplu scrierea şi depanarea aplicaţiei "C/C++" a durat 4 săptămâni pe când implementarea MATLAB, folosind funcţiile predefinite existente în Neural Network Toolbox, ia câteva minute.

7.3 Implementarea hardware

Procesoarele numerice de semnal (DSP) par a fi una dintre opțiunile viabile pentru implementarea hardware a RNA datorită vitezei mari de procesare şi a arhitecturii optimizate pentru procesarea paralelă cerută în cazul acestor aplicații. În [102] autorul tezei şi colectivul prezintă un sistem care permite dezvoltarea rapidă a aplicațiilor industriale ale RNA iar în [103] este prezentată o aplicație concretă a principiilor enunțate în [102]. Procedura de dezvoltare implică doi paşi:

a) Simularea RNA folosind un mediu corespunzător (ex.: MATLAB, BPNNSim), pentru stabilirea configurației şi parametrilor optimi pentru aplicația în cauză;

b) Implementarea RNA pe un sistem cu procesor de semnal care să îndeplinească cerințele necesare operării în timp real.

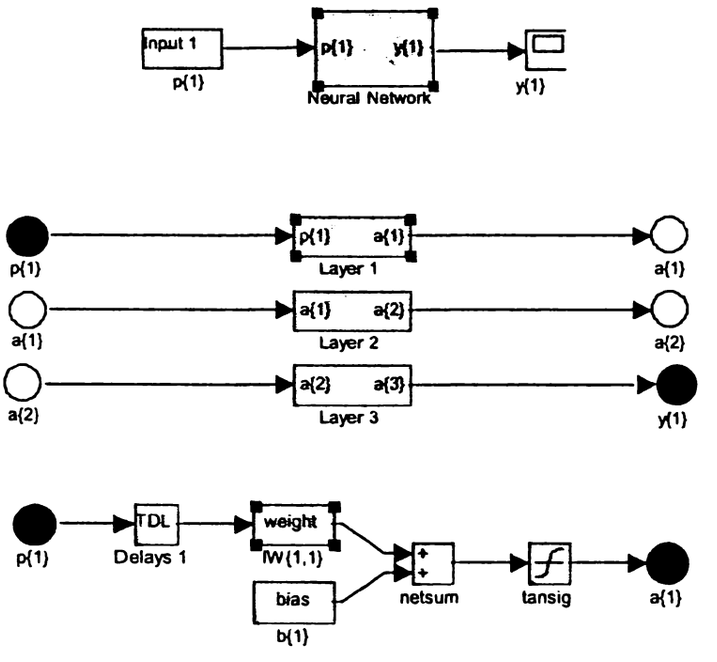
Problema principală constă în definirea unui obiect de tip rețea neuronală care să poată circumscrie totalitatea arhitecturilor posibile (RNA tip MLP, RBF, SOFM etc.) printr-o reprezentare unificatoare. Un exemplu de rezolvare pentru această problemă este propus în [100]; se pot observa în fig. 7.3.1 a şi b proprietăție unui obiect de tip RNA pentru cazul MLP cu un singur strat ascuns şi implementarea la nivel de bloc. De asemenea există posibilitatea implementării unor arhitecturi RNA atipice (fig.7.3.3).

În concluzie, cvasitotalitatea RNA pot fi implementate definind doar 3 tipuri de blocuri: a) tip intrare; b) tip funcție de transfer; c) tip funcție pondere (fig.7.3.2). Practic, printr-o interfață grafică simplă, câmpurile obiectului "rețea neuronală" sunt trimise DSP-ului. Acesta dispune de totalitatea blocurilor implementate în limbajul de asamblare specific DSP-ului, ceea ce conferă viteza necesară unei aplicații în timp real.

O posibilă soluție hardware constă din procesorul de semnal de uz general, preț scăzut, TMS320VC33 [104] folosit ca şi coprocesor şi un microcontroller tip 8051 care să furnizeze semnalele de comandă pentru proces şi să asigure schimbul de date (RS232/RS485) între sistem şi computer. Sistemul cu DSP poate conține o placă de extensie ("daughter board") pentru achizie de date, întreg ansamblul fiind guvernat de către un sistem de operare care rulează pe microcontroller.

```
net1 =

  Neural Network object:

  architecture:

      numInputs: 1
      numLayers: 2
    biasConnect: [1; 1]
   inputConnect: [1; 0]
   layerConnect: [0 0; 1 0]
  outputConnect: [0 1]
  targetConnect: [0 1]

     numOutputs: 1  (read-only)
     numTargets: 1  (read-only)
  numInputDelays: 0  (read-only)
  numLayerDelays: 0  (read-only)

  subobject structures:

         inputs: {1x1 cell} of inputs
         layers: {2x1 cell} of layers
        outputs: {1x2 cell} containing 1 output
        targets: {1x2 cell} containing 1 target
         biases: {2x1 cell} containing 2 biases
   inputWeights: {2x1 cell} containing 1 input weight
   layerWeights: {2x2 cell} containing 1 layer weight

  functions:

      adaptFcn: 'adaptwb'
       initFcn: 'initlay'
     performFcn: 'mse'
      trainFcn: 'trainrp'

  parameters:

     adaptParam: .passes
      initParam: (none)
    performParam: (none)
     trainParam: .epochs, .show, .goal, .time,
                 .min_grad, .max_fail, .delt_inc, .delt_dec,
                 .delta0, .deltamax, .lr, .mc

  weight and bias values:

              IW: {2x1 cell} containing 1 input weight matrix
              LW: {2x2 cell} containing 1 layer weight matrix
               b: {2x1 cell} containing 2 bias vectors

  other:

  userdata: (user stuff)
```

a)



b)

Fig.7.3.1 a) Proprietăţile unui obiect de tip RNA. b) Schema bloc ierarhizată pentru implementarea unei RNA-MLP cu un singur strat ascuns.

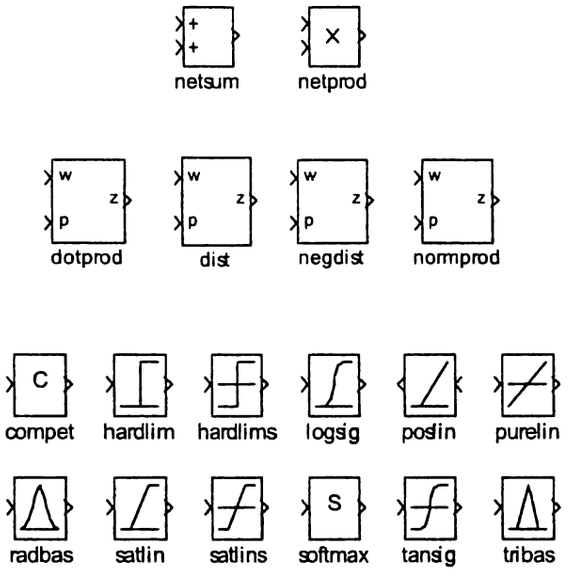


Fig.7.3.2 Blocuri necesare pentru implementarea RNA: tip intrare, pondere şi transfer.

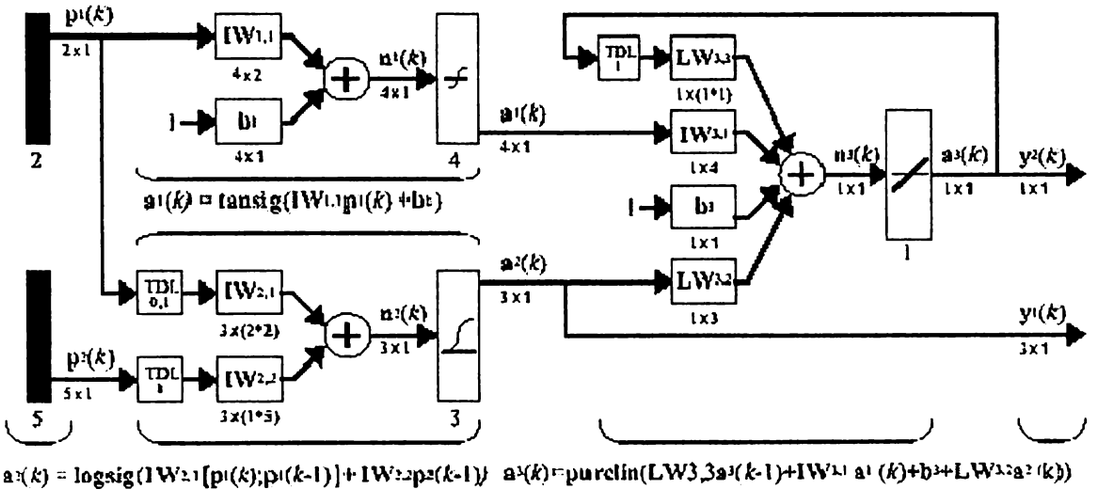


Fig. 7.3.3 Folosind specificaţiile pentru obiectul "reţea neuronală" şi blocurile funcţionale predefinite se poate implementa orice tip de arhitectură de RNA. Se prezintă aici un caz atipic de RNA cu două straturi ascunse.

7.4 Concluzii

În cadrul acestui capitol se tratează problema implementării unui sistem de recunoaştere facială în timp real. Sunt subliniate posibilităţile pe care o astfel de aplicaţie trebuie să le ofere: posibilitatea **interfaţării aplicaţiei cu sursa de semnal video** folosită la captura imaginilor faciale, posibilitatea de **selecţie şi/sau detecţie automată a prezenţei unei imagini faciale** într-o imagine, posibilităţi de **manipulare** a unei **baze de date** care să cuprindă imagini faciale, posibilitatea lansării unui **proces de recunoaştere**.

Sunt propuse două metode concrete de implementare: **software** şi **hardware**. Referitor la opţiunea software se poate concluziona faptul că oferă o viteză de execuţie în general mai mică decât cea hardware dar avantajul unei versatilităţi sporite. Va fi deci deosebit de importantă în etapa de dezvoltare a aplicaţiei. Metoda hardware oferă prin posibilitatea procesării paralele a instrucţiunilor, viteză de lucru sporită la un cost scăzut, însă flexibilitatea metodei este redusă.

În ceea ce priveşte modalitatea software de implementare sunt analizate posibilităţile oferite de mediile **MATLAB v.5.3** şi **Microsoft Visual C++ 5**. Sunt evidenţiate comparativ avantajele şi dezavantajele celor două medii. Principalele concluzii sunt următoarele: **MATLAB** oferă, prin folosirea funcţiilor specifice domeniului reţelelor neuronale ("Neural Network Toolbox") şi celor specifice procesării de imagini ("Image Processing Toolbox") posibilitatea **dezvoltării extrem de rapide a unei aplicaţii**. Se pot modifica cu uşurinţă diverşi parametri ai procesului de antrenament, inclusiv diverşi algoritmi de antrenament. **Facilităţile de vizualizare** în acest caz sunt impresionante. Există deasemenea posibilitatea **translaţiei** codului MATLAB **în C/C++** sau folosirea funcţiilor MATLAB **din C/C++**.

Se arată de către autorul tezei că cea de-a doua soluţie de implementare software, mediul **Visual C++ 5.0**, oferă un spor de viteză mediu de 15% în cazul implementării unei RNA de tip feedforward, faţă de implementarea MATLAB. Având în vedere faptul că algoritmul care descrie funcţionarea unei RNA este deosebit de

complex, costul dezvoltării acestuia în C/C++ devine prohibitiv.

În concluzie, pe baza argumentației sus-menționate, pentru sistemul de recunoaştere facială, în versiune software, este aleasă implementarea MATLAB.

Se remarcă importanța deosebită a principiilor care stau la baza designul unei interfețe grafice utilizator (simplitatea, consistența şi familiaritatea – pentru aspectul static al GUI; imediate, continue şi reversibile – pentru aspectele dinamice ale GUI) şi se concluzionează că succesul aplicației depinde în mare măsură şi de calitatea interfeței grafice utilizator. Pe baza acestor principii se realizează, în MATLAB v.5.3, sistemul de recunoaştere facială (§ 7.2.3) şi, în Visual C++, un mediu de dezvoltare pentru aplicații cu RNA-MLP (§ 7.2.4). ANEXA 5 conține codul sursă MATLAB pentru sistemul de recunoaştere facială propus de către autorul tezei.

În ceea ce priveşte implementarea hardware o posibilă soluție propusă de autorul tezei ([102], [103]) constă din procesorul de semnal de uz general, preț scăzut, TMS320VC33 folosit ca şi coprocesor şi un microcontroller tip 8051 care să furnizeze semnalele de comandă pentru proces şi să asigure schimbul de date (RS232/RS485) între sistem şi computer. Sistemul cu DSP poate conține o placă de extensie ("daughter board") pentru achiziție de date, întreg ansamblul fiind guvernat de către un sistem de operare care rulează pe microcontroller.

CAPITOLUL 8

Concluzii

În prezenta lucrare se abordează domeniului prelucrărilor digitale de imagini (PDI) din perspectiva rețelelor neuronale artificiale (RNA). Se arată că majoritatea prelucrărilor de imagini (filtrare, codare, compresie, segmentare, reprezentare şi interpretare) pot fi abordate cu succes de către RNA şi se concluzionează faptul că acestea oferă certe avantaje față de metodele clasice ale PDI.

Particularizând aspectele sus menționate, teza tratează detaliat problema detecției şi recunoaşterii faciale bazate pe RNA. Autorul tezei propune o arhitectură pentru un sistem de recunoaştere facială bazată pe o combinație dintre metode convenționale ale PDI, în sensul în care acestea nu folosesc RNA, pentru partea de detecție şi extragere de trăsături şi RNA tip perceptron multistrat (RNA-MLP) într-o structură paralelă şi ierarhică pentru etapa de clasificare.

Pentru elaborarea modelului sistemului de recunoaştere facială sunt analizate şi comparate metodele convenționale sau cu RNA prezentate în literatura detecției şi recunoaşterii faciale. În urma acestei analize se desprinde concluzia referitoare la utilitatea unei soluții hibride convențional-neuronal pentru problema recunoaşterii faciale.

Este identificat un principal dezavantaj al soluțiilor RNA şi anume acela al timpului considerabil aferent etapei de antrenament, mai ales în cazul des întâlnit în PDI în care vectorii de antrenament au o dimensionalitate ridicată. În sensul rezolvării acestei probleme, autorul tezei propune două metode de antrenament pentru cazurile în care nu sunt/sunt extrase trăsături din imaginea ce urmează a fi prelucrată: o soluție neuro-fuzzy pentru adaptarea ratei de învățare şi o soluție obținută din modificarea algoritmilor bazați pe metoda gradientului conjugat.

În final autorul tezei oferă implementarea software a considerațiilor teoretice

referitoare la arhitectura optimă a unui sistem de recunoaştere facială. Sunt identificate însă şi posibilităţi hardware de implementare a sistemului.

8.1 Comparaţie cu alte sisteme de recunoaştere facială

Efectuând comparaţia sistemului de recunoaştere facială propus în cadrul acestei lucrări cu alte sisteme de recunoaştere facială prezentate în literatură, asupra aceleiaşi baze de date şi în aceleaşi condiţii de experimentare, se constată ocuparea poziţiei secunde într-un clasament al performanţelor celor mai bune 8 sisteme (tab.8.1.1). Criteriul de comparaţie este cel standard folosit în cazul bazei de date ORL şi anume rata erorii asupra datelor de test pentru cazul 5 imagini/persoană în etapa de antrenament. Performanţa sistemului propus de autorul tezei este situată la o diferenţă de doar 0,6% faţa de cea mai bună metodă prezentată în literatura recunoaşterii faciale.

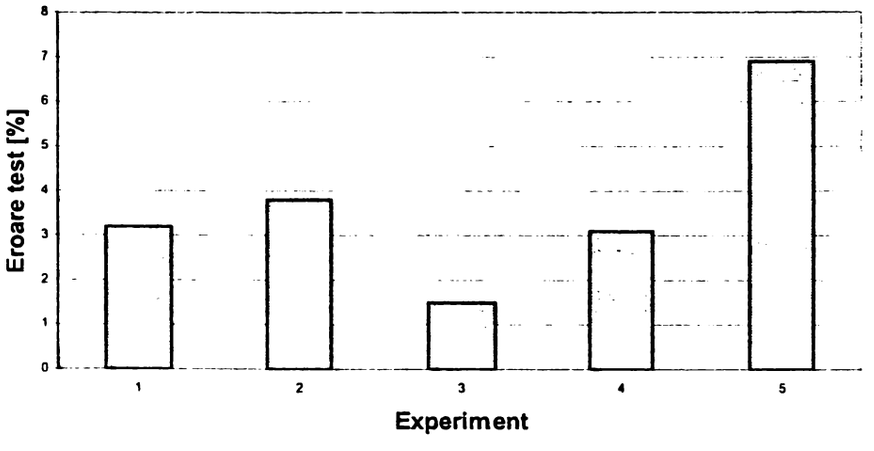


Fig.8.1.1 Eroare medie în etapa de test obţinută ca medie a erorilor în 5 experimente consecutive. Se constată nivelul record al erorii de test 1,5% obţinut în experimentul nr.3.

De subliniat faptul că performanţa de ***3,7%*** eroare asupra datelor de test reprezintă o ***medie efectuată asupra a 5 experimente succesive***. După cum se poate remarca în fig.8.1.1 au fost obţinute ***rezultate individuale*** în care eroarea de test scădea

sub *1.5%*, performanţă neraportată până în prezent de nici unul din sistemele de recunoaştere facială prezentate în literatură. Motivul pentru care se obţin rezultate diferite pentru experimente consecutive este justificat prin modul de alcătuire aleator pentru imaginile de test şi antrenament, cele 200 de imagini de test şi 200 de antrenament fiind de fiecare dată altele. La aceasta s-ar mai putea adăuga, cu o influenţă mai mică însă, şi modul aleator de iniţializare a ponderilor RNA-MLP.

Autor	Tehnică	Eroare date test [%] Număr de imagini/persoană 1	3	5	Timp de prelucrare (caz 5 img./pers.) Antrenament	Clasificare
1. S.Z. Li şi J. Lu [71]	Linia trăsăturilor	-	-	3,125	-	-
2. C.D. Căleanu [95]	Operator interes + 3 x MLP + MED.	31,27	10,28	3,70	18 min. INTEL CELERON 533MHz	0,03 s
3. Lawrence şi colectiv [75]	SOM + CN	30,0	11,8	3,80	4 ore SGI INDY R4400 100MHz	<0,5 s
4. Lin şi colectiv [77]	PDBNN	-	-	4,00	20 min. SGI INDY R4400 100MHz	< 0,1s
5. Samaria şi Hartner [105]	Pseudo 2D-HMM	-	-	5,00	-	4min.
6. Howell şi Buxton [79]	Gabor+RBF Gabor+RBF cu eliminare	51 16	38 9	14 5	8 s 8 s Sun SPARC 20	0,01s 0,01s
7. Pessoa şi Leitao [83]	Celule complexe	26,25	10,95	6,65	-	-
8. Lawrence şi colectiv [75]	Feţe proprii PCA + CN	39 44	18 13	11 8	- -	- -

Tab.8.1.1 Comparaţia între cele mai bune sisteme de recunoaştere facială asupra bazei de date ORL în condiţiile standard de experimentare.

8.2 Contribuții

Sunt sumarizate în continuare contribuțiile originale ale autorului tezei, în ordinea în care acestea apar în structura tezei.

1. **Contribuții în domeniul rețelelor neuronale artificiale folosite în prelucrarea digitală a imaginilor.**

- Sunt identificate proprietăți ale RNA care fac utilă folosirea acestora în PDI [13], [39].
- Se investighează modalitățile de implementare a prelucrărilor de imagini (filtrare, codare, compresie, segmentare, reprezentare şi interpretare) cu ajutorul RNA. Prin rezolvarea câtorva probleme concrete autorul tezei demonstrează că RNA reprezintă o soluție viabilă pentru PDI [29], [20], [21], [22] .

2. **Contribuții privind algoritmi rapizi de antrenament pentru rețele neuronale cu propagare înainte (feedforward).**

- Se prezintă algoritmii de antrenament pentru o RNA feedforward într-o manieră unitară, ca fiind derivați ai unei strategii de optimizare.
- Se efectuează analiza performanțelor principalilor algorimilor de antrenament.
- Autorul tezei propune două metode de antrenament pentru cazurile în care nu sunt/sunt extrase trăsături din imaginea ce urmează a fi prelucrată: o soluție neuro-fuzzy pentru adaptarea ratei de învățare şi o soluție obținută din modificarea algoritmilor bazați pe metoda gradientului conjugat [44], [45], [106].
- Este dezvoltată, conform standardelor MATLAB, o funcție (*trainfuzzy*) care să implementeze metoda de antrenament cu adaptarea ratei de învățare (ANEXA 1) şi sunt comparate performanțele acesteia cu alte metode de antrenament în trei tipuri de probleme (ANEXA 2).

3. Contribuţii în domeniul detecţiei şi recunoaşterii faciale.

- Se realizează o privire de ansamblu asupra metodelor convenţionale şi neuronale de recunoaştere facială [85]. Se compară avantajele şi dezavantajele metodelor mai sus menţionate în ceea ce priveşte modalităţile de preprocesare, extragere trăsături şi clasificare şi se deduce faptul că arhitectura optimă a unui sistem de recunoaştere trebuie să fie bazată pe o soluţie hibridă convenţional-neuronală [84], [85].
- Autorul tezei propune o arhitectură pentru un sistem de detecţie şi recunoaştere facială. Pentru fiecare din etapele ce le presupune acest sistem sunt oferite soluţii de implementare.
- Se demonstrează viabilitatea unei metode de extragere a trăsăturilor, denumită metoda operatorului de interes pentru cazul recunoaşterii faciale. După cunoştinţa autorului tezei această metodă nu a mai fost aplicată în problema recunoaşterii faciale. Se arată că prin folosirea acestei metode se obţin: o eroare aferentă datelor de test pentru problema ORL de cca. 8,80% deci o acurateţe a performanţelor clasificatorului cu 40% mai bună faţă de cazul în care imaginile ar fi fost aplicate direct, o reducere a dimensionalităţii vectorilor de antrenament de 12 ori şi o reducere a timpului de antrenament de 4 ori [93].
- Se studiază efectul rezoluţiei imaginii faciale asupra ratei de clasificare. S-a constatat că performanţele sistemului de recunoaştere facială cresc pe măsură ce rezoluţia imaginilor descreşte, ating un maxim pentru un factor de reeşantionare de 1/2 iar dacă rezoluţia scade în continuare (factor de reeşantionare 1/4) performanţele se înrăutăţesc. Totodată s-a studiat efectul dimensiunii ferestrei din care sunt extrase trăsăturile asupra performanţei sistemului de recunoaştere facială [94]. Codul sursă MATLAB aferent acestor experimente este oferit în ANEXA 3.
- Se arată că performanţele sistemului pot fi în continuare îmbunătăţite prin folosirea unei structuri paralele şi ierarhice ale ansamblului extragere trăsături – clasificator. Se investighează două modalităţi concrete de agregare a răspunsurilor RNA individuale: printr-o RNA ierarhic superioară şi prin medierea răspunsurilor RNA

individuale. A doua metodă oferă rezultatele cele mai bune, obținându-se pentru baza de date ORL, cazul 5 imagini/persoană, performanța de 3,70% pentru eroarea aferentă datelor de test (valaoare medie a 5 experimente consecutive) ceea ce situează metoda propusă pe poziția a doua în topul celor mai buni 10 algoritmi de recunoaştere facială [95]. Codul sursă MATLAB aferent acestor experimente este oferit în ANEXA 4.

4. Contribuții în implementarea unui sistem de recunoaştere facială.

- Se analizează posibilitățile software (MATLAB şi Visual C++) de implementare a sistemului de recunoaştere facială propus şi se compară performanțele obținute. Se concluzionează că deşi varianta MATLAB are un deficit de 10% în ceea ce priveşte viteza de execuție pentru cazul concret al unei aplicații de recunoaştere facială, este preferabilă celei Visual C++ pentru implementarea sistemului de recunoaştere facială datorită facilităților oferite de Neural Network Toolbox şi Graphical User Interface Design Environment.
- Se propune şi o soluție hardware bazată pe procesorul de semnal de uz general, preț scăzut, TMS320VC33 folosit ca şi coprocesor în conjuncție cu un microcontroller tip 8051 [102], [103].
- În final se realizează un sistem de recunoaştere facială capabil să opereze în timp real. Codul sursă MATLAB aferent acestuia este prezentat în ANEXA 5.

8.3 Direcţii viitoare de cercetare

Problema recunoaşterii faciale este studiată de către autorul tezei, ca de altfel şi de cvasitotalitatea celorlalte abordări din literatura specifică acestui domeniu, din perspectiva prelucrării imaginilor reprezentate în tonuri de gri.

O posibilă direcţie de investigaţie ar reprezenta-o prelucrarea imaginilor color, având în vedere că la ora actuală majoritatea camerelor de luat vederi oferă posibilitatea achiziţiei acestor tipuri de imagini. Este probabil, ca prin tehnici de extragere a trăsăturilor adecvate, să se obţină informaţie suplimentară faţă de cazul imaginilor alb-negru care să ajute în final la îmbunătăţirea performanţelor etapei de clasificare.

ANEXA 1

A1.1 Definiție controler fuzzy.

A1.2 Funcție de antrenament fuzzy (trainfuzzy) pentru RNA tip feedforward definită conform standardului MATLAB.

A.1.1 Definiție controler fuzzy

```
[System]
Name='controler0'
Type='mamdani' % tip controler
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=25
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid' % metoda de defuzzyficare

[Input1]
Name='p'
Range=[0.5 1.5]
NumMFs=5
MF1='p1':'trapmf',[0.5 0.5 0.7 0.85]
MF2='p2':'trimf',[0.75 0.85 1]
MF3='p3':'trimf',[0.95 1 1.05]
MF4='p4':'trimf',[1 1.15 1.25]
MF5='p5':'trapmf',[1.15 1.3 1.5 1.5]

[Input2]
Name='dp'
Range=[-0.05 0.05]
NumMFs=5
MF1='dp1':'trapmf',[-0.05 -0.05 -0.03 -0.02]
MF2='dp2':'trimf',[-0.025 -0.015 -0.005]
MF3='dp3':'trimf',[-0.01 0 0.01]
MF4='dp4':'trimf',[0.005 0.015 0.025]
MF5='dp5':'trapmf',[0.02 0.03 0.05 0.05]

[Output1]
Name='c'
Range=[0.4 1.3]
NumMFs=5
MF1='c1':'trapmf',[0.4 0.4 0.65 0.85]
MF2='c2':'trimf',[0.75 0.85 1]
MF3='c3':'trimf',[0.95 1 1.05]
MF4='c4':'trimf',[1 1.1 1.2]
```

```
MF5='c5':'trapmf',[1.1 1.2 1.3 1.3]

[Rules]
1 1, 5 (1) : 1
1 2, 5 (1) : 1
1 3, 4 (1) : 1
1 4, 3 (1) : 1
1 5, 2 (1) : 1
2 1, 5 (1) : 1
2 2, 4 (1) : 1
2 3, 4 (1) : 1
2 4, 2 (1) : 1
2 5, 2 (1) : 1
3 1, 4 (1) : 1
3 2, 4 (1) : 1
3 3, 4 (1) : 1
3 4, 2 (1) : 1
3 5, 1 (1) : 1
4 1, 3 (1) : 1
4 2, 2 (1) : 1
4 3, 2 (1) : 1
4 4, 2 (1) : 1
4 5, 1 (1) : 1
5 1, 2 (1) : 1
5 2, 2 (1) : 1
5 3, 1 (1) : 1
5 4, 1 (1) : 1
5 5, 1 (1) : 1
```

A1.2 Funcție de antrenament fuzzy (trainfuzzy) pentru RNA tip feedforward definită conform standardului MATLAB.

```
function [net,tr] = trainfuzzy(net,Pd,Tl,Ai,Q,TS,VV,TV)
fismat = readfis('controler0');

if isstr(net)
  switch (net)
    case 'pnames',
      net = fieldnames(trainfuzzy('pdefaults'));
    case 'pdefaults',
     trainParam.epochs = 200;
     trainParam.goal = 0;
     trainParam.lr = 0.01;
     c=1;
     trainParam.lr_dec = 0.7;
     trainParam.lr_inc = 1.05;
     trainParam.max_fail = 5;
     trainParam.max_perf_inc = 1.04;
     trainParam.mc = 0.9;
     trainParam.min_grad = 1.0e-6;
     trainParam.show = 25;
     trainParam.time = inf;
     net = trainParam;
    otherwise
      error('Unrecognized code.')
```

```
  end
  return
end

% CALCULATION
% ===========

% Constants
this = 'trainfuzzy0';
epochs = net.trainParam.epochs;
goal = net.trainParam.goal;
lr = net.trainParam.lr;
lr_inc = net.trainParam.lr_inc;
lr_dec = net.trainParam.lr_dec;
max_fail = net.trainParam.max_fail;
max_perf_inc = net.trainParam.max_perf_inc;
mc = net.trainParam.mc;
min_grad = net.trainParam.min_grad;
show = net.trainParam.show;
time = net.trainParam.time;
doValidation = ~isempty(VV);
doTest = ~isempty(TV);

% Initialize
stop = '';
startTime = clock;
X = getx(net);
[perf,E,Ac,N,Zb,Zi,Zl] = calcperf(net,X,Pd,Tl,Ai,Q,TS);
perf1=perf;
[gX,normgX] = calcgx(net,X,Pd,Zb,Zi,Zl,N,Ac,E,perf,Q,TS);
dX = lr*gX;
if (doValidation)
  VV.net = net;
  VV.perf = calcperf(net,X,VV.Pd,VV.Tl,VV.Ai,VV.Q,VV.TS);
  VV.numFail = 0;
end
tr = newtr(epochs,'perf','vperf','tperf','lr');
performanta=1;
dperformanta=0;
% Train
for epoch=0:epochs

  % Training Record
  epochPlus1 = epoch+1;
  tr.perf(epochPlus1) = perf;
  tr.lr(epochPlus1) = lr;
  if (doValidation)
    tr.vperf(epochPlus1) = VV.perf;
  end
  if (doTest)
    tr.tperf(epochPlus1) =
calcperf(net,X,TV.Pd,TV.Tl,TV.Ai,TV.Q,TV.TS);
  end

  % Stopping Criteria
  currentTime = etime(clock,startTime);
  if (perf <= goal)
```

```
    stop = 'Performance goal met.';
  elseif (epoch == epochs)
    stop = 'Maximum epoch reached, performance goal was not met.';
  elseif (currentTime > time)
    stop = 'Maximum time elapsed, performance goal was not met.';
  elseif (normgX < min_grad)
    stop = 'Minimum gradient reached, performance goal was not met.';
  elseif (doValidation) & (VV.numFail > max_fail)
    stop = 'Validation stop.';
   net = VV.net;
  end

  % Progress
  if ~rem(epoch,show) | length(stop)
    fprintf(this);
   if isfinite(epochs) fprintf(', Epoch %g/%g',epoch, epochs); end
   if isfinite(time) fprintf(', Time %g%%',currentTime/time/100); end
   if isfinite(lr) fprintf(', lr %g%',lr); end
   if isfinite(lr) fprintf(', c %g%',c); end
   if isfinite(lr) fprintf(', perf/dperf
%g/%g',performanta,dperformanta); end
   if isfinite(goal) fprintf(', %s
%g/%g',upper(net.performFcn),perf,goal); end
   if isfinite(min_grad) fprintf(', Gradient %g/%g',normgX,min_grad);
end
   fprintf('\n')
   plotperf(tr,goal,this,epoch)
    if length(stop) fprintf('%s, %s\n\n',this,stop); break; end
  end

% Gradient Descent with Momentum and Adaptive Learning Rate

  dX = mc*dX + (1-mc)*lr*gX;
  X2 = X + dX;
  net2 = setx(net,X2);
  [perf2,E,Ac,N,Zb,Zi,Zl] = calcperf(net2,X2,Pd,Tl,Ai,Q,TS);
  performanta=(perf2/perf);
  dperformanta=(performanta-(perf/perf1));

  if  performanta > 1.5
     performanta=1.49;
  end
  if  performanta < 0.5
     performanta=0.51;
  end
  if  dperformanta > 0.05
     dperformanta=0.049;
  end
  if  dperformanta < -0.05
      dperformanta=-0.049;
  end
  c=evalfis([performanta dperformanta],fismat);
 lr=lr*c;

  if  performanta >= 1.04
    dX = lr*gX;
    perf1=perf;
```

```
    %dperformanta=1;
    else
     if (perf2 < perf)
     end
   X = X2;
   net = net2;
   perf1=perf;
   perf = perf2;
   [gX,normgX] = calcgx(net,X,Pd,Zb,Zi,Zl,N,Ac,E,perf,Q,TS);
   norm_g = sqrt(sum(sum(gX.^2)));
end
  % Validation
  if (doValidation)
    vperf = calcperf(net,X,VV.Pd,VV.Tl,VV.Ai,VV.Q,VV.TS);
   if (vperf < VV.perf)
     VV.perf = vperf; VV.net = net; VV.numFail = 0;
   elseif (vperf > VV.perf)
      VV.numFail = VV.numFail + 1;
   end
  end
end

% Finish
tr = cliptr(tr,epoch);
```

ANEXA 2

A2.1 Comparația metodei "trainfuzzy" cu alte metode de antrenament în problema aproximării unei funcții.

A2.2 Comparația metodei "trainfuzzy" cu alte metode de antrenament în problema mine versus stânci.

A2.3 Comparația metodei "trainfuzzy" cu alte metode de antrenament în probleme de clasificare a tiparelor.

A2.1 Comparația metodei "trainfuzzy" cu alte metode de antrenament în problema aproximării unei funcții.

```
% Comparatie intre algoritmi pentru problema
% aproximarii unei functii
% @ CATALIN-DANIEL CALEANU, 2000

clear all;
P = 0:0.1:1; % suportul functiei, faza de antrenament
T = 0.2 + 0.8*(P + 0.7*sin(2*pi*P)); % functie
testx=0:0.01:1; % suportul functiei, faza de test
testy=0.2 + 0.8*(testx + 0.7*sin(2*pi*testx));

% Implementarea si antrenamentul RNA, met. gradient descendent
net1= newff(minmax(P),[9 1],{'tansig' 'logsig'},'traingd');
net1.trainParam.lr=0.01;
net1.trainParam.mc=0.3;
net1.trainParam.min_grad=1e-10;
net1.trainParam.show = 1;
net1.trainParam.epochs = 150;
net1.trainParam.goal = 0.01;
t1 = clock;
[net1,tr1]= train(net1,P,T);
time1=etime(clock,t1);

% Implementarea si antrenamentul RNA, met. delta-delta
net2= newff(minmax(P),[9 1],{'tansig' 'logsig'},'traingdx');
net2.trainParam.lr = net1.trainParam.lr;
net2.trainParam.mc = net1.trainParam.mc;
net2.trainParam.min_grad = net1.trainParam.min_grad;
net2.trainParam.show = net1.trainParam.show;
net2.trainParam.epochs = net1.trainParam.epochs;
net2.trainParam.goal = net1.trainParam.goal;
t2 = clock;
[net2,tr2]= train(net2,P,T);
```

```
time2=etime(clock,t2);

% Implementarea si antrenamentul RNA, met. fuzzy
net3= newff(minmax(P),[9 1],{'tansig' 'logsig'},'trainfuzzy');
net3.trainParam.lr = net1.trainParam.lr;
net3.trainParam.mc = net1.trainParam.mc;
net3.trainParam.min_grad = net1.trainParam.min_grad;
net3.trainParam.show = net1.trainParam.show;
net3.trainParam.epochs = net1.trainParam.epochs;
net3.trainParam.goal = net1.trainParam.goal;
t3 = clock;
[net3,tr3]= train(net3,P,T);
time3=etime(clock,t3);


y1test = sim(net1,testx);
y2test = sim(net2,testx);
y3test = sim(net3,testx);
y1antr = sim(net1,P);
y2antr = sim(net2,P);
y3antr = sim(net3,P);


er1test=(sum(sum(abs(ones(1,length(testy))-
(y1test/testy)))))/length(testy);
er2test=(sum(sum(abs(ones(1,length(testy))-
(y2test/testy)))))/length(testy);
er3test=(sum(sum(abs(ones(1,length(testy))-
(y3test/testy)))))/length(testy);
ertest=[er1test er2test er3test];
er1antr=(sum(sum(abs(ones(1,length(T))-(y1antr/T)))))/length(T);
er2antr=(sum(sum(abs(ones(1,length(T))-(y2antr/T)))))/length(T);
er3antr=(sum(sum(abs(ones(1,length(T))-(y3antr/T)))))/length(T);
erantr=[er1antr er2antr er3antr];

time=[time1 time2 time3]

clf;
figure
plot(testx,testy,'k');
title('Aproximare functiei 0.2 + 0.8*(x + 0.7*sin(2*pi*x))');
hold
plot(testx,y1test,':');
plot(testx,y2test,'--');
plot(testx,y3test,'.-');
xlabel('x');
ylabel('traingd ..  traingdx --  trainfuzzy0 .-')
hold off
figure
semilogy(tr1.epoch,tr1.perf,':');
hold
semilogy(tr2.epoch,tr2.perf,'--');
semilogy(tr3.epoch,tr3.perf,'.-');
title('EPM versus numar de epoci');
xlabel('Numar de epoci');
ylabel('traingd ..  traingdx --  trainfuzzy0 .-')
hold off
```

```
figure
bar((ertest)*100);
ylabel('Eroare aproximare test [%]');
xlabel('traingd 1 traingdx 2 trainfuzzy1 3')
figure
bar((erantr)*100);
ylabel('Eroare aproximare antrenament [%]');
xlabel('traingd 1 traingdx 2 trainfuzzy0 3')
figure
barh(time,'r');
xlabel('Timp prelucrare [s]');
ylabel('traingd 1 traingdx 2 trainfuzzy0 3')
figure
plot(tr1.epoch,tr1.lr,':');
hold
plot(tr2.epoch,tr2.lr,'--');
plot(tr3.epoch,tr3.lr,'.-');
xlabel('Numar de epoci');
ylabel('traingd ..  traingdx --  trainfuzzy0 .-')
hold off


ertest=[er1test er2test er3test]*100
erantr=[er1antr er2antr er3antr]*100
```

A2.2 Comparaţia metodei "trainfuzzy" cu alte metode de antrenament în problema mine versus stânci

```
% Comparatie intre algoritmi pentru problema
% mine vs stanci
% @ CATALIN-DANIEL CALEANU, 2000

clear all;
clc;
cmu;
A=1;B=0;C=cat(2,B,A);
D=C;
for i = 1:(60-1)
   D=cat(1,C,D);
end
Nhid=20;
Nout=1;

net1= newff(D,[Nhid Nout],{'logsig' 'tansig'},'traingd');
net1.LW{2,1} = net1.LW{2,1}*0.1;
net1.b{2} = net1.b{2}*0.1;
net1.trainParam.lr = 0.1;
net1.trainParam.mc = 0.9;
net1.trainParam.show = 200;
net1.trainParam.epochs = 800;
net1.trainParam.goal = 0.05;

net2= newff(D,[Nhid Nout],{'logsig' 'tansig'},'traingdx');
net2.LW{2,1} = net2.LW{2,1}*0.1;
net2.b{2} = net2.b{2}*0.1;
```

```
net2.trainParam.lr=net1.trainParam.lr;
net2.trainParam.mc=net1.trainParam.mc;
net2.trainParam.show = net1.trainParam.show;
net2.trainParam.epochs = net1.trainParam.epochs;
net2.trainParam.goal = net1.trainParam.goal;

net3= newff(D,[Nhid Nout],{'logsig' 'tansig'},'trainfuzzy');
net3.LW{2,1} = net3.LW{2,1}*0.1;
net3.b{2} = net3.b{2}*0.1;
net3.trainParam.lr=net1.trainParam.lr;
net3.trainParam.mc=net1.trainParam.mc;
net3.trainParam.show = net1.trainParam.show;
net3.trainParam.epochs = net1.trainParam.epochs;
net3.trainParam.goal = net1.trainParam.goal;

t1=clock;
[net1,tr1]= train(net1,x_train,y_train_num);
time1=etime(clock,t1);
t2 = clock;
[net2,tr2]= train(net2,x_train,y_train_num);
time2=etime(clock,t2);
t3 = clock;
[net3,tr3]= train(net3,x_train,y_train_num);
time3=etime(clock,t3);
time=[time1 time2 time3]

yantr1 = hardlims(sim(net1,x_train));
erantr1=((sum(abs(yantr1-y_train_num)))/2)/length(y_train_num);
yantr2 = hardlims(sim(net2,x_train));
erantr2=((sum(abs(yantr2-y_train_num)))/2)/length(y_train_num);
yantr3 = hardlims(sim(net3,x_train));
erantr3=((sum(abs(yantr3-y_train_num)))/2)/length(y_train_num);

ytest1 = hardlims(sim(net1,x_test));
ertest1=((sum(abs(ytest1-y_test_num)))/2)/length(y_test_num);
ytest2 = hardlims(sim(net2,x_test));
ertest2=((sum(abs(ytest2-y_test_num)))/2)/length(y_test_num);
ytest3 = hardlims(sim(net3,x_test));
ertest3=((sum(abs(ytest3-y_test_num)))/2)/length(y_test_num);

erantr=[erantr1 erantr2 erantr3]*100
ertest=[ertest1 ertest2 ertest3]*100

clf;
figure
semilogy(tr1.epoch,tr1.perf,':');
hold
semilogy(tr2.epoch,tr2.perf,'--');
semilogy(tr3.epoch,tr3.perf,'.-');
title('MSE versus number of epochs');
xlabel('Number of epochs');
ylabel('traingd ..  traingdx --  trainfuzzy .-')
hold off
figure
bar(ertest);
ylabel('Eroare aproximare test [%]');
xlabel('traingd 1 traingdx 2 trainfuzzy 3')
```

```
figure
bar(erantr);
ylabel('Eroare aproximare antrenament [%]');
xlabel('traingd 1 traingdx 2 trainfuzzy 3')
figure
barh(time,'r');
xlabel('Timp prelucrare [s]');
ylabel('traingd 1 traingdx 2 trainfuzzy 3')
figure
semilogy(tr1.epoch,tr1.lr,':');
hold
semilogy(tr2.epoch,tr2.lr,'--');
semilogy(tr3.epoch,tr3.lr,'.-');
title('Learning rate versus number of epochs');
ylabel('traingd ..  traingdx --  trainfuzzy .-')
xlabel('Number of epochs');
hold off
```

A2.3 Comparația metodei "trainfuzzy" cu alte metode de antrenament în probleme de clasificare a tiparelor

```
% Comparatie intre algoritmi pentru problema
% recunoasterii tiparelor de mari dimensiuni
% @ CATALIN-DANIEL CALEANU, 2000

clear all;

A=255;B=0;C=cat(2,B,A);
D=C;
for i = 1:(46*56-1)
   D=cat(1,C,D);
end

antr=[ 11  72  93  74  25  106  47  18  49  810 ...
      101  42  23  24  35   86  77  48  29  410 ...
       41  32  53  34  65   76  97  98  79  910 ...
       51  82  73 104  55   26  67 108  19  310 ...
       21  52  63  44  75   16 107  78  89 1010 ];
for i=1:length(antr)
   P{i}=double((imread(strcat(int2str(antr(i)),'.bmp'))));
   P{i}=P{i}(:);
end
Pantr=cell2mat(P);
T=double(eye(10));
Tantr=[T T T T T];

k=1;
for i=1:10, %pozitia
   for j=1:10, %persoana

Q{k}=double((imread(strcat(strcat(int2str(i),int2str(j)),'.bmp'))));
      Q{k}=Q{k}(:);
      k=k+1;
   end
end
```

```
Ptot=cell2mat(Q);
Ttot=[T T T T T T T T T T];

net1= newff(D,[50 10],{'tansig' 'purelin'},'traingd');
net1.LW{2,1} = net1.LW{2,1}*0.1;
net1.b{2} = net1.b{2}*0.1;
net1.trainParam.lr = 0.1;
net1.trainParam.mc = 0.9
;
net1.trainParam.show = 1;
net1.trainParam.epochs = 80;
net1.trainParam.goal = 0.03;

net2= newff(D,[50 10],{'tansig' 'purelin'},'traingdx');
net2.LW{2,1} = net2.LW{2,1}*0.1;
net2.b{2} = net2.b{2}*0.1;
net2.trainParam.lr=net1.trainParam.lr;
net2.trainParam.mc=net1.trainParam.mc;
net2.trainParam.show = net1.trainParam.show;
net2.trainParam.epochs = net1.trainParam.epochs;
net2.trainParam.goal = net1.trainParam.goal;

net3= newff(D,[50 10],{'tansig' 'purelin'},'trainfuzzy');
net3.LW{2,1} = net3.LW{2,1}*0.1;
net3.b{2} = net3.b{2}*0.1;
net3.trainParam.lr=net1.trainParam.lr;
net3.trainParam.mc=net1.trainParam.mc;
net3.trainParam.show = net1.trainParam.show;
net3.trainParam.epochs = net1.trainParam.epochs;
net3.trainParam.goal = net1.trainParam.goal;

t1=clock;
[net1,tr1]= train(net1,Pantr,Tantr);
time1=etime(clock,t1);
t2 = clock;
[net2,tr2]= train(net2,Pantr,Tantr);
time2=etime(clock,t2);
t3 = clock;
[net3,tr3]= train(net3,Pantr,Tantr);
time3=etime(clock,t3);
time=[time1 time2 time3]

yantr1 = compet(sim(net1,Pantr));
erantr1=(sum(sum(abs(yantr1-Tantr)))/2)/length(Tantr);
yantr2 = compet(sim(net2,Pantr));
erantr2=(sum(sum(abs(yantr2-Tantr)))/2)/length(Tantr);
yantr3 = compet(sim(net3,Pantr));
erantr3=(sum(sum(abs(yantr3-Tantr)))/2)/length(Tantr);

ytot1 = compet(sim(net1,Ptot));
ertot1=(sum(sum(abs(ytot1-Ttot)))/2)/length(Ttot);
ytot2 = compet(sim(net2,Ptot));
ertot2=(sum(sum(abs(ytot2-Ttot)))/2)/length(Ttot);
ytot3 = compet(sim(net3,Ptot));
ertot3=(sum(sum(abs(ytot3-Ttot)))/2)/length(Ttot);

erantr=[erantr1 erantr2 erantr3]*100
```

```
ertot=[ertot1 ertot2 ertot3]*100
ertest=[((100*ertot1-50*erantr1)/50)*100 ...
        ((100*ertot2-50*erantr2)/50)*100 ...
        ((100*ertot3-50*erantr3)/50)*100]

clf;
figure
semilogy(tr1.epoch,tr1.perf,':');
hold
semilogy(tr2.epoch,tr2.perf,'--');
semilogy(tr3.epoch,tr3.perf,'.-');
title('MSE versus number of epochs');
xlabel('Number of epochs');
ylabel('traingd ..  traingdx --  trainfuzzy .-')
hold off
figure
bar(ertest);
ylabel('Eroare aproximare test [%]');
xlabel('traingd 1 traingdx 2 trainfuzzy 3')
figure
bar(erantr);
ylabel('Eroare aproximare antrenament [%]');
xlabel('traingd 1 traingdx 2 trainfuzzy 3')
figure
barh(time,'r');
xlabel('Timp prelucrare [s]');
ylabel('traingd 1 traingdx 2 trainfuzzy 3')
figure
semilogy(tr1.epoch,tr1.lr,':');
hold
semilogy(tr2.epoch,tr2.lr,'--');
semilogy(tr3.epoch,tr3.lr,'.-');
title('Learning rate versus number of epochs');
ylabel('traingd ..  traingdx --  trainfuzzy .-')
xlabel('Number of epochs');
hold off
```

ANEXA 3

A3.1. Studiul influenței folosirii tehnicii de extragere a trăsăturilor de tip operator de interes asupra procesului de recunoaştere facială.

A3.2. Influența rezoluției imaginii faciale și a dimensiunilor ferestrei de extragere a trăsăturilor asupra performanțelor procesului de recunoaştere facială.

a) Program principal

```
% 1 RNA-MLP fara/cu extragere trasaturi
% @ CATALIN-DANIEL CALEANU, 2000.

clear all
clc;
goal=0.01    % eroarea patratica medie (EPM)
nrhid1=80    % nr. neuroni ascunsi
nrexperim=1 % numar total de experimente
t0 = clock; % pornire cronometru

for contorret=1:nrexperim

   disp('Extragere trasaturi pentru experimentul nr:')
   disp(contorret)
   gentipaleat; % generare aleatoare de vectori de antrenament si test

   ultimaperf=1;
      while ultimaperf>goal
         % se creeaza RNA-MLP
         net1= newff(D,[nrhid1 nrpersantr],{'tansig'
         'purelin'},'trainscg');
         net1.LW{2,1} = net1.LW{2,1}*0.1;
         net1.b{2} = net1.b{2}*0.1;
         % se specifica parametrii procesului de antrenament
         net1.trainParam.lr = 0.1;
         net1.trainParam.mc = 0.9;
         net1.trainParam.show = 10;
         net1.trainParam.epochs = 550;
         net1.trainParam.goal = goal;
         % se antreneaza RNA
         [net1,tr1]= train(net1,Pantr,Tantr);
         ultimaperf=tr1.perf(max(tr1.epoch+1));
            if ultimaperf>goal    % daca nu s-a ajuns la EPM
               clear net1;        % se reia procesul de antrenament
            end % end if
      end % end while

   % se calculeaza ies. RNA si eroarea pentru datele de antrenament
   yantr{contorret} = compet(sim(net1,Pantr));
```

```
   erantr(contorret)=(sum(sum(abs(yantr{contorret}-
Tantr)))/2)/length(Tantr);
   % se calculeaza ies. RNA si eroarea pentru datele globale
(antrenament+test)
   ytot{contorret} = compet(sim(net1,Ptot));
   ertot(contorret)=(sum(sum(abs(ytot{contorret}-
Ttot)))/2)/length(Ttot);

   % se calculeaza erorile exprimate in procente
   ertest(contorret)=((nrpersantr*nrpoz*ertot(contorret)- ...
   nrpersantr*nrpozantr*erantr(contorret))/(nrpersantr*nrpoz-
nrpersantr*nrpozantr))*100
   erantr(contorret)=erantr(contorret)*100
   ertot(contorret)=ertot(contorret)*100
   clear net1;

end % end for    - sfarsit experiment

% se calculeaza erorile medii dupa numarul "nrexperim" de experimente
ertestmed=( sum(ertest) )/nrexperim
erantrmed=( sum(erantr) )/nrexperim
ertotmed =( sum(ertot)  )/nrexperim
% durata tuturor experimentelor, in minute
totalminute=(etime(clock,t0))/60
% se salveaza intr-un fisier marimile de interes
save rezultate.mat nrpersantr nrpozantr reesantionare extrtras nrhid1
nrexperim ertestmed erantrmed ertotmed ertest erantr ertot dimferx
dimfery totalminute
```

b) Generarea vectorilor de antrenament şi test

```
% Generare aleatoare de vectori de antrenament si test
% din baza de date ORL, cu/fara extragere trasaturi
% @ CATALIN-DANIEL CALEANU, 2000.

clc

nrpers=40;      % nr. total de pers. din baza de date (BD)
nrpoz=10;       % nr. de imagini disponibile/persoana
nrpersantr=40; % nr. de persoane pentru setul de antrenament
nrpozantr=5;    % nr. de imagini/persoana de antrenament

locatie='d:\ATT112x92PNG\s'; % locatie baza de date
reesantionare='da'; % 'da'/'nu' - reesantionare/fara reesantionare
extrtras='da';      % 'da'/'nu' - cu/fara extragere de trasaturi
Ta=double(eye(nrpersantr)); % vector de iesire pentru RNA

if (reesantionare=='da')
   nrlin=56; % nr. de linii dupa reesantionare
   nrcol=46; % nr. de coloane dupa reesantionare
end
if (extrtras=='da')
   % img va fi procesata in format nrcol x nrlin !
   % dimensiunea ferestrei de extragere a trasaturilor
```

```
   dimferx=8;
   dimfery=8;
end

rand('state',sum(100*clock));    % se init. gen. de nr. aleat.
persid=randperm(nrpers);         % ordonare aleator nrpers
pozid=randperm(nrpoz);           % ordonare aleator nrpoz

persidantr=persid(1:nrpersantr);
pozidantr=pozid(1:nrpozantr);

for i=1:nrpers
   %se creeaza sir de caractere cu toate dir
   director{i}=strcat(locatie,int2str(i));
end

% date antrenament
k=1;
for j=1:nrpozantr
   for i=1:nrpersantr
   cale=strcat(director(persidantr(i)),'\');
   numefis=int2str(pozidantr(j));
   cale_numefis=strcat(cale,numefis);
   P{k}=(double(imread(cell2mat(cale_numefis),'png'))');
      if (reesantionare=='da')
         P{k} = imresize(P{k},[nrcol nrlin],'nearest');
      end
      if (extrtras=='da')
         % extragere trasaturi prin met. operatorului de interes
         P{k}=blkproc(P{k},[dimferx dimfery],'interestop(x)');
      end
   P{k}=P{k}(:);
   k=k+1;
   end
end
Pantr=cell2mat(P); % vectorii de intrare in etapa de antrenament
Tantr=Ta;
for i=1:nrpozantr-1
   Tantr=cat(2,Tantr,Ta); % vectorii de iesire in etapa de antrenament
end

%date totale
k=1;
for j=1:nrpoz
   for i=1:nrpersantr
   cale=strcat(director(persidantr(i)),'\');
   numefis=int2str(j);
   cale_numefis=strcat(cale,numefis);
   Q{k}=(double(imread(cell2mat(cale_numefis),'png'))');
      if (reesantionare=='da')
         Q{k} = imresize(Q{k},[nrcol nrlin],'nearest');
      end
      if (extrtras=='da')
         % extragere trasaturi prin met. operatorului de interes
         Q{k}=blkproc(Q{k},[dimferx dimfery],'interestop(x)');
      end
   Q{k}=Q{k}(:);
```

```
    k=k+1;
    end
end
Ptot=cell2mat(Q); % vectorii de intrare pentru intregul set de date
Ttot=Ta;
for i=1:nrpoz-1
    Ttot=cat(2,Ttot,Ta); % vectorii de iesire pentru intregul set de
date
end

% matrice min-max pentru intrarile RNA
A=max(max(Ptot));B=min(min(Ptot));C=cat(2,B,A);
D=C;
for i = 1:(size(Pantr,1)-1)
    D=cat(1,C,D);
end
```

c) Funcţia ce implementează extragerea trăsăturilor prin metoda operatorului de interes

```
function [y]=interestop(x)
% Implementarea metodei de extragere a trasaturilor prin
% METODA OPERATORULI DE INTERES
% Functia returneaza un vector 5 dimensional [s,sh,sv,s135,s45]
% ce exprima variantele directionale
% pentru fereastra "x" dintr-o imagine
% @ CATALIN-DANIEL CALEANU, 2000.


s=0;sh=0;sv=0;s135=0;s45=0;
[a b]=size(x);
s=(std(x(:),1));
for j = 1:a,
              for i = 1:(b-1),
                 sh=sh+(x(j,i+1)-x(j,i))^2;
              end
end
for j = 1:(a-1),
              for i = 1:b,
                sv=sv+(x(j+1,i)-x(j,i))^2;
              end
end
for j = 1:(a-1),
              for i = 1:(b-1),
                  s135=s135+(x(j+1,i+1)-x(j,i))^2;
                  s45=s45+(x(j+1,i)-x(j,i+1))^2;
               end
end
sh=sqrt(sh);sv=sqrt(sv);
s135=sqrt(s135);s45=sqrt(s45);
y=[s,sh,sv,s135,s45];
```

ANEXA 4

A4.1 Influența folosirii unor arhitecturi paralele şi ierarhice de RNA-MLP asupra procesului de recunoaştere facială.

a) Program principal

```
% 3 RNA-MLP distincte, fara/cu extragere trasaturi
% decizie pe baza medierii raspunsurilor sau
% pe baza unei RNA-MLP ierarhic superioara
% @ CATALIN-DANIEL CALEANU, 2000.

clear all
clc;
goal=0.01;    % eroarea patratica medie (EPM)
nrhid1=400;  % nr. neuroni ascunsi pt. RNA 1...4
nrhid2=400;
nrhid3=400;
nrhid4=400;
nrexperim=5; % numar total de experimente
t0 = clock;  % pornire cronometru

for contorret=1:nrexperim

   gentipaleat3; % generare aleatoare de vectori de antrenament si
test

   % se creeaza 3 RNA-MLP dupa care se declanseaza procesul de
antrenament

   ultimaperf=1;
      while ultimaperf>goal
         net1= newff(D1,[nrhid1 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net1.LW{2,1} = net1.LW{2,1}*0.1;
         net1.b{2} = net1.b{2}*0.1;
         net1.trainParam.lr = 0.1;
         net1.trainParam.mc = 0.9;
         net1.trainParam.show = 100;
         net1.trainParam.epochs = 550;
         net1.trainParam.goal = goal;
         [net1,tr1]= train(net1,Pantr1,Tantr1);
         ultimaperf=tr1.perf(max(tr1.epoch+1));
            if ultimaperf>goal
               clear net1;
            end % end if
      end % end while

   ultimaperf=1;
      while ultimaperf>goal
```

```
         net2= newff(D2,[nrhid2 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net2.LW{2,1} = net2.LW{2,1}*0.1;
         net2.b{2} = net2.b{2}*0.1;
         net2.trainParam.lr = 0.1;
         net2.trainParam.mc = 0.9;
         net2.trainParam.show = 100;
         net2.trainParam.epochs = 550;
         net2.trainParam.goal = goal;
         [net2,tr2]= train(net2,Pantr2,Tantr2);
         ultimaperf=tr2.perf(max(tr2.epoch+1));
         if ultimaperf>goal
            clear net2;
         end % end if
      end % end while

   ultimaperf=1;
      while ultimaperf>goal
         net3= newff(D3,[nrhid3 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net3.LW{2,1} = net3.LW{2,1}*0.1;
         net3.b{2} = net3.b{2}*0.1;
         net3.trainParam.lr = 0.1;
         net3.trainParam.mc = 0.9;
         net3.trainParam.show = 100;
         net3.trainParam.epochs = 550;
         net3.trainParam.goal = goal;
         [net3,tr3]= train(net3,Pantr3,Tantr3);
         ultimaperf=tr3.perf(max(tr3.epoch+1));
         if ultimaperf>goal
            clear net3;
         end % end if
      end % end while

% se calculeaza ies. RNA si eroarea pentru datele de antrenament
yantr1{contorret} = sim(net1,Pantr1);
yantr1c{contorret} = compet(sim(net1,Pantr1));
ytot1{contorret} = sim(net1,Ptot1);
ytot1c{contorret} = compet(sim(net1,Ptot1));
ertot1(contorret)=(sum(sum(abs(ytot1c{contorret}-
Ttot1)))/2)/length(Ttot1);
%erantr1=(sum(sum(abs(yantr1c-Tantr1)))/2)/length(Tantr1)

yantr2{contorret} = sim(net2,Pantr2);
yantr2c{contorret} = compet(sim(net2,Pantr2));
ytot2{contorret} = sim(net2,Ptot2);
ytot2c{contorret} = compet(sim(net2,Ptot2));
ertot2(contorret)=(sum(sum(abs(ytot2c{contorret}-
Ttot2)))/2)/length(Ttot2);
%erantr2=(sum(sum(abs(yantr2c-Tantr2)))/2)/length(Tantr2)

yantr3{contorret} = sim(net3,Pantr3);
yantr3c{contorret} = compet(sim(net3,Pantr3));
ytot3{contorret} = sim(net3,Ptot3);
ytot3c{contorret} = compet(sim(net3,Ptot3));
ertot3(contorret)=(sum(sum(abs(ytot3c{contorret}-
Ttot3)))/2)/length(Ttot3);
```

```
%erantr3=(sum(sum(abs(yantr3c-Tantr3)))/2)/length(Tantr3)

% se calculeaza decizia prin medierea raspunsurilor RNA
sumaiesantr=(yantr1{contorret}+yantr2{contorret}+yantr3{contorret})/3;
Pantr4=[yantr1{contorret}; yantr2{contorret}; yantr3{contorret}];
yantr{contorret}=compet(sumaiesantr);
erantr(contorret)=(sum(sum(abs(yantr{contorret}-
Tantr1)))/2)/length(Tantr1);

% se implementeaza RNA ierarhic superioara
sumaiestot=(ytot1{contorret}+ytot2{contorret}+ytot3{contorret})/3;
Ptot4=[ytot1{contorret}; ytot2{contorret}; ytot3{contorret}];
ytot{contorret}=compet(sumaiestot);
ertot(contorret)=(sum(sum(abs(ytot{contorret}-
Ttot1)))/2)/length(Ttot1);

ertest(contorret)=((nrpersantr*nrpoz*ertot(contorret)-
nrpersantr*nrpozantr*erantr(contorret))/(nrpersantr*nrpoz-
nrpersantr*nrpozantr))*100
erantr(contorret)=erantr(contorret)*100
ertot(contorret)=ertot(contorret)*100
clear net1;clear net2;clear net3;

A=max(max(Ptot4));B=min(min(Ptot4));C=cat(2,B,A);
D4=C;
for j = 1:(3*nrpers-1)
   D4=cat(1,C,D4);
end

Tantr4=Tantr1;
Ttot4=Ttot1;

ultimaperf=1;
      while ultimaperf>goal
         net4= newff(D4,[nrhid4 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net4.LW{2,1} = net4.LW{2,1}*0.1;
         net4.b{2} = net4.b{2}*0.1;
         net4.trainParam.lr = 0.1;
         net4.trainParam.mc = 0.9;
         net4.trainParam.show = 100;
         net4.trainParam.epochs = 550;
         net4.trainParam.goal = goal;
         [net4,tr4]= train(net4,Pantr4,Tantr4);
         ultimaperf=tr4.perf(max(tr4.epoch+1));
            if ultimaperf>goal
               clear net4;
            end
      end

% se calculeaza decizia pe baza RNA ierarhic superioara
yantr4{contorret} = sim(net4,Pantr4);
yantr4c{contorret} = compet(sim(net4,Pantr4));
ytot4{contorret} = sim(net4,Ptot4);
ytot4c{contorret} = compet(sim(net4,Ptot4));
erantr4(contorret)=(sum(sum(abs(yantr4c{contorret}-
Tantr1)))/2)/length(Tantr1);
```

```
ertot4(contorret)=(sum(sum(abs(ytot4c{contorret}-
Ttot4)))/2)/length(Ttot4);
ertest4(contorret)=((nrpersantr*nrpoz*ertot4(contorret)-
nrpersantr*nrpozantr*erantr(contorret))/(nrpersantr*nrpoz-
nrpersantr*nrpozantr))*100
ertot4(contorret)=ertot4(contorret)*100
erantr4(contorret)=erantr4(contorret)*100

end % end for -  sfarsit experiment

% se calculeaza erorile medii dupa numarul "nrexperim" de experimente
ertestmed=( sum(ertest) )/nrexperim
erantrmed=( sum(erantr) )/nrexperim
ertotmed =( sum(ertot)  )/nrexperim
ertestmed4=( sum(ertest4) )/nrexperim
erantrmed4=( sum(erantr4) )/nrexperim
ertotmed4 =( sum(ertot4)  )/nrexperim
% durata tuturor experimentelor, in minute
totalminute=(etime(clock,t0))/60
% se salveaza intr-un fisier marimile de interes
save complet.mat
```

b) Generarea vectorilor de antrenament şi test pentru trei regiuni dintr-o imagine facială

```
% Generare aleatoare de vectori de antrenament si test
% din baza de date ORL, cu/fara extragere trasaturi
% pentru 3 regiuni dintr-o imagine faciala
% © CATALIN-DANIEL CALEANU, 2000.

clc

% nr. de linii din subimagine
n1=28;
n2=28;
n3=28;

nrpers=40;      % nr. total de pers. din baza de date (BD)
nrpoz=10;       % nr. de imagini disponibile/persoana
nrpersantr=40; % nr. de persoane pentru setul de antrenament
nrpozantr=3;    % nr. de imagini/persoana de antrenament

locatie='d:\ATT112x92PNG\s'; % locatie baza de date
reesantionare='da'; % 'da'/'nu' - reesantionare/fara reesantionare
extrtras='da';      % 'da'/'nu' - cu/fara extragere de trasaturi
Ta=double(eye(nrpersantr));  % vector de iesire pentru RNA


if (reesantionare=='da')
   nrlin=56; % nr. de linii dupa reesantionare
   nrcol=46; % nr. de coloane dupa reesantionare
end
if (extrtras=='da')
```

```
   %img in format nrlin x nrcol
   % dimensiunea ferestrei de extragere a trasaturilor
   dimferx=6;
   dimfery=7;
end

rand('state',sum(100*clock));    % se initializeaza gen. de nr. aleat.
persid=randperm(nrpers);         % ordonare aleator nrpers
pozid=randperm(nrpoz);           % ordonare aleator nrpoz

persidantr=persid(1:nrpersantr);
pozidantr=pozid(1:nrpozantr);

for i=1:nrpers
   % se creeaza sir de caractere cu toate directoarele
   director{i}=strcat(locatie,int2str(i));
end

% date antrenament
k=1;
for j=1:nrpozantr
   for i=1:nrpersantr
   cale=strcat(director(persidantr(i)),'\');
   numefis=int2str(pozidantr(j));
   cale_numefis=strcat(cale,numefis);
   P{k}=(double(imread(cell2mat(cale_numefis),'png'))); % nu apare '
      if (reesantionare=='da')
         P{k} = imresize(P{k},[nrlin nrcol],'nearest');
      end
      [row colum]=size(P{k});
      if (extrtras=='da')
         % extragere trasaturi prin met. operatorului de interes
         P1{k}=blkproc(P{k}(1:n1,1:colum),[dimferx
dimfery],'interestop(x)');
         P2{k}=blkproc(P{k}((row-n2)/2+1:(row-
n2)/2+n2,1:colum),[dimferx dimfery],'interestop(x)');
         P3{k}=blkproc(P{k}(row-n3+1:row,1:colum),[dimferx
dimfery],'interestop(x)');
      end
      P1{k}=P1{k}(:);
      P2{k}=P2{k}(:);
      P3{k}=P3{k}(:);
   k=k+1;
   end
end
% vectorii de intrare in etapa de antrenament
Pantr1=cell2mat(P1);
Pantr2=cell2mat(P2);
Pantr3=cell2mat(P3);

Tantr=Ta;
for i=1:nrpozantr-1
   Tantr=cat(2,Tantr,Ta);
end
% vectorii de iesire in etapa de antrenament
Tantr1=Tantr;
Tantr2=Tantr;
```

```
Tantr3=Tantr;

% date totale
k=1;
for j=1:nrpoz
   for i=1:nrpersantr
   cale=strcat(director(persidantr(i)),'\');
   numefis=int2str(j);
   cale_numefis=strcat(cale,numefis);
   Q{k}=(double(imread(cell2mat(cale_numefis),'png')));
      if (reesantionare=='da')
         Q{k} = imresize(Q{k},[nrlin nrcol],'nearest');
      end
      if (extrtras=='da')
         % extragere trasaturi prin met. operatorului de interes
         Q1{k}=blkproc(Q{k}(1:n1,1:colum),[dimferx
dimfery],'interestop(x)');
         Q2{k}=blkproc(Q{k}((row-n2)/2+1:(row-
n2)/2+n2,1:colum),[dimferx dimfery],'interestop(x)');
         Q3{k}=blkproc(Q{k}(row-n3+1:row,1:colum),[dimferx
dimfery],'interestop(x)');
      end
      Q1{k}=Q1{k}(:);
      Q2{k}=Q2{k}(:);
      Q3{k}=Q3{k}(:);

   k=k+1;
   end
end
% vectorii de intrare pentru intregul set de date
Ptot1=cell2mat(Q1);
Ptot2=cell2mat(Q2);
Ptot3=cell2mat(Q3);

Ttot=Ta;
for i=1:nrpoz-1
   Ttot=cat(2,Ttot,Ta);
end
% vectorii de iesire pentru intregul set de date
Ttot1=Ttot;
Ttot2=Ttot;
Ttot3=Ttot;

% specificare val. min si max pentru totalitaea intrarilor

A=max(max(Ptot1));B=min(min(Ptot1));C=cat(2,B,A);
D1=C;
for i = 1:(length(P1{1})-1)
   D1=cat(1,C,D1);
end

A=max(max(Ptot2));B=min(min(Ptot2));C=cat(2,B,A);
D2=C;
for i = 1:(length(P2{1})-1)
   D2=cat(1,C,D2);
end
```

```
A=max(max(Ptot3));B=min(min(Ptot3));C=cat(2,B,A);
D3=C;
for i = 1:(length(P3{1})-1)
   D3=cat(1,C,D3);
end
```

ANEXA 5

A5.1 Subrutinele principale ale programului destinat recunoaşterii faciale bazate pe tehnica de extragere a trăsăturilor tip operator de interes şi clasificatori paraleli de tip neuronal.

a) Program principal

```
% Recunoastere faciala bazata pe RNA
% - Program principal -
% © CATALIN-DANIEL CALEANU, 2000.
clear all
close all
% incarca baza de date faciala
load bazadate;
flag_modificare=1;
% afiseaza GUI
fereastra
% afiseaza imagini faciale
thumbnail(P,nume);
axes(findobj('Tag','Axes0'));
% asteapta comenzi de la utilizator
```

b) Afişare bază de date

```
function thumbnail(x,sir);
% afiseaza in GUI imaginile faciale curente
% se apeleaza de fiecare data
% cand baza de date s-a modificat

Pshow=x;
nshow=sir;

if length(x)<10
for i=length(x)+1:10
   Pshow{i}=255*ones(56,46);
   nshow{i}=' ';
end
end

for i=1:10
   string1=strcat('Axes',int2str(i));
   string2=strcat('StaticText',int2str(i));
   ha(i)=findobj(gcf,'Tag',string1);
   axes(ha(i));
   imshow(Pshow{i});
   set(gca,'Tag',string1);
```

```
   set(findobj(gcf,'Tag',string2),'String',nshow{i});
end
```

c) Extragerea trăsăturilor

```
% extragere trasaturi din imaginile faciale
% prin metoda operatorului de interes
n1=28;
n2=28;
n3=28;

Tantr=double(eye(length(P)));
dimferx=6;
dimfery=7;
k=1;
[row colum]=size(P{k});

if flag_modificare==1

%date antrenament
clear P1 P2 P3 Pantr1 Pantr2 Pantr3;
h = waitbar2(0,'Extragere trasaturi din baza de date...');
for i=1:length(P)
   P1{k}=blkproc(double(P{k}(1:n1,1:colum)),[dimferx
dimfery],'interestop(x)');
   P2{k}=blkproc(double(P{k}((row-n2)/2+1:(row-
n2)/2+n2,1:colum)),[dimferx dimfery],'interestop(x)');
   P3{k}=blkproc(double(P{k}(row-n3+1:row,1:colum)),[dimferx
dimfery],'interestop(x)');
   P1{k}=P1{k}(:);
   P2{k}=P2{k}(:);
   P3{k}=P3{k}(:);
   k=k+1;
   waitbar2(i/length(P),h)
end
close(h)
drawnow;
Pantr1=cell2mat(P1);
Pantr2=cell2mat(P2);
Pantr3=cell2mat(P3);

%specificare val min,max pentru totalitaea intrarilor
A=max(max(Pantr1));B=min(min(Pantr1));C=cat(2,B,A);
D1=C;
for i = 1:(length(P1{1})-1)
   D1=cat(1,C,D1);
end

A=max(max(Pantr2));B=min(min(Pantr2));C=cat(2,B,A);
D2=C;
for i = 1:(length(P2{1})-1)
   D2=cat(1,C,D2);
end
A=max(max(Pantr3));B=min(min(Pantr3));C=cat(2,B,A);
```

```
D3=C;
for i = 1:(length(P3{1})-1)
   D3=cat(1,C,D3);
end

end %flag inscriere

%date test
h = waitbar2(0,'Extragere trasaturi din imaginea de test...');
Ptest1=blkproc(double(img(1:n1,1:colum)),[dimferx
dimfery],'interestop(x)');
waitbar2(0.33,h);
Ptest2=blkproc(double(img((row-n2)/2+1:(row-
n2)/2+n2,1:colum)),[dimferx dimfery],'interestop(x)');
waitbar2(0.66,h);
Ptest3=blkproc(double(img(row-n3+1:row,1:colum)),[dimferx
dimfery],'interestop(x)');
waitbar2(1,h);
Ptest1=Ptest1(:);
Ptest2=Ptest2(:);
Ptest3=Ptest3(:);
close(h);
drawnow;
```

d) Clasificare

```
% procedura de recunoastere implica
% extragerea de trasaturi
% si furnizarea acestora unei structuri
% paralele si ierarhice de RNA-MLP

extrtrasaturi;

if flag_modificare==1
ha11=findobj('Tag','Axes11');
axes(ha11);
goal=0.01;
nrhid1=400;
nrhid2=400;
nrhid3=400;
nrhid4=400;
nrpersantr=length(P);


   ultimaperf=1;
      while ultimaperf>goal
         net1= newff(D1,[nrhid1 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net1.LW{2,1} = net1.LW{2,1}*0.1;
         net1.b{2} = net1.b{2}*0.1;
         net1.trainParam.lr = 0.1;
         net1.trainParam.mc = 0.9;
         net1.trainParam.show = 1;
         net1.trainParam.epochs = 550;
```

```
         net1.trainParam.goal = goal;
         [net1,tr1]= train(net1,Pantr1,Tantr);
         ultimaperf=tr1.perf(max(tr1.epoch+1));
            if ultimaperf>goal
               clear net1;
            end
      end%while

   ultimaperf=1;
      while ultimaperf>goal
         net2= newff(D2,[nrhid2 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net2.LW{2,1} = net2.LW{2,1}*0.1;
         net2.b{2} = net2.b{2}*0.1;
         net2.trainParam.lr = 0.1;
         net2.trainParam.mc = 0.9;
         net2.trainParam.show = 1;
         net2.trainParam.epochs = 550;
         net2.trainParam.goal = goal;
         [net2,tr2]= train(net2,Pantr2,Tantr);
         ultimaperf=tr2.perf(max(tr2.epoch+1));
         if ultimaperf>goal
            clear net2;
         end
      end

   ultimaperf=1;
      while ultimaperf>goal
         net3= newff(D3,[nrhid3 nrpersantr],{'tansig'
'purelin'},'trainscg');
         net3.LW{2,1} = net3.LW{2,1}*0.1;
         net3.b{2} = net3.b{2}*0.1;
         net3.trainParam.lr = 0.1;
         net3.trainParam.mc = 0.9;
         net3.trainParam.show = 1;
         net3.trainParam.epochs = 550;
         net3.trainParam.goal = goal;
         [net3,tr3]= train(net3,Pantr3,Tantr);
         ultimaperf=tr3.perf(max(tr3.epoch+1));
         if ultimaperf>goal
            clear net3;
         end
      end

      set(gca,'Tag','Axes11');
end %if flag

yantr1 = sim(net1,Ptest1);
yantr1c = compet(sim(net1,Ptest1));

yantr2 = sim(net2,Ptest2);
yantr2c = compet(sim(net2,Ptest2));

yantr3 = sim(net3,Ptest3);
yantr3c = compet(sim(net3,Ptest3));

sumaiesantr=(yantr1+yantr2+yantr3)/3;
```

```
if max(sumaiesantr)>=val_slider
yantr=compet(sumaiesantr);
[val poz]=max(yantr);
ha_id=findobj('Tag',strcat('Axes',num2str(poz)));
hi_id=findobj('Parent',ha_id);
temp_CData=get(hi_id,'CData');
for i=1:20
   set(hi_id,'CData',255*ones(56,46));
   pause(0.1);
   set(hi_id,'CData',temp_CData);
   pause(0.1);
end
else
   CreateStruct.WindowStyle='modal';
   CreateStruct.Interpreter='tex';
   msgbox('Nerecunoscut','Nerecunoscut!',CreateStruct);
end %if

axes(findobj('Tag','Axes0'));
flag_modificare=0;
```

e) Ştergerea unei imagini din baza de date

```
% stergere imagine din baza de date
      nume_de_sters=get(findobj(gcbf,'Tag','EditText1'),'String');
      locatie=-1;
      for i=1:length(nume)
         if strcmp(nume{i},nume_de_sters)
            locatie=i;
         end
      end
      if locatie==-1
         errordlg('Nume inexistent!');
      else
         temp_nume=nume;
         temp_P=P;
         for i=locatie:length(nume)-1
            temp_nume{i}=temp_nume{i+1};
            temp_P{i}=temp_P{i+1};
         end
         clear P nume;
         for i=1:length(temp_nume)-1
            P{i}=temp_P{i};
            nume{i}=temp_nume{i};
         end
         close(findobj('Tag','Fig2'));
         thumbnail(P,nume);
      end
      flag_modificare=1;
axes(findobj('Tag','Axes0'));
```

f) Înscrierea unei imagini în baza de date

```
% inscriere in baza de date
nume_nou=get(findobj(gcbf,'Tag','EditText1'),'String');
nume = cat(2,nume,mat2cell(nume_nou));
img=imresize(img,[56 46],'nearest');
P{length(P)+1}=img;
close(findobj('Tag','Fig2'));
thumbnail(P,nume);
flag_modificare=1;
```

g) Selecţie imagine facială

```
function imgret = afisrect(x)
%functia afiseaza imaginea selectata cu mouse-ul

r=getrect;
if [r(4) r(3)]<[56 46]
   msgbox('Fereastra prea mica !');
   imgret=x;
else
   imgret = x(round(r(2)):round(r(2)+r(4)),
round(r(1)):round(r(1)+r(3)));
   imshow(imgret);
   set(findobj('Tag', 'PushRecunoastere'),'Enable','on');
end
```

BIBLIOGRAFIE

[1] H. Wechsler, P. J. Phillips, V. Bruce, F.F. Soulie, T.S. Huang – "*Face Recognition. From theory to Applications*", NATO ASI series, Springer-Verlag, 1998.

[2] A.Petland, T. Starner, N. Etcoff, A. Masoiu, O. Oliyde, M. Turk – "*Experiments with eigenfaces*", Proc. Looking at People Wkshp, Int. Joint Conf. Artificial Intell., Chamberry, Franța, 1993.

[3] R. Chellappa, C.L. Wilson, S. Sirohey – " *Human and machine recognition of faces: A survey*", Proc. IEEE, vol.83, pag.705-740, 1995.

[4] B. Miller – "*Vital signs of identity*", IEEE Spectrum, pag. 22-30, Feb., 1994.

[5] S. Rogers, M. Kabrisky, D. Ruck, M. Oxley – "Neural Networks for Fighting Crime" în J.M. Zurada, R.J. Marks, C.J. Robinson (editori) – "Computational Intelligence Imitating Life", pag.406-415, IEEE Press, NY, USA, 1995.

[6] G.T. Poulton – "*Face Recognition Research at CSIRO*", în H. Wechsler şi colectiv – "*Face Recognition. From theory to Applications*", pag. 599-609, NATO ASI series, Springer-Verlag, 1998.

[7] V.I. Pavlovic, R.Sharma, T.S. Huang – "*Visual interpretation of hand gestures for human-computer interaction: a review*", IEEE Trans. PAMI, vol.19, nr.7, pag.677-696, Iulie, 1997.

[8] I. Craw, P. Cameron – "*Face Recognition by Computer*", Proc. British Machine Vision Conference, pag.489-507, editori D. Hogg şi R. Boyle, Springer Verlag, 1992.

[9] F. Ravaut, G. Stamon – "*Face image processing supporting epileptic seizure analysis*", pag. 610 – 616, în H. Wechsler şi colectiv – "*Face Recognition. From theory to Applications*", NATO ASI series, Springer-Verlag, 1998.

[10] J. Hertz, A. Krogh, R. Palmer – "*Introduction to the Theory of Neural Computation*", Lectures Notes, Santa Fe Institute, Addison-Wesley Publishing

Company, pag.1, 1995.

[11] E. T. Rolls – "*Neurophysiological mechanism underlying face processing within and beyond the temporal cortical visual areas*", Processing the Facial Image, Proc. Royal Soc. Discussion Meeting, editori V. Bruce, A. Cowey, Clarendon Press, Oxford, U.K., 1992.

[12] J. Daugman – "*Face and Gesture Recognition: Overview*", IEEE Trans. PAMI, vol.19, nr.7, pag.675-676, Iulie, 1997.

[13] V. Tiponuţ, C.D. Căleanu – "*Reţele neuronale. Arhitecturi şi algoritmi*", Ed. Politehnica, Timişoara, 2000.

[14] M. Gupta, K. Knopf – "*Neuro – Vision Systems. Principles and Applications*", IEEE Press, pag.14-16, 19, 1994.

[15] R.C. Gonzales, R.E. Woods – "*Digital Image Processing*", Addison-Wesley, 1992.

[16] *** *Special Issue on Applications of Artificial Neural Networks to Image Processing*, IEEE Trans. Image Processing, August, 1998.

[17] B. Widrow, R. Winter, R. Baxter – "*Layered Neral Nets for Pattern Recognition*", în M. Gupta, G. Knopf – "*Neuro-Vision Systems. Principles and Applications*", IEEE Press, pag.375-391, 1994.

[18] Z. Zhang, N. Ansari – "*Structure and Properties of Generalized Adaptive Neural Filters for Signal Enhancement*", IEEE Trans. Neural Networks, pag.857-868, Iulie, 1996.

[19] J. Daugman – "*Complete Discrete 2-D Gabor Transform by Neural Networks for Image Analysis and Compresion*", IEEE Acoust., Speech, Signal Processing, nr.7, 1988.

[20] M. Mocofan, C.D. Căleanu, C. Toma – "*Conscience Algorithm based Neural Network for Unsupervised Texture Image Segmentation*", International Conference Beyond 2000: Engineering Research Strategies, Sibiu, November 25-26, vol.XXXVIII, pag.93-96, ISSN 1221-4949, 1999.

[21] C.D. Căleanu, M. Mocofan – "*Hierarchical Structure of Modular Self-Organizing Neural Networks for Unsupervised Texture Image Segmentation*",

Buletinul Ştiinţific al Universităţii POLITEHNICA Timişoara, pag. 135-141, 2000.

[22] M.Mocofan, C.D. Căleanu, D. Lacrămă, F. Alexa – "*Unsupervised texture image segmentation*", Proc. 5th Seminar on Neural Network Applications in electrical engineering, NEUREL 2000, Belgrad, Yugoslavia., pag. 101-104, 2000.

[23] M.J. Arrowsmith, M.R. Varley, P.D. Picton, and J.D. Heys – "*Hybrid Neural Network System for Texture Analysis*" – comunicare personală.

[24] V. Phoha, W. Oldham – "*Image Recovery and Segmentation Using Competitive Learning in a Layered Network*", IEEE Trans. Neural Networks, pag.843-856, Iulie, 1996.

[25] G. Tonder, J. Kruger – "*Shape Encoding. A Biologically Inspired Method of Transforming Boundary Images into Ensemble of Shape-Related Features*", IEEE Trans. Systems, Man, and Cybernetics, pag.749-759, Oct., 1997.

[26] K. Fukushima – "*Neocognitron: A Hierarchical Network for Visual Pattern Recognition*", în M. Gupta, T. Yamakawa - "*Fuzzy Computing*", Amsterdam, pag.71-88, 1988.

[27] C. Neubauer – "*Evaluation of Convolutional Neural Networks for Visual Recognition*", IEEE Trans. Neural Networks, pag.685-697, Iulie, 1998.

[28] L. Wang, S. Der, N. Nasrabadi – "*Automatic Target Recognition using a Feature Decomposition and Data Decomposition Modular Neural Network*", IEEE Trans. Image Processing, pag.1113-1122, August, 1998.

[29] C.D. Căleanu – "*Realizări şi perspective în prelucrarea digitală a imaginilor cu ajutorul reţelelor neuronale*", Referat doctorat nr.1, UPT, Timişoara, pag. 20-45, 1997.

[30] E. Viennet, F. Soulie – "*Connectionists Methods for Human Face Processing*", în H. Wechsler şi colectiv – "Face Recognition. From theory to Applications", pag. 124-156, NATO ASI series, Springer-Verlag, 1998.

[31] S. Thiria, Y. Lechevallier, O.Gascuel, S. Canut (editori) – "*Statistiques et methodes*

[56] M. Hestenes – "*Conjugate direction methods in optimization*", NY, Springer-Verlag, 1990.

[57] E.M. Johansson, F.U. Dowla, D.M. Goodman – "*Backpropagation learning for multi-layer feed-forward neural networks using the conjugate gradient method*", Int. Journal of Neural Systems, 2(4), pag.291-302, 1991.

[58] C.D. Căleanu – "*Neural system for facial recognition*", Zilele Academice Timişene, Ed. VII, Timişoara 24-25 Mai, 2001 –în curs de publicare.

[59] M.T. Hagan, M. Mehanj – "*Training feedforward networks with the Marquardt algorithm*", IEEE Trans. Neural Networks, vol.5, nr.6, pag. 989-993, 1994.

[60] R. Battiti - "*First and second order methods for learning: Between steepest descent and Newton's method*," Neural Computation, vol. 4, no. 2, pag. 141–166, 1992.

[61] M. T. Hagan, H. B. Demuth, M. H. Beale – "*Neural Network Design*", Boston, MA: PWS Publishing, 1996.

[62] M. Gupta, G. Knopf – "*Neuro-Vision Systems: A Tutorial*", în M. Gupta, G. Knopf (editori) – "*Neuro-Vision Systems. Principles and Applications*", IEEE Press, pag. 4-14, 1994.

[63] T. Kanade - "*Picture processing by computer complex and recognition of human faces*", Ph.D. dissertation, Kyoto Univ., Japan, 1973.

[64] R. Brunelli, T. Poggio - "*Face recognition: Features versus templates*", IEEE Trans. Pattern Analysis and Machine Intelligence, vol.15, pag.1042-1052, 1993.

[65] I.J. Cox, J.Ghosn, P.N. Yianilos – "*Feature-Based Face Recognition Using Mixture-Distance*", Proceedings of the 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pag. 209-216, Iunie 18-20, San Francisco, CA, USA, 1996.

[66] K. Sutherland, D.Renshaw, P.B.Denyer – "*Automatic face recognition*", în Proc. 1st Int. Conf. Intell. Syst. Eng., Piscataway, NJ, IEEE Press, pag. 29-34, 1992.

[67] M. Turk and A. Petland - "*Eigenfaces for face recognition*", J. Cognitive Neurosci., vol.3, pag. 71-86, 1991.

[68] B. Moghaddam, A. Petland – "*Face recognition using view-based and modular*

eigenspaces", în Automat. Syst. Identification Inspection of Humans, vol.2257, 1994.

[69] B. Moghaddam, W. Wahid, A. Petland – "*Beyond Eigenfaces: Probabilistic Matching for Face Recognition*", Proc. Of the 3rd IEEE Int. Conf. On Automatic Face and Gesture Recognition, Nara, Japan, pag. 30-35, 1998.

[70] C. Liu, H. Wechsler – "*Evolution of Optimal Projection Axes (OPA) for Face Recognition*", Proc. Of the 3rd IEEE Int. Conf. On Automatic Face and Gesture Recognition, Nara, Japan, pag. 282-287, 1998.

[71] S.Z. Li, Juwei Lu – "*Face Recognition using the Nearest Feature Line Method*", IEEE Trans. on Neural Networks, vol.10, nr.2, pag.439-443, Martie, 1999.

[72] R.J. Baron – "*Mechanism of human facial recognition*", Int. J. Man Machine Studies, vol.15, pag.137-178, 1981.

[73] S. Ben-Yacoub, Y Abdeljaoued, E. Mayoraz – "*Fusion of Face and Speech Data for Person Identity Verification*", IEEE Trans. on Neural Networks, vol.10, nr.5, pag.1065-1073, Sept., 1999.

[74] M. Lades şi colectiv - "*Distorsion invariant object recognition in the dynamic link architecture*", IEEE Trans. Comput., vol.42, pag.300-311, 1993.

[75] I.S. Lawrence, C.L. Giles, A.C. Tsoi, A.D. Bock - "*Face recognition: A convolutional neural-network approach*", IEEE Trans. on Neural Networks, vol.8, pag. 98-113, 1997.

[76] C. Neubauer - "*Evaluation of convolutional neural networks for visual recognition*", IEEE Trans. Neural Networks, vol.9, no.4, pag. 685-695, Iulie, 1998.

[77] S.H. Lin, S.Y. Kung, L.J. Lin - "*Face recognition/detection by probabilistic decision-based neural networks*", IEEE Trans. Neural Networks, vol.8, pag.114-113, 1997.

[78] J. Wilder – "*Face recognition using transform coding of gray scale projections and the neural tree network*", în R.J. Mammone – "Artificial Neural Networks for Speech and Vision", Chapman&Hall, London, pag.521-535, 1994.

[79] A.J. Howell, H. Buxton – "*Recognition People and Behaviours*", în H. Wechsler şi

colectiv – "*Face recognition. From Theory to Applications*", NATO ASI Series, Springer-Verlag, pag.480-492, 1998.

[80] K. Sato, S. Shah, L.J. Aggarwal - "*Partial Face Recognition using Radial Basis Function Network*", Proceedings of the third IEEE International Conference on Automatic Face and Gesture Recognition, Aprilie 14-16, pag. 288-292, Nara, Japan, 1998.

[81] E. Cabello, A. Sanchez, L. Pastor – "*Some Experiments on Face Recognition with Neural Networks*", în H. Wechsler şi colectiv – "Face Recognition. From theory to Applications", pag. 589-598, NATO ASI series, Springer-Verlag, 1998.

[82] M. Zhang, J. Fulcher – "*Face Recognition using Artificial Neural Network Group-Based Adaptive Tolerance (GAT) Trees*", IEEE Trans on Neural Networks, vol.7, nr.3, pag. 555-567, Mai, 1996.

[83] L. Pessoa and A.P Leitao, "*Complex Cell Prototype Representation for Face Recognition*", IEEE Trans. Neural Networks, vol.10, no.6, pag.1528-1531, Nov. 1999.

[84] C.D. Căleanu – "*Rețele neuronale în recunoaşterea imaginilor. Recunoaşterea facială*", Referat doctorat nr.2, UPT, Timişoara, pag.5-15, 1998.

[85] C.D. Căleanu – "*Conventional versus neural networks techniques for facial recognition*", Proc. of the Symposium of Electronics and Telecommunications, Etc2000, vol.II. pag. 61-64, Timişoara, 2000.

[86] L. Hang, A. Join – "*Integrating Face and Fingerprints for Personal Identification*", IEEE PAMI, vol.20, nr.12, pag.1295-1307, Dec., 1998.

[87] H.A. Rowley, S. Baluja, T. Kanade – "*Neural Network – Based Face Detection*", Proc. of the 1996 IEEE Comp. Soc. Conf. Com. Vision and Pattern Rec., pag. 203-208, Iunie 18-20, CA, USA, 1996.

[88] K. K. Sung, T. Poggio – "*Example –based learning for view-based human face detection*", AI Memo1521, CBCL Paper 112, MIT, Dec., 1994.

[89] B. Moghaddam, A. Petland – "*Probabilistic Visual Learning for Object Representation*", IEEE PAMI, vol.19, nr.7, pag.696-709, Iulie, 1997.

[90] H. Moravec – "*Robot Rover Visual Navigation*", Ann Arbor, MI: Univ. of Michigan Research Press, 1981.

[91] N.M. Nasrabadi, C.Y. Choo – "*Hopfield Network for Stereo Vision Correspondance*", în M. Gupta, G. Knopf (Eds.) - "*Neuro-Vision Systems. Principles and Applications*", IEEE Press, pag. 442-449, 1994.

[92] L.C. Wang, S.Z. Der, N.M. Nasrabadi – "*Automatic Target Recognition Using a Feature-Decomposition and Data-Decomposition Modular Neural Network*", IEEE Trans. on Image Processing, vol.7, nr.8, pag.1113-1121, Aug., 1998.

[93] C.D. Căleanu – "*Performances Improvement of Neural Networks Multilayer Perceptron Type in Facial Recognition*", 5th International Conference on Engineering of Modern Electric Systems, Oradea, pag.116-122, Mai, 1999.

[94] C.D. Căleanu – "*Procesarea neuronală paralelă a regiunilor de interes în recunoaşterea facială*", Bul. St. UPT, Tom 44(58), Fasc. 1, pag.129-136, 1999.

[95] C.D. Căleanu – "*Facial Recognition using Committee of Neural Networks*", Proc. 5th Seminar on Neural Network Applications in electrical engineering, NEUREL 2000, Belgrad, Yugoslavia., pag. 97-100., 2000.

[96] *** - "*Using MATLAB*", ver.5, The MATHWORKS Inc., 1999.

[97] * * * - "*MATLAB Function Reference*", vol.1:Language, ver.5, The MATHWORKS Inc., 1999.

[98] M. Ghinea, V. Fireţeanu, - "*MATLAB. Calcul numeric-Grafică-Aplicaţii*", Ed. Teora, 1995.

[99] www.mathworks.com.- The MATHWORKS Inc.

[100] H. Demuth, M. Beale – "*Neural Network Toolbox. User's Guide*", ver.3.0, The MATHWORKS Inc., 1998.

[101] *** - "*Building GUIs with MATLAB*",ver.5, The MATHWORKS Inc., 1999.

[102] S.Tiponuţ, C.D. Căleanu, V. Tiponuţ – "DSP-MATLAB based tool for Artificial Neural Network Applications Development", Analele Fac. de Inginerie, Hunedoara, an 2, fasc.2, ISSN 1454 – 6531, pag. 193 – 196, 2000.

[103] V. Tiponuţ, B. Şerbănescu, C.D. Căleanu – "*Echipament pentru conducerea*

automată pe rost a capului de sudare", **Noutăți în sudarea structurilor metalice, Buzău, pag. 182 – 188, 1999.**

[104] *** - User's Guide TMS320C3X Texas Instruments 1997 SPRU031E

[105] F.S. Samaria, A.C. Hartner – "*Parametrisation of a stochastic model for human face identification*", Proc. of 2nd IEEE Workshop on Applications of Computer Vision, Sarasota, 1994.

[106] C.D. Căleanu – "*Hybrid Neural Networks: An Overview*", Sesiunea de comunicări ştiinţifice, 19-20 Oct., Hunedoara, Tom II, Fasc.2, ISSN 145-6531, pag.189-192, 2000.