1. INTELIGENȚA ARTIFICIALĂ ŞI SISTEME EXPERT ÎN MANAGEMENT

În contextul noului mediu economic, utilizarea calculatorului, a informaticii economice, nu determină doar dezvoltarea companiilor, dar în multe situații asigură chiar supraviețuirea lor.

Utilizarea tehnicii de calcul pentru rezolvarea problemelor complexe de evaluare, ajustare, diagnosticare, prognozare a unui sistem economic, are o istorie care poate fi considerată deja îndelungată. Generația actuală de calculatoare reuşeşte să implementeze metode şi algoritmi care să fie capabili să înglobeze experiența acumulată referitoare la o gamă largă de situații, şi chiar să învețe din experiențe noi. Aceste metode sunt grupate generic sub denumirea de tehnici de **inteligență artificială (IA)**, şi vin în sprijinul cursei frenetice a forțelor de piață ce obligă firmele de producție să lanseze noile generații de produse la perioade de timp din ce în ce mai scurte.

1.1. Sisteme Expert, definiții

Cuvântul ***inteligență*** provine din limba latină (intelligentia - pricepere, înțelegere, cunoaştere) fiind regăsit în majoritatea limbilor moderne, cu aceeaşi semantică. Lista definițiilor care urmează încearcă să creeze o înțelegere cât mai complexă a implicării acestei noțiuni în terminologia specifică a Sistemelor Expert.

- **Inteligența** – [Mar-78] reprezintă înțelegerea profundă, uşoară a unor lucruri, mai ales în domeniul culturii şi al ştiinței; facultatea de a înțelege, de a pricepe fenomenele, lucrurile, etc.
- **Întelegerea** – [Ros-75] este activitatea gândirii prin care se descoperă legăturile dintre obiecte şi fenomene. În formă elementară, înțelegerea este cuprinsă chiar în procesul perceperii. În formă mai complexă, înțelegerea este implicată în descoperirea legăturilor dintre cauză şi efect, a semnificației unei lucrări artistice sau ştiințifice, a motivelor conduitei oamenilor, etc. Înțelegerea este implicată, mai ales, în procesul de rezolvare al problemelor şi se bazează în general pe experiența trecută şi pe utilizarea acesteia, într-o situație nouă.
- **Inteligența Artificială** – **IA** [Pop-81] reprezintă un domeniu de cercetare al cărui scop constă în studiul şi modelarea inteligenței, prin crearea de sisteme capabile să îndeplinească activități inteligente. Sistemele realizate nu trebuie să copieze în mod necesar metodele şi tehnicile utilizate de om pentru îndeplinirea activităților, importantă fiind numai efectuarea ieftină, sigură şi eficientă a activităților propuse.
  Inteligența Artificială – **IA** [Sfe-93] cuprinde eforturile depuse pentru dotarea calculatoarelor cu capacități, care în mod obişnuit, sunt atributele inteligenței umane: achiziția de cunoştințe, percepția (vizuală, auditivă), raționamentul, luarea deciziei, etc.
  Inteligența Artificială - [Bee-93] reprezintă acea arie a ştiinței calculatoarelor preocupată cu realizarea maşinilor care execută anumite acțiuni, care dacă ar fi realizate de oameni ar fi considerate că exprimă comportamentul uman.
- **Sistemul Expert –SE** [Bee-93] reprezintă o formă a inteligenței artificiale. SE este proiectat pentru a reproduce tehnica de rezolvare a problemei unui expert într-o arie îngustă de specializare, în care se justifică mult mai mult raționamentul decât calculul.
  Un **Sistem Expert – SE** [Bra-88] este un program care rezolvă problemele dintr-un domeniu îngust de aplicație, asemeni unui expert uman.
  Un **Sistem Expert** [Lug-91] este un program bazat pe cunoaştere care oferă soluții de calitate expert, într-un domeniu specific.

Un **Sistem Expert** [Fei-82] este un program inteligent care utilizează cunoştinţe şi proceduri de inferenţă pentru rezolvarea unor probleme care sunt suficient de dificile pentru a solicita expertize efectuate de specialişti.

- **Raţionamentul -** [Bee-93] este abilitatea de a concepe, sau încercarea de a ajunge la o concluzie pornind de la premize valide sau invalide.

Sistemele Expert reprezintă o ramură a inteligenţei artificiale care folosec cunoştinţe specializate pentru a rezolva o problemă la nivelul unui expert uman. Toate definiţiile Sistemelor Expert scot în evidenţă o trăsătura specifică lor şi anume faptul că un SE utilizează informaţii extrase din experienţa umană, putând deci furniza decizii la nivelul de competenţă corespunzător informaţiilor primite şi a metodelor de raţionare implementate. Un sistem expert nu este numit program, ci sistem, deoarece încorporeaza multe tehnologii diferite, cum ar fi baza de cunoştinţe, mecanisme de interferenţă, facilităţi de explicare etc. [www-07].

Realizarea unei maşini inteligente, care să imite performanţele complexe ale comportamentului uman reprezintă o reală provocare datorită lipsei de abilitate a omului de a înţelege în întregime mecanismul de procesare, respectiv puterea creierului uman. [Bee-93]. Avantajele şi dezavantajele unei maşini inteligente pot fi cel mai bine reliefate prin realizarea unei analize comparative între comportamentul uman şi sistemele inteligente aşa cum se poate observa în Tabelul 1.1 şi Tabelul 1.2, care sintetizează câteva diferenţe semnificative.

Tabelul 1.1 Avantajele maşinii inteligente

Caracteristica	Abilitatea umană	Abilitatea maşinii
Cunoaşterea	Perisabilă	Permanentă
Raţionament	Inconsecventă	Consecventă
Expertiza	Valoare mare în unităţi monetare	Valoare medie în unităţi monetare
Oameni	Mobilă	Imobilă
Abilitatea de procesare	Inconsecventă	Foarte consecventă
Rezistenţa	Limitată	Nelimitată
Expertiza	Restrânsă	Vastă
Gândirea	Mortală	Fără moarte

Tabelul 1.2 Limitările maşinii inteligente

Caracteristica	Abilitatea umană	Abilitatea maşinii
Cunoaşterea	Evolutivă	Statică
Raţionament	Nelimitată	Limitată
Expertiza	Adaptabilă	Rigidă
Oameni	Receptivă	Nereceptivă
Abilitatea de procesare	Multiplă	Singulară
Gândirea	Conştientă	Inconştientă
Expertiza	Creativă	Fără inspiraţie

În general sistemele inteligente sunt sisteme deductive, putând genera concluzii pe baza cunoaşterii incorporate sau furnizate din exterior şi respectiv neputând genera singure noi cunostinte.

Conceperea şi dezvoltarea unui sistem inteligent în management necesită cunoştinţe şi experienţă complexă în domenii multiple cum ar fi: proiectare, producţie, programare, utilizarea calculatoarelor şi experienţa economică a societăţilor comerciale. Capacitatea de a realiza o distincţie clară a noţiunilor de activitate (sarcină), cunoaşterea problemelor din domeniu, de metodă de rezolvare a problemelor, precum şi integrarea unui model conceptual bine structurat, constituie o bază necesară pentru dezvoltarea unui sistem inteligent în management.

1.1.1. Structura unui Sistem Expert

Structura unui Sistem Expert este reprezentată în fig. 1.1, modulele componente având următoarele semnificații:

- **Baza de cunoştinţe** înglobează o colecție de cunoştințe relevante despre un anumit domeniu, fiind formată din:
  - **baza generală de cunoştinţe** – modul ce conține regulile referitoare la operațiile care se pot efectua asupra elementelor de cunoaştere conținute în baza de date specifică cazului. În esență, regulile constituie un ansamblu complet şi necontradictoriu de cunoştințe necesare rezolvării unei probleme.
  - **datele cazului specific** – modul ce conține informațiile relative la domeniul de aplicație studiat. Faptele reprezintă partea dinamică a bazei de cunoştințe şi au rolul de a reprezenta starea obiectelor la un moment dat.

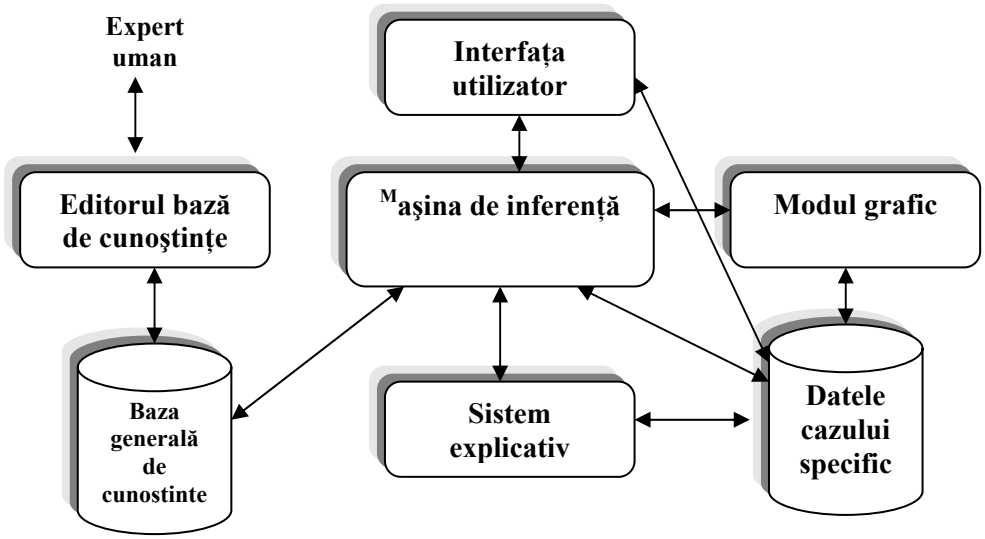


Fig. 1.1. Structura unui Sistem Expert

- **Maşina de inferență -** este un program general care implementează mecanismul prin care se construiesc deducțiile, prelucrează cunoştințele şi datele pe baza acestor deducții. Motorul de inferență conține cunoaşterea procedurală şi de control.

- **Interfața utilizator -** realizează o legătură facilă între sistem şi utilizator, intermediind un dialog eficient între aceştia. Interfața permite utilizatorului să pună întrebări sistemului expert, să introducă noi informații, sau să obțină o imagine asupra procesului de rezolvare.

- **Editorul bază de cunoştințe -** permite completarea sau modificarea bazei de cunoştințe a SE. Acțiunea propriu-zisă de achiziționare se realizează prin chestionarea experților umani.

- **Sistemul explicativ -** are rolul de a furniza explicații operatorului în legătură cu raționamentul folosit pentru a ajunge la concluzia prezentată.

- **Modulul grafic –** reprezintă interfața grafică care facilitează experților din domeniu, posibilitatea de a-şi formaliza cunoştințele într-o manieră cât mai reprezentativă, cu o asistență cât mai limitată a inginerului de cunoştințe.

Sistemele inteligente, respectiv SE au fost concepute pentru a rezolva o serie de probleme din domenii largi, cum ar fi: afaceri, inginerie, chimie, geologie, ştiința calculatoarelor, medicină, matematică, drept, apărare, educație, ş.a.

Câteva din domeniile de aplicație ale IA sunt prezentate în continuare:

1. **Vedere artificială** – recunoaşterea formelor
2. **Prelucrarea vocii** – constituirea şi sinteza vocii umane
3. **Prelucrarea limbajului natural** – înțelegerea, prelucrarea şi redarea limbajului natural
4. **Recunoaşterea formelor** – perceperea, şi clasificarea diferitelor forme
5. **Rezolvarea problemelor** – formalizarea şi rezolvarea unor clase generale de probleme
6. **Procesarea informației** – se ocupă de programe capabile să înțeleagă informația scrisă sau citită, să realizeze rezumate, să răspundă la diferite întrebări;
7. **Jocuri pe calculator**;

Categoriile de probleme solvabile de către un Sistem Expert pot fi sintetizate în modul următor [Lug-91].

1. **Interpretare** – deducerea unor concluzii intermediare sau finale pentru anumite piese de cunoaştere provenite din baze de date primare.
2. **Diagnoză** – stabilirea tipului de defect a unui sistem cât şi a cauzelor generatoare conform unui model prestabilit.
3. **Predicția** – prezicerea cu un anumit grad de probabilitate a consecințelor evoluției unui sistem dat.
4. **Proiectare** – determinarea configurației unui sistem, atât în concordanță cu specificațiile funcționale prestabilite cât şi cu specificațiile exogene sistemului.
5. **Planificare** – programarea unei suite de activități în funcție de dependențele tehnologice dintre ele, precum şi de condițiile prestabilite de start şi/sau de final.
6. **Supraveghere** – verificarea evoluției unui sistem dat în comparație cu evoluția prescrisă.
7. **Depanare** – localizarea defectelor şi implementarea remediilor în cazul funcționării defectuase a unui sistem dat.
8. **Instruire** – ghidarea utilizatorului în procesul de învățare, precum şi localizarea si corectarea deficiențelor de înțelegere .
9. **Sisteme bazate pe cunoştințe -** pot fi utilizate ca şi sistem de rezolvare a problemelor generale, însă nu pot fi cu adevărat numite sisteme expert datorită lipsei expertizei specifice.

1.1.2. Conceptul de cunoaştere şi reflectarea sa în Sistemele Expert

[Păn-00], [Str-99], [www-10], [www-18]

Cunoaşterea presupune o reflectare activă în conştiință a lumii reale, a esențialului şi generalului din fenomene şi a legăturilor obiective ale realității. Astăzi, omul în tentativa lui de cunoaştere a lumii înconjurătoare dispune de un instrument suplimentar, calculatorul, care mai mult decât hârtia, îl poate ajuta la stocarea şi procesarea informațiilor despre obiecte, fapte, fenomene.

Cunoaşterea este un tip special de activități, reflectând interacțiunea între individul uman, ca subiect cunoscător şi realitatea fizică sau socială existentă, independentă de el [Geo-82].

Cunoaşterea este considerată cel mai adesea o măsură a inteligenței.

Utilizarea calculatorului în procesarea informațiilor poate conduce la atingerea unor detalii cu o viteză semnificativă, urmărindu-se un anumit scop prin aplicarea unei metode specifice. Tratarea sistemică a informațiilor le ridică din starea de elemente de mulțime (relația de apartenență), în postura de elemente de structură şi le conferă acestora proprietăți structurale (relații compoziționale, cauzale, modale, epistemice), precum şi o denumire nouă, aceea de *date*.

Cunoaşterea poate fi clasificată în mai multe moduri cum ar fi: **cunoaşterea comună**, fără metodă şi **cunoaşterea ştiințifică**, cu metodă; **cunoaşterea empirică**, dacă informațiile despre obiecte, fenomene, procese provin de la organele senzoriale sau de la instrumentele de măsură şi **cunoaşterea teoretică**, dacă ea provine dintr-un raționament care operează cu legi cauzale, etc. Cunoaşterea teoretică se dezvoltă din cunoaşterea empirică prin analiză, sinteză, deducție, particularizare, etc.

Finalitatea actului de cunoaştere ştiințifică este reprezentată de **teoria ştiințifică.** Cuvântul *teorie* provine din grecescul "*theoria*" cu înțelesul de contemplare, meditație. Acest termen are astăzi, în filozofia ştiinței, conținut divers. Teoria poate fi definită ca şi:

- o mulțime de *reguli şi principii de procedură;*
- o *schemă de terminologie şi clasificare;*
- un *sistem de concepte;*
- un *mod de descriere;*
- un *sistem de propoziții,* formulate asupra unor entități neobservabile;
- un *sistem ipotetico-deductiv.*

Teoria desemnează un sistem de propoziții, logic organizat, care sintetizează o anumită cantitate de informații, referitoare la un domeniu al realității, pe care îl descrie şi îl expică[Geo-82].

Cunoaşterea este întotdeauna parțială şi incompletă. Esența cunoaşterii aparține "obiectului", prin urmare ne este inaccesibilă [Hei-88]. Transpunerea activității de cunoaştere, pe suportul numit "calculator", este o problemă a "inteligenței artificiale".

Omul nu creează o altă inteligență, diferită de a sa, ci îşi foloseşte propria inteligență pentru a face calculatorul să aibe un astfel de "comportament" încât, să poată fi perceput ca inteligent. De aici rezultă şi o definiție demitizatoare a **Inteligenței Artificiale**, potrivit căreia:

Inteligența Artificială [Cât-95] reprezintă abilitatea omului de a instrui o maşină, astfel încât, în anumite împrejurări particulare, maşina să se comporte, prin reacții la stimulii externi, ca o entitate inteligentă.

Maşina inteligentă trebuie să-şi reprezinte raționamentele prin intermediul anumitor convenții şi simboluri, cu care să fie capabil să opereze. Metoda de reprezentare presupune o anumită ordine conceptuală, care cuprinde:

- sistemul de meta-reprezentare;
- sistemul de clasificare ;
- sistemul de organizare.

Problema fundamentală a Inteligenței Artificiale şi respectiv a sistemelor expert este cea de definire a unor metode pentru reprezentarea unor cantități semnificative de cunoştințe, metode care să permită stocarea şi utilizarea eficientă a acesteia. Metodele de reprezentare ale cunoaşterii pot fi grupate în:

- metode logice;

- metode procedurale.
- metode relaționale (modele structurate);

Metodele logice *descriu cunoaşterea ca pe o înlănțuire de acțiuni care se implică reciproc atât din punct de vedere al cunoştințelor cât şi cel al relațiilor dintre acestea.*

Avantajul metodelor logice constă în aplicarea directă a regulilor de inferență, asupra elementelor de cunoaştere.

Dezavantajul metodelor logice constă în existența soluțiilor nesatisfăcătoare de sistematizare a bazei de cunoştințe, în dificultatea reprezentării cunoaşterii despre acțiuni şi a regulilor euristice.

Calcul predicatului de ordinul întâi ca şi exemplu de metodă logică, reprezintă o bază de cunoştințe construită în limbajul logic, incluzând piesele de cunoaştere reprezentate cu ajutorul unor expresii (propoziții simple) înlănțuite între ele cu formule ale acestui limbaj. Astfel, un element de cunoaştere reprezentat inițial în limbaj natural este descompus în propoziții elementare adevărate denumite "**aserțiuni**" (care specifică fapte, proprietăți, relații) legate de elementul de cunoaştere. Acesta este motivul pentru care metoda se mai numeşte şi "*Reprezentarea faptelor*" .Fiecare fapt poate fi tratat din punct de vedere informatic ca un lanț de caractere specific, ca de exemplu: RANDAMENT_proiect_02 = 90%

Pentru a procesa mai uşor informațiile, se structurează lanțurile de caractere. Un procedeu curent îl reprezintă utilizarea tripletelor:

<PREDICAT>(<SUBIECT>,<OBIECTE ASUPRA CARORA SE EFECTUEAZĂ PREDICAȚIA>)

Fiecare propoziție elementară este generată de un predicat cu un număr finit de locuri în care sunt specificate variabile formale sau obiecte din mulțimea suport.

Pentru crearea elementului de cunoaştere, propozițiile elementare sunt înlănțuite prin intermediul conectorilor logici ($\forall$, $\exists$, <, >, =, $\rightarrow$, $\neq$, $\Leftrightarrow$, $\Rightarrow$, ^, V) putându-se efectua raționamentele în mod logic.

Particularități: metodele logice de reprezentare a cunoaşterii sunt ***declarative*** deoarece oferă facilități pentru specificarea aspectelor **statice** ale cunoaşterii, prin care sunt descrise componente, proprietăți şi fapte despre obiecte, evenimente şi stări, precum şi relațiile dintre acestea în cadrul universului discursului.

Metodele procedurale *sunt acelea în care cunoaşterea este reprezentată sub formă de proceduri, care permit obținerea stărilor momentane, pornind de la stările inițiale sau intermediare.*

Ideea reprezentării procedurale a cunoştințelor a apărut inițial ca o încercare de a scoate în evidență controlul implicit al secvenței stărilor din cadrul metodelor logice de reprezentare. Într-o reprezentare procedurală, cunoştințele despre lume sunt incluse în proceduri [Tac-98].

Reprezentarea procedurală scoate în evidență relațiile dintre elementele de cunoaştere, descriind practic paşii de prelucrare printr-un algoritm, o procedură de calcul, o strategie sau prin descrierea unui proces.

Elementele de cunoaştere declarative şi procedurale sunt puternic corelate, prezentând multe puncte comune, astfel:

- Cunoştințele pot fi reprezentate atât declarativ, cât şi procedural. Spre exemplu relația $y=x^2$ poate fi reprezentată atât declarativ, sub formă de tabel, cât şi procedural prin precizarea paşilor de calcul pentru y, cunoscându-l pe x.

- Un element de cunoaştere poate fi interpretat atât declarativ cât şi procedural. Implicația p→q poate fi interpretată declarativ, putând fi considerată adevărată sau falsă, sau procedural după cum urmează: pentru a deduce pe q este necesar a demonstra pe p.
  Diferențele dintre cele două reprezentări depind de domeniul de utilizare. Astfel, interpretările procedurale sunt legate uzual de un domeniu concret de utilizare, în timp ce reprezentarea declarativă permite mai multe variante de utilizare. În acest ultim caz , cunoştințele pot fi mai uşor modificate, dar în acelaşi timp ocupă mai multă memorie.

Sisteme de reguli de producție ca şi exemplu de metodă procedurală de reprezentare a cunoaşterii, sunt cele care stau la baza funcționării sistemelor expert. Regulile de producție îşi au originea în sistemele care se bazează pe un set de transformări succesive ale unui context inițial, aşa cum sunt gramaticile formale sau automatele finite.[Păn-00]. În acest sens, partea de condiție a unei reguli se mai numeşte şi parte *contextuală*, ea indicând starea în care regula este aplicabilă, iar cea de acțiune se numeşte şi parte *transformațională*, prin aceea că determină modificarea contextului.

In anumite domenii regulile sunt dificil de formalizat, iar numarul acestora poate deveni atât de mare încât să nu mai fie posibilă manipularea lor datorită restricțiilor apărute pentru sistemul de calcul.

Metodele relaționale (modele structurate) *sunt acelea prin care cunoaşterea este reprezentată, pornind de la relațiile dintre obiecte, sub formă de grafuri şi rețele.*

Metodele din această categorie utilizează modele de tip structurat şi permit organizarea cunoştințelor funcție de omogenitatea acestora, fiind concepute pentru: clasificare, prelucrare a limbajului natural, planificarea activităților.

Avantaje: Se evidențiază în mod explicit relațiile dintre entități, programul având astfel acces rapid la obiectele "înrudite" crescând eficiența computațională. Se pot astfel genera tipuri de inferențe specializate bazate pe mecanisme eficiente.

Dezavantaje: Se pierde din generalitatea pe care o are un model de tip logic.

Reprezentarea cunoaşterii prin rețele semantice (metodă relațională) dezvoltată inițial de Quillian în 1968 [Qui-85], în vederea reprezentării conceptelor specifice cu caracter de asociativitate ale memoriei umane, este utilizată prin reprezentarea grafică a cunoştințelor declarative exprimate sub formă de propoziții. Astfel, obiectele, evenimentele sau diferitele situații cu o anumită structură sunt reprezentate prin intermediul unui graf. *Nodurile* grafului simbolizează obiecte, proprietăți ale obiectelor sau valori ale proprietăților, concepte sau situații. *Arcele* (uzual orientate) reprezintă relațiile dintre noduri. Semnificația nodurilor şi a arcelor este precizată prin etichete. Această metodă de reprezentare exprimă cunoaşterea sub formă de *declarații binare*, respectând următorul formalism: R(a,b)
în care R exprimă relația, iar a şi b entitățile conectate prin relația R , fig. 1.2.

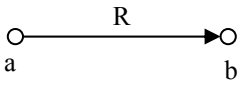


Fig. 1.2. Graful corespunzător expresiei R(a,b)

Rețeaua semantică reprezintă prin urmare o *mulțime de declarații binare*, oferind în acelaşi timp o *direcționare a căutării informației* prin structura inclusă în formalismul declarațiilor binare.

Reprezentarea cunoaşterii bazată pe cadre (metodă relațională)

Cadrele (Frame) introduse inițial de către Minsky [Gia–89] au la bază ideea conform căreia inteligența presupune frecvent folosirea unor *scheme* de cunoaştere, bazate pe o experiență anterioară în domeniul problemei care se rezolvă, *scheme* cu ajutorul cărora se interpretează noile situații apărute; adesea, comportarea inteligentă presupune nu o rezolvare de la "zero", ci o adaptare a unor tipare (prototipuri) deja acumulate [Păn-00]. Aceasta înseamnă că acest model este adecvat mai ales pentru a reprezenta cunoştințe cu un caracter stereotip, caracteristici obişnuite ale obiectelor şi relații tipice între obiecte. Pentru realizarea acestui model structurat se poate defini termenul de *schemă* în modul următor:

O schemă se bazează pe o structură care conține cunoştințe tipice asupra unor concepte sau obiecte, incluzând atât cunoştințe declarative cît şi cunoştințe procedurale.

Cadrele reprezintă un tip de schemă care combină o serie de concepte referitoare la reprezentarea cunoştințelor, oferind structuri adecvate pentru prezentarea situațiilor stereotipe sau obiecte tipice. În particular, cadrele reprezintă structuri potrivite pentru reprezentarea cunoştințelor generale, universale, de obicei greu reprezentabile în sistemele de calcul.

Un cadru este un obiect structurat, care conține informații despre un obiect (acțiune, regulă) din universul reprezentat, într-o formă stereotipă.

Concret, un cadru este o structură care se identifică printr-o etichetă – *numele cadrului*, conținând perechi de forma: *caracteristică (proprietate, atribut) –valoare ("slot-value").*

Cadrele reprezintă un *formalism orientat pe obiect,* fiind des utilizate pentru reprezentarea cunoştințelor care operează cu concepte distincte, care pot fi descrise într-o manieră stereotipă şi care interacționează între ele prin mecanisme precise [Deb-94].

Avantaje:

- Organizarea structurată a modelelor de rețea semantică şi cadre, determină focalizarea atenției în efectuarea raționamentelor – toate cunoştințele despre o entitate sunt grupate, putând fi apelate dintr-un singur loc. Datorită structurării proprii, aceste modele uşurează (în comparație cu modelul calculului cu predicate), eliminarea detaliilor nesemnificative din cadru proceselor de inferență.

- Din punct de vedere al achiziției de cunoştințe, reprezentarea cunoaşterii prin modele structurate se realizează mai uşor prin intermediul unei rețele semantice sau un cadru decât prin intremediul calculului cu predicate. Cadrele şi scenariile permit constituirea unor modele coerente şi unitare , chiar şi atunci când achiziția de cunoştințe este discontinuă sau incompletă. Uşurința în achiziționare provine şi din faptul că rețelele semantice şi cadrele nu impun stabilirea tuturor relațiilor din cadrul unui concept, oferind ulterior posibilitatea formulării sau achiziționării cunoştințelor care lipsesc (ex. se dispune doar de o parte din subconceptele unui concept).

- Modelele structurate, şi dintre ele în primul rând cadrele şi scenariile, sunt adecvate pentru a reprezenta cunoştințe cauzale, prin aceea că succesiunea scenelor sau legăturilor care se fixează între cadrele şi scenariile ce compun baza de cunoştințe pot fi realizate conform relațiilor cauză – efect. Tot un avantaj este şi acela că aceste modele pot ghida procesul de inferență şi deci soluționarea unei probleme pe baza experienței în domeniul respectiv; din acest punct de vedere aceste variante se apropie de modelul de reprezentare a cunoaşterii bazat pe reguli.

1.1.3. Condițiile necesare dezvoltării unui sistem expert

Pentru crearea unui sistem expert performant trebuie să se stabilească un plan de dezvoltare care să includă [Lug-91]:

1. selectarea unui formalism pentru reprezentarea cunoaşterii
2. proiectarea unei maşini de inferență
3. adăugarea facilităților de interacțiune cu utilizatorul
4. adăugarea unei facilități de manevrare a incertitudinii

Odată ce s-a stabilit modul de reprezentare a cunoaşterii, pentru a se putea determina codificarea cunoştințelor domeniului specific în program, dezvoltarea sistemului expert parcurge două etape esențiale:

a. *acumularea cunoştințelor* necesare în rezolvarea problemelor de către sistemul expert
b. *dezvoltarea programelor* care să proceseze cunoştințele acumulate

a. etapa de acumulare a cunoştințelor este esențială, tinând cont de faptul că oricât de performante ar fi tehnicile de programare, sistemul nu va fi capabil să depăşească limitele cunoaşterii achiziționate [Păn-00]. Pentru ca simultan cu acumularea să se imbunătățească şi performanțele de cunoaştere, este necesară o clasificare a noilor cunoştințe, oferindu-se astfel posibilitatea de a fi regăsite atunci când sunt necesare. De asemenea, noile cunoştințe în multe situații sunt în interacțiune cu cele vechi, creându-se relații care trebuie puse în evidență. Cunoaşterea este acumulată prin procese de observare a realității, învățare şi abstractizare.

   Cunoaşterea este alcătuită dintr-o mulțime de "*piese de cunoaştere*", adică elemente care au o anumită semnificație, pe baza cărora SE să poată derula raționamente necesare în rezolvarea problemelor. În mod uzual se disting două tipuri de piese de cunoaştere:

   - *conceptele* – cele care condensează rezultatele procesului de abstractizare, prin care se specifică însuşirile esențiale, necesare şi suficiente pentru a decide apartenența obiectelor la anumite clase
   - *instanțele* – obiecte individuale

b. etapa de dezvoltare a programelor este necesară pentru a codifica şi procesa cunoştințele acumulate, în scopul rezolvării în mod autonom a problemelor propuse.

   Programele dezvoltate în mod specific pentru rezolvarea anumitor probleme trebuie să faciliteze atât găsirea faptelor relevante din baza de cunoştințe cât şi efectuarea raționamentelor pe baza cunoştințelor extrase pentru identificarea alternativelor şi soluțiilor corecte.

Proiectarea, realizarea şi utilizarea unui Sistem Expert este posibilă în urma îndeplinirii unor condiții:

- strategice;
- informatice;
- de specialitate;
- de procedură.

Condițiile strategice presupun **:**

- definirea domeniului;
- definirea scopului;
- definirea funcţiilor sistemului;
- existența perspectivei progresului;
- existența resurselor materiale.

Condiţiile informatice presupun**:**

- deținerea şi cunoaşterea softului specific;
- cunoaşterea metodelor de reprezentare a cunoasterii;
- existența metodelor de stabilire a coerenței bazei de cunoştinte.

Cunoştinţele de specialitate presupun**:**

- cunoaşterea domeniului pentru care se dezvoltă sistemul;
- existența posibilitații definirii claselor;
- posibilitatea extragerii parametrilor necesari de următoarele tipuri:
    - de performanța tehnică;
    - de performanță economică şi comercială;

Condiţiile de procedură presupun**:**

- stabilirea fazelor din procesul de proiectare, care sunt adecvate abordării în această manieră;
- stabilirea fazelor ce urmează să fie rezolvate, cu ajutorul Sistemului Expert;
- definirea restricţiilor de proiectare şi extragerea cunoştinţelor.

Surse pentru extragerea cunoştinţelor:

- din experiența de proiectare;
- din experiența de producție;
- din experiența utilizării şi programării calculatoarelor;
- din cunoştinţe matematice în domeniu;
- din experiența economică a societăților comerciale.

1.2. Rețele neuronale artificiale (RNA)

1.2.1. Considerații asupra RNA

În ultimul deceniu a crescut exponențial interesul pentru domeniul rețelelor neuronale artificiale deşi primele referiri teoretice la astfel de rețele au fost făcute în anul 1943 când Warren McCulloch (neuro-psiholog) şi Walter Pitts au construit un model utilizând rezistoare şi amplificatoare, care simula neuronii naturali, biologici. Neuronii electronici primeau anumite semnale de intrare, pe care, în funcție de câțiva parametri, le trimiteau sau nu mai departe către alți neuroni, care la rândul lor propagau sau nu

semnalele. Modelul construit de McCulloch şi Pitts reprezenta o reţea de celule interconectate, fiecare în legătură funcţională cu următoarele. Interesul actual pentru acest domeniu este justificat în primul rând de posibilităţile ultimelor generaţii de calculatoare, care oferă puterea de calcul necesară cercetării RNA.

RNA au ca punct de inspiraţie sistemul nervos uman. Specialiştii consideră ca la ora actuală sistemul biologic este insuficient explorat şi de aceea modulele utilizate pentru conceperea unei RNA reprezintă introducerea într-un model biologic simplificat.

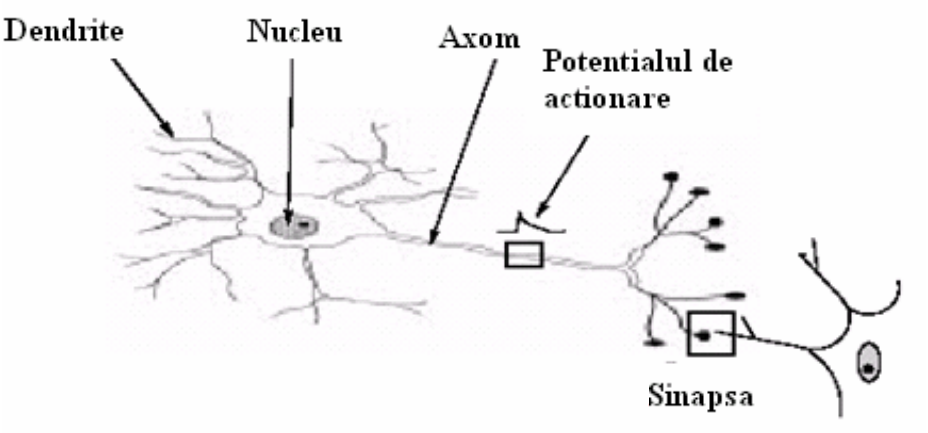


Fig. 1.3. Neuronul biologic

Unitatea de organizare a sistemului nervos este neuronul, (fig. 1.3) o celulă care reprezintă un număr de dendrite şi un axon, prin intermediul căruia se conectează la alţi neuroni. Dendritele constituie intrările în celula neuronală. Funcţia dendritelor este aceea de a recepţiona extitaţii şi de a le conduce până la corpul neuronului. Axonul reprezintă ieşirea. Funcţia axonilor este aceea de a conduce influxul nervos de la corpul celular la dendritele sau corpul celular al altui neuron. RNA este alcătuită dintr-o mulţime de noduri în care se află neuronii artificiali , elemente de procesare neliniară care operează în paralel. Prin analogie cu neuronul biologic, un neuron artificial are mai multe intrări şi o ieşire, care se poate conecta la intrările altor neuroni.

În cadrul sistemului nervos biologic neuronii sunt conectaţi prin spaţiile denumite sinapse. Sinapsele reprezintă unităţi structurale funcţionale, care stabilesc interacţiunile dintre neuroni. Sinapsele impun o excitare sau o inhibare neuronului receptor.

1.2.2. Caracteristicile RNA

Caracteristica esenţială a RNA este aceea că pot descrie o problemă şi să o rezolve prin autoorganizare şi nu prin program. Această autoorganizare are loc pe parcursul unui proces de învăţare obţinut prin cooperarea unei topologii iniţiale , a unor reguli de învăţare şi a unor sesiuni de antrenament.

Capacitatea de învăţare

Caracteristica esenţială a unei RNA este capacitatea de învăţare, respectiv capacitatea de imbunătăţire a performanţelor. Învăţarea se realizează prin intermediul unui proces iterativ de eliminare a erorilor, respectiv de ponderări aplicate conexiunilor şi pragurilor sinaptice.

RNA nu necesită programe puternice, specifice problemei de rezolvat, dar au nevoie de antrenamente asupra unui set de date (învăţare prin exemple)

Fiind dat un set de intrări şi răspunsul dorit , RNA sunt capabile de a se autoorganiza în urma procesului de antrenament, în scopul rezolvării problemei. Există o gamă largă de metode de antrenament, fiecare cu avantajele şi dezavantajele lor, cu aria proprie de aplicare. Indiferent de metoda de antrenament ponderile conexiunilor sunt ajustate pe baza unor modele selectate.

O RNA se consideră instruită daca aplicarea unei multimi de vectori de intrare generează iesirile dorite. Cunoasterea dobândită de RNA este memorata de sinapsele neuronale, mai concret de ponderile conexiunilor dintre neuroni.

Capacitatea de generalizare

În urma unui antrenament corespunzător, RNA sunt capabile să dea răspunsuri corecte şi pentru seturi de intrări diferite de seturile de atrenament atât timp cât diferențele nu sunt foarte mari. Este important de subliniat că aceasta este o caracteristică intrinsecă a RNA, şi nu a unor algoritmi speciali (care înseamnă inteligență umană suplimentară) creați în acest scop.

Capacitatea de sinteză

RNA pot lua decizii sau pot trage concluzii chiar şi atunci când sunt confruntate cu informații parțiale, complexe sau zgomote.

1.2.3. Neuronul artificial

Modelul neuronului artificial construit de McCulloch şi Pitts fig. 1.4 este cunoscut în prezent, sub numele de unitate prag - TU (threshold unit) sau neuron McCulloch-Pitts. Unui neuron de tip TU i se furnizează pe fiecare dintre conexiunile sale de intrare un semnal boolean (0 sau 1) şi emite la ieşire tot un semnal boolean. Conexiunile de intrare pot fi de două tipuri şi anume: inhibatoare şi excitatoare. În oricare dintre modele, neuronul artificial este o aproximare a neuronului biologic, fiind format dintr-un corp, un set de intrări şi o ieşire. Intrările sunt ponderate, fiind ulterior însumate. Suma obținută se aplică unei funcții de activare, care are ca rezultat ieşirea neuronului respectiv.

Semnificațiile notațiilor din fig. 1.4 sunt următoarele:

- inp 1, inp 2, inp n, reprezintă valorile intrărilor;
- W1, W2, Wn sunt ponderile aplicate intrărilor;

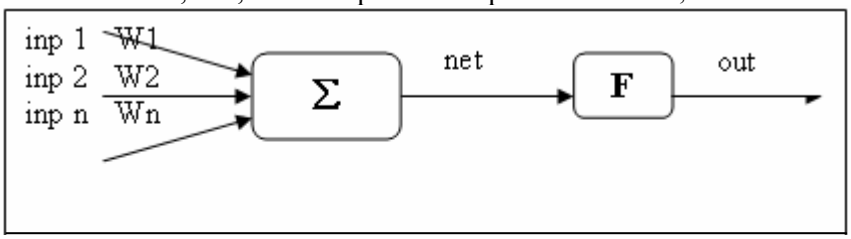


Fig. 1.4. Neuronul Artificial
(modelul McCulloch-Pits)

$$net = \sum_{i=1} inp_i W_i \; ; \quad out = F(net)$$

Funcția de activare poate avea mai multe forme, câteva exemple în acest sens sunt reprezentate în fig. 1.5.

$$F(x) = \begin{cases} 1, & x > t \\ 0, & x \le t \end{cases}$$ sau

$$F(x) = \frac{1}{1 + e^{-x}}$$ sau

$$F(x) = \frac{2}{1 + e^{-x}} - 1$$

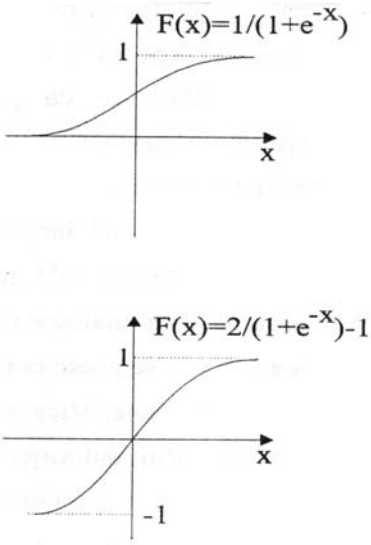


Fig. 1.5. Funcții de activare

1.2.4. Ponderile

Ponderile corespunzătoare fiecărei intrări (W1, W2,….Wn) reprezintă numere reale. Dacă Wj >0 ponderea este sinaptică excitatoare, iar dacă Wj<0 este o pondere inhibitoare. Aceste ponderi sunt stabilite de către rețea în timpul procesului de învățare prin algoritmi specifici, fiind de importanță vitală pentru buna funcționare a rețelei neuronale.

1.2.5. Clasificările RNA

a) Clasificarea după existența buclelor, împarte RNA în:

- ***RNA asociative sau feedforward*** în cazul în care nu există bucle închise (reacție) sau altfel spus nu se permite aplicarea la intrarea unui neuron k a ieşirii unui neuron j, dependent de neuronul k, Fig. 1.6. a

- ***RNA autoasociative sau feedback*** care prezintă bucla de reacție; ieşirea neuronului k este aplicată intrării neuronului j, cu a cărui ieşire se formează reacția. – Fig. 1.6. b

b) clasificarea după numărul de suprafețe neuronale (planuri sau straturi):

- cu un singur strat sau fără "straturi ascunse" Fig. 1.7. a

- cu mai multe straturi (cel puțin un "strat ascuns") Fig. 1.7. b

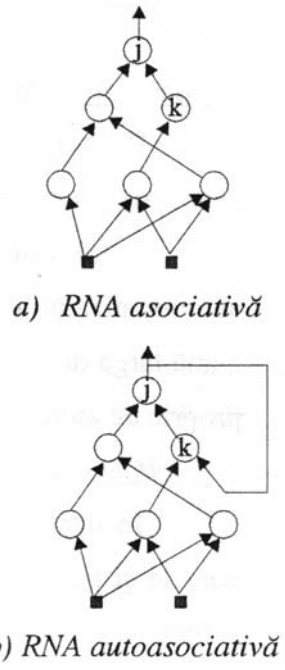
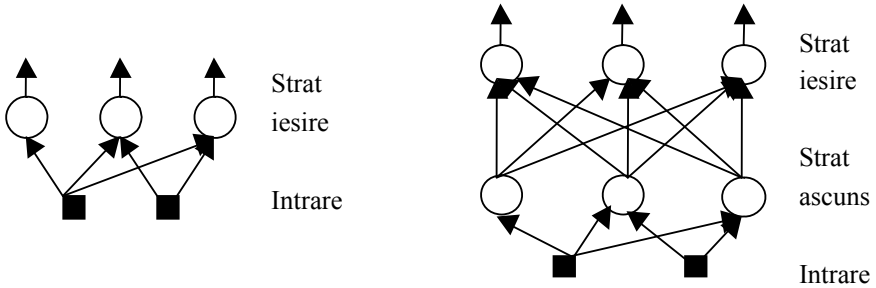


Fig. 1.6. Rețele neuronale cu bucle

c) clasificarea după funcția de transfer a neuronului
d) clasificarea după modul de învățare:
   - supervizat
   - nesupervizat

În procesul de învățare se stimulează continuu parametrii variabili ai RNA. Modul de stimulare al parametrilor este diferit în învățarea supervizată față de învățarea nesupervizată.



a) cu un singur strat (fără straturi ascunse) b) cu un strat ascuns (hidden layer)

Fig. 1.7. Retele nuronale

Învățarea supervizată

În acest caz se furnizează rețelei o mulțime de exemple de instruire, de perechi formate dintr-un vector de intrare, care urmează să se potrivească cu ieşirea dorită. Aplicând rețelei vectorul de intrare, se calculează ieşirea şi se compară cu vectorul de ieşire citit din fişierul de antrenare. Diferența dintre ieşirea dorită şi cea obținută reprezintă eroarea rețelei la momentul respectiv. Perechile vectorilor de intare – ieşire sunt furnizate de către un sistem de antrenare externă, sau de către sistemul care conține rețeaua.

Ponderile rețelei se modifică conform unui algoritm de minimizare a erorii. Vectorii din mulțimea de instruire sunt aplicați secvențial (ciclic), până când eroarea totală asociată întregii mulțimi de instruire atinge o valoare acceptabilă.

Învățarea nesupervizată

În cazul învățării nesupervizate sau autoorganizarea, ajustarea ponderilor nu se bazează pe compararea cu răspunsuri ideale predeterminate, mulțimea de instruire constând din vectorii de intrare. Pe baza unei funcții de autoorganizare, aceste modele se grupează după formele vectorilor de intrare similari.

1.2.6. Analiza comparativă a Sistemelor Expert (SE) şi Rețelele Neuronale Artificiale (RNA), cu posibilități de integrare

[Dum-96], [FIL-99], [Mey-95], [Tac-98]

RNA au avantajul că sunt structuri autonome. Ele pot oferi pentru sistemele complexe o estimare exactă a dinamicii, fără să fie necesare informații despre modelul sistemului. De exemplu, în majoritatea cazurilor, roboții includ foarte multe articulații şi configurații complexe, pentru care este dificil să se conceapă o bază de cunoştințe consistentă integrată într-un SE.

SE şi RNA au origini şi scopuri comune. Ambele abordări au scopul de a efectua raționamente logice, de a simula inteligența artificială. Ambele combină informații cantitative şi calitative. Avantajele şi dezavantajele celor două tehnici sunt în general complementare. Slăbiciunile SE în ceea ce priveşte achiziția de cunoştințe şi reprezntarea lor poate fi compensată de abilitatea cu care RNA pot învăța din exemple tipice. Pe de altă parte, posibilitățile puțin satisfăcătoare oferite de RNA în ceea ce priveşte interfața om – maşină şi capabilitățile de explicare a raționamentului care a condus la o anumită concluzie pot fi teoretic compensate de SE.

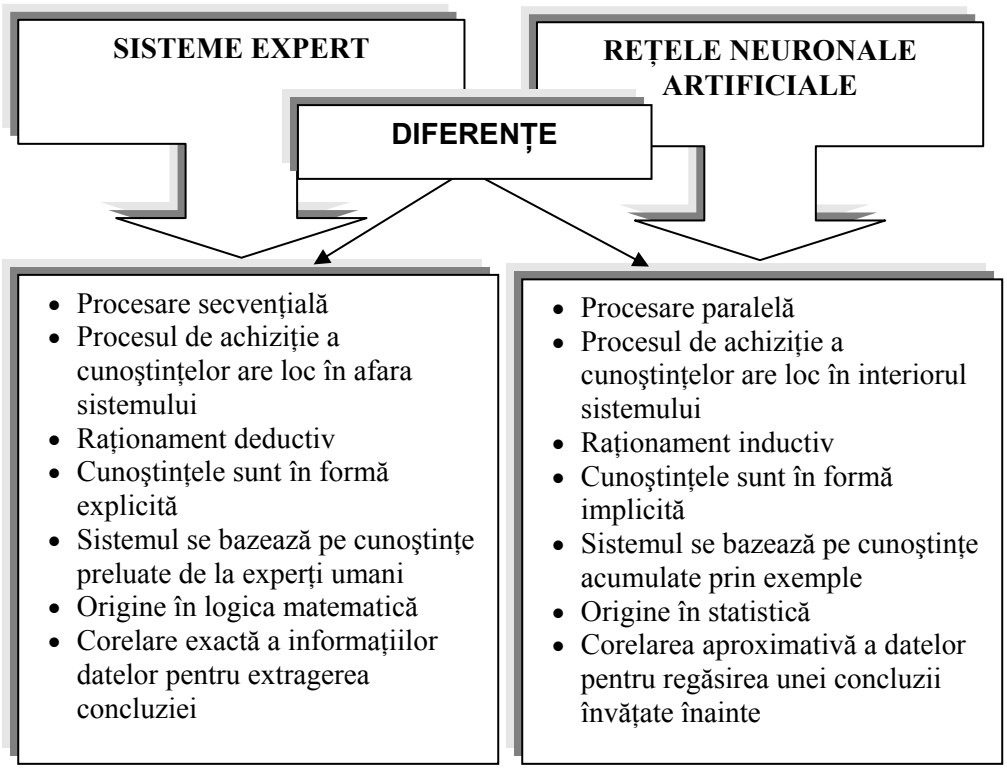


Fig. 1.8. Diferențe între SE şi RNA

RNA fiind concepute conform modelului creierului uman au capacitatea de a învăța, spre deosebire de sistemele inteligente convenționale, care şi-au dovedit superioritatea în raționamente bazate doar pe operații aritmetice şi algoritmi. SE şi RNA, aşa cum se poate observa în fig. 1.8, sintetizează câteva diferențe semnificative.

- O diferență importantă este baza raționamentului: SE sunt bazate pe algoritmi şi deducții pe când RNA încearcă să simuleze mecanismele neuronului biologic.
- Cele două abordări utilizează tehnici de procesare diferite: SE utilizează metode secvențiale de procesare, pe când RNA procesează paralel – fiecare nod (neuron) al RNA efectuează funcțiuni în paralel cu celelalte noduri.

Procesele de învățare şi raționament sunt şi ele diferite: în cazul SE învățarea se efectuează de regulă în exteriorul sistemului – cunoştinţele sunt obținute în afară iar apoi sunt codate în baza de cunoştinţe iar în cazul RNA se acumulează sub forma ponderii legăturilor între noduri. Procesul de învățare este intern şi poate fi dinamic – pot fi implementări care ajustează permanent cunoştinţele pe măsura apariției de noi exemple.

- Metodele de raționament ale SE se bazează pe deducție pentru construirea unei baze interne de cunoştinţe, pe când cele ale RNA se bazează pe inducție.
- Algoritmii de inferență ai SE se bazează pe înlănțuirea logică înainte sau înapoi în baza de cunoştinţe şi necesită o corelare precisă a componentelor acesteia. În mod diferit, RNA utilizează corelarea aproximativă a componentelor bazei de cunoştinţe, pentru a regăsi elemente învățate anterior. SE au deja o utilizare largă în multe domenii(tehnic, comercial, medicină etc.) pe când RNA au mai degrabă o prezență teoretică (e drept din ce în ce mai intensă) în literatura de specialitate.

RNA pot achiziționa cunoştinţe prin învățare directă pe exemple, utilizând algoritmi de învățare specifici, având alte beneficii în ceea ce priveşte achiziția cunoştinţelor, printre care este semnificativă

posibilitatea de a învăța din date de intrare incomplete, parțial incorecte sau parțial contradictorii. De aici derivă şi capabilitatea de generalizare a RNA. SE au avantajul unei interfețe prietenoase şi interactive cu utilizatorul, în special în ceea ce priveşte capabilitatea de a explica raționamentul care a condus la o anumită concluzie. SE utilizează reprezentarea simbolică a cunoştințelor şi oferă posibilități de incorporare a elementelor de raționament euristic. SE pot fi folosite ca şi ghid în selectarea, construirea şi întreținerea RNA.

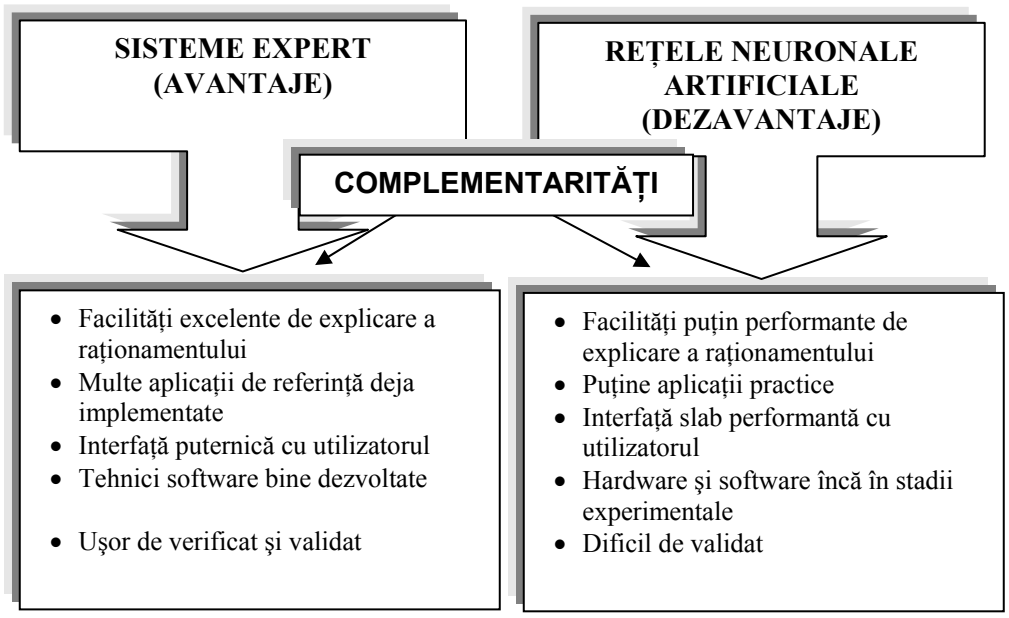


Fig. 1.9. Complementarități între SE (avantaje) şi RNA (dezavantaje)

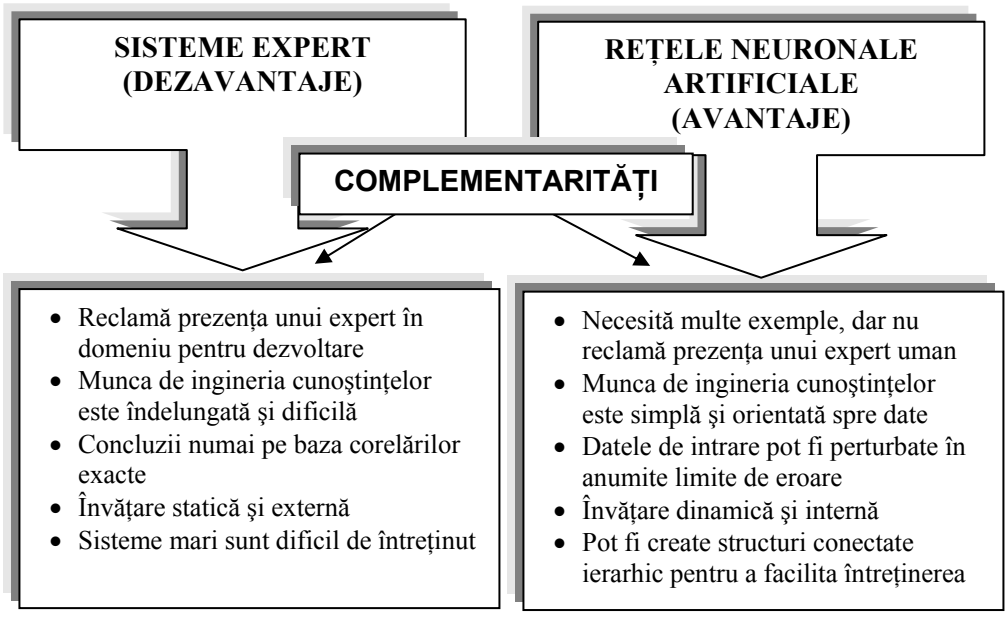


Fig. 1.10. Complementarități între SE (dezavantaje) şi RNA (avantaje)

Odata dezvoltat un sistem expert pe o anumită direcție este aproape imposibil să mai fie schimbat sau adaptat. Dacă provocările sau schimbările care apar ies din cadrul ontologiilor cu care operează,

sistemele expert nu mai au posibilitatea de a reacționa corespunzător. Datorită faptului că rețelele neurale au abilitatea de a învăța, procedurile actuale de integrare a rețelelor neuronale cu sistemele expert asigură o creştere asupra performanţelor sistemelor expert. Astfel prin identificarea complementarităților dintre SE şi RNA fig. 1.9, fig.1.10 atât din punctul de vedere al avantajelor SE, care sunt dezavantaje pentru RNA, cât şi din punctul de vedere al dezavantajelor SE, care sunt avantaje pentru RNA, se poate concepe o activitate de adaptare a sistemelor expert prin integrarea cu rețelele neuronale şi cu o mai mică contribuție de cunoştințe inginereşti.

Literatura de specialitate menționează câteva modele ale integrării SE şi RNA în sisteme hibride. După gradul de interdependență al celor două tehnici se pot identifica următoarele modele:

- SE şi RNA de sine stătătoare – care presupun existența de componente software total independente, care nu interacționează automat în nici un fel. Acest model poate fi folosit pentru asigurarea redundanței între procese, una dintre tehnici validând-o pe cealaltă. Pot exista abordări în care cunoştințele acumulate prin antrenament cu RNA să fie folosite în dezvoltarea ulterioară a unui SE.
- Cuplajul slab între cele două sisteme reprezintă cu adevărat primul nivel de integrare a RNA şi SE. Acest model, constă în aplicații diferite care comunică între ele prin intermediul unor fişiere. Spre exemplu, o RNA poate fi utilizată pentru condiționarea şi validarea datelor de intrare înainte ca acestea să fie tansmise SE. RNA poate elimina zgomotul din informația de intrare, pentru a identifica obiectele sau pentru recunoaşterea specimenului. SE poate utiliza aceste informații pentru a rezolva aplicații de clasificare, identificare etc.

- În cazul cuplajului strâns între SE şi RNA informația este transmisă prin structuri de date rezidente în memoria calculatorului, dar şi prin fişiere de date pe disc. Aria de aplicare a acestui model este aproximativ aceeaşi cu cea a modelului cu cuplaj slab, cu deosebirea că asigură o viteză mult mai mare de procesare.

- Sistemele complet integrate utilizează în comun structuri de date şi cunoştințe. Comunicația între cele două componente se realizează atât cu informații simbolice (caracteristice SE) cât şi cu structuri proprii RNA (coeficienți de pondere)

Cele mai des întâlnite strategii de integrare sunt următoarele:

- Distribuirea componentelor aplicațiilor între abordări SE şi RNA, fiecare subproblemă a aplicației fiind rezolvată în tehnica mai potrivită acesteia. Complementaritățile între SE şi RNA fac deosebit de tentantă această strategie.

- RNA pot fi incluse într-un SE, devenind părți ale acestuia. Spre exemplu RNA poate fi folosită pentru identificarea rapidă a condițiilor în regulile SE, (DACĂ…ATUNCI…) stabilind astfel foarte rapid ce regulă trebuie aplicată pentru un set dat de condiții.

- RNA poate fi antrenată pentru a rezolva o problemă. Dacă sunt necesare explicații relative la modul de raționament, se aplică unui SE datele de intrare împreună cu răspunsul RNA, iar printr-un proces de înlănțuire logică SE "inventează" o explicație pentru situația prezentată.

- În strategia expertului artificial, RNA este antrenată să rezolve o problemă apoi răspunsurile acestei sunt analizate pentru a extrage un set de reguli.

1.3. Mediul NeuroShell 2

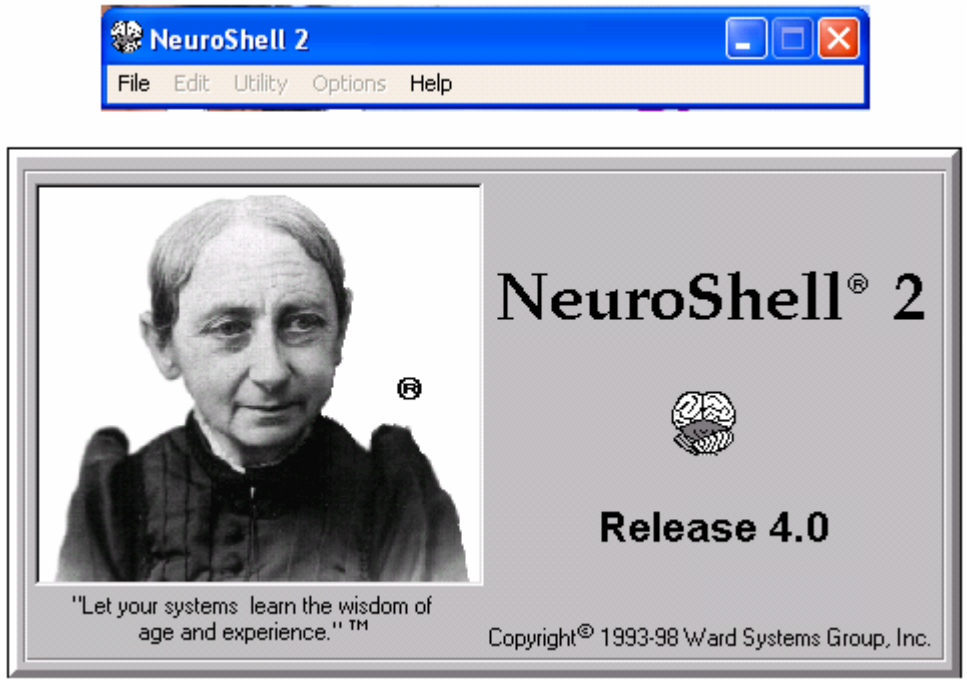


Fig. 1.11. Neuro Shell - afişajul iniţial al programului

Asemeni creierului uman, reţelele neuronale nu garantează întotdeauna un răspuns absolut corect, mai ales dacă modelele sunt, sub o formă sau alta, incomplete sau conflictuale. Reţelele neuronale caută modele în setul de date de intrare, învaţă aceste modele şi-şi crează abilitatea de a clasifica corect, de a prezice sau decide asupra noilor modele apărute.

Pentru a activa lansarea mediului NeuroShell se realizează "dublu clic" pe icoana , a cărei imagine reprezintă un creier uman şi care este plasată în grupul programelor afişate, imediat după instalarea programului. Va apărea o bară mică de meniuri, detaşată de restul afişajului pe ecran, Fig. 1.11. Se selectează în continuare meniul **File** din meniul principal, din care se alege fie o problemă nouă, fie se deschide o problemă existentă pe disc şi care are deja ataşat un nume.

1.3.1. Numele unei probleme

Fiecare problemă se referă la o aplicaţie a unei reţele neuronale. De-a lungul stagiilor variate de procesare a problemei, NeuroShell construieşte câteva fişiere asociate aplicaţiei iniţiale. Fişierele asociate sunt diferite unele faţă de celelalte, insă numele lor este identic, având doar extensiile diferite.

În cazul salvării unei probleme noi sistemul asociază denumirii problemei extensia DSC (description).

1.3.2. NeuroShell – Meniul Principal

Fereastra meniului principal oferă trei moduri de utilizare ale mediului NeuroShell: **Beginner's Neural Networks** (reţele neuronale pentru începători), **Advanced Neural Networks** (reţele neuronale pentru avansaţi) şi **Runtime System** (sistem cu timp de rulare).Toate cele trei moduri de utilizare sunt concepute pe baza unor subprograme înlănţuite numite "module", fiecare modul fiind reprezentat de câte o icoană. Fig.1.12.

Sugestie: este indicat ca fiecare utilizator să parcurgă în prima etapă modulul pentru începători.

Pentru a utiliza un modul se realizează “dublu clic” cu mouse-ul pe icoana respectivului modul.

Pentru a beneficia de facilitățile **Help** pentru utilizarea modulelor, se selectează meniul **Help** după care se selectează **Current Context.**

La apariția unui grup de icoane pe ecran, ordinea operării acestora este de la stânga la dreapta , iar ordinea de operare a icoanelor într-o coloană este de sus în jos.

Observație: Nu este absolut necesară utilizarea tuturor coloanelor care apar pe ecran pentru a crea o aplicație bazată pe o rețea neuronală funcțională. Multe module sunt funcționale, depinzând de tipul aplicației create.

NeuroShell este un mediu modular. După activarea unei icoane programul afişează un nou modul. Fiecare modul are una sau mai multe ferestre.

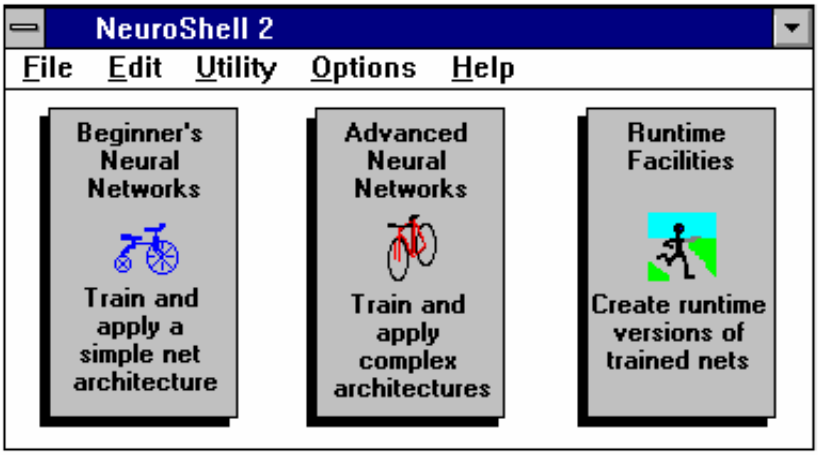


Fig. 1.12. NeuroShell 2 – meniu principal.

1.3.3. Rețele Neuronale pentru Începători (Beginner's Neural Network)

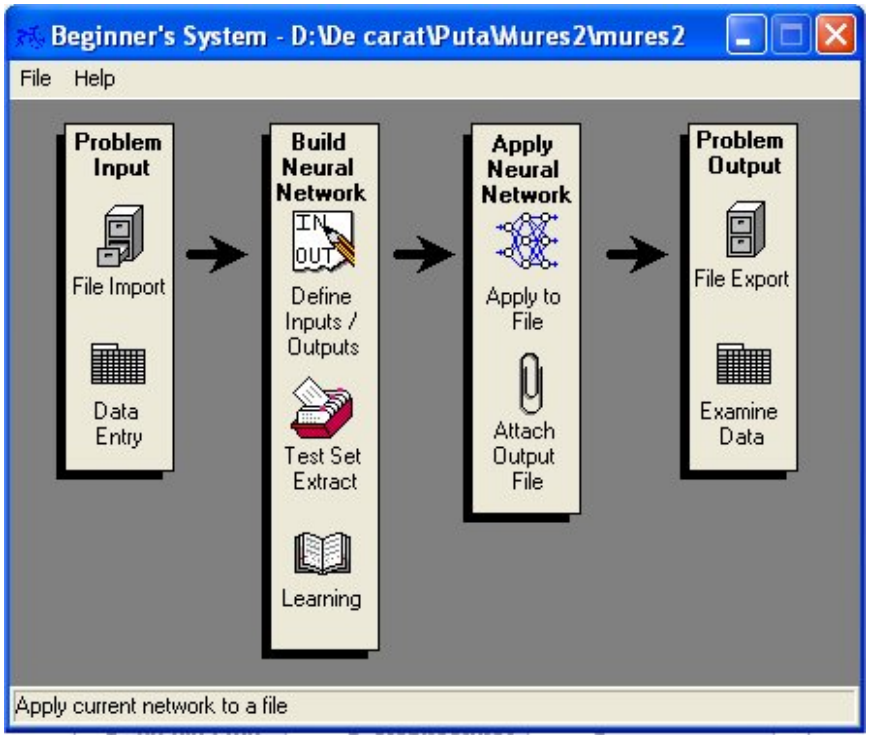


Fig. 1.13. Fereastra modulului „Beginer's Neural Networks".

Modulul Rețelelor Neuronale pentru Începători (**Beginner's Neural Network**), fig.1.13, reprezintă un set simplificat de proceduri pentru construirea şi executarea unei aplicații pentru o rețea neuronală puternică şi completă, totul fiind integrat într-un sistem mult mai uşor de utilizat decât un sistem avansat.

Sistemul pentru începători utilizează o rețea cu retropropagare a erorii, având trei straturi şi o arhitectură universală cu abilități de generalizare reuşite pentru o varietate largă de probleme.

Sistemul pentru începători prestabileşte parametrii rețelei după cum urmează: *rata de învățare* (**learning rate**), *inerția rețelei* (**momentum**) şi *numărul de neuroni ascunşi* (**hiden neurons**).

1.3.4 Rețele Neuronale Avansate (Advanced Neural Networks)

Modulul Rețelelor Neuronale pentru Avansați (**Advanced Neural Network**) prezintă un tablou extins de elemente cu ajutorul cărora se pot implementa rețele puternice, cu trăsături de procesare cum sunt : translatarea simbolurilor în fişierele de date, implementarea unor reguli pentru crearea de noi variabile sau analiza grafică a rezultatelor obținute. În fig. 1.14 este prezentat modulul „**Advanced Neural Networks**" cu toate facilitățile de care dispune.

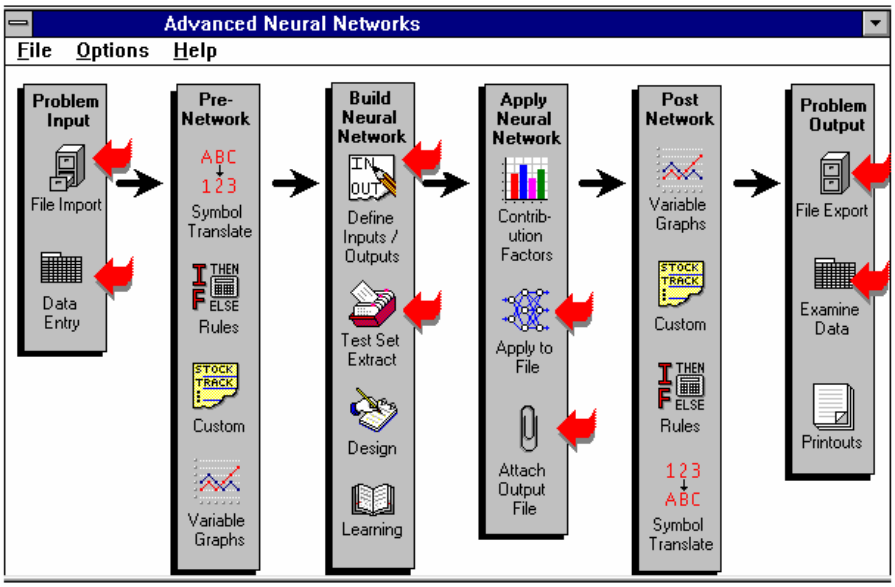


Fig. 1.14. Fereastra modulului „Advanced Neural Networks”.

Modulul „Advanced Neural Network” oferă 12 arhitecturi de implementare al modelului „backpropagation”, care include următoarele arhitecturi pentru rețele: „feedforward”, „jump conection” şi rețele recurente. În plus, modulul oferă încă patru arhitecturi de implementare a altor modele complexe după cum urmează:

- General Regression Nets,
- Probabilistic Nets,
- Kohonen’s Unsupervised Learning,
- Polinomyal (GMDH) Nets.

Activarea icoanei „**Design**” apelează o fereastră cu două icoane. De aici se poate selecta arhitectura rețelei neuronale, respectiv criteriile de antrenare si de oprire a antrenării conform unor parametrii atinşi.

Prin selectarea icoanei „**Arhitecture and Parameters**”, se etalează un tablou care conține 16 arhitecturi posiblile. În acest pas se pot modifica parametrii şi factorii de antrenare ai rețelei.

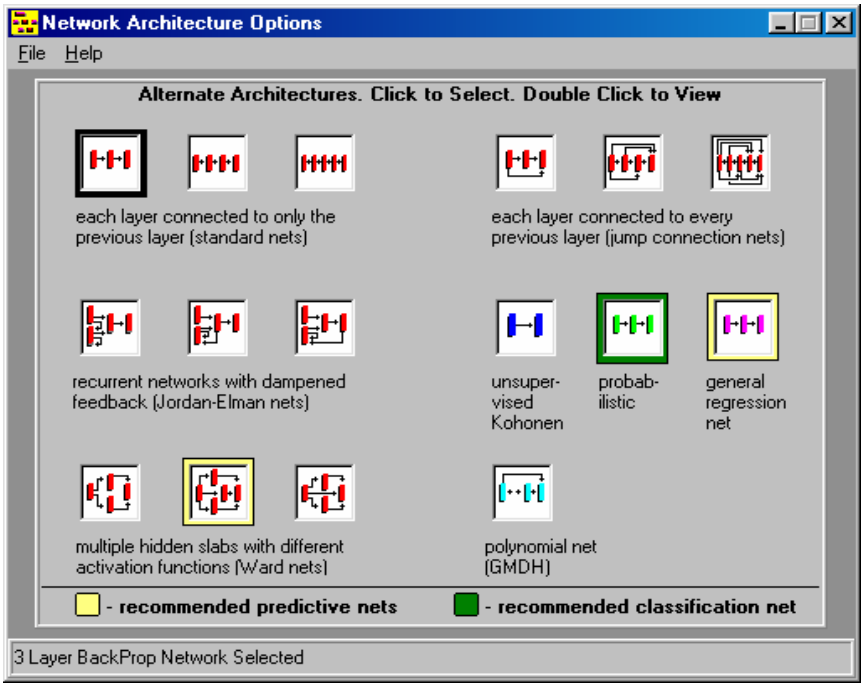


Fig. 1.15. Arhitecturi de rețea disponibile în mediul NeuroShell 2.

Selectarea se realizează prin deplasarea chenarului negru pe structura dorită., fig. 1.15. NeuroShell 2 recomandă anumite tipuri de rețele în funcție de natura problemei. Astfel, pentru o rețele predictivă NeuroShell sugerează o arhitectură – backpropagation cu trei neuroni, având funcții de activare diferite, iar pentru rețele folosite în clasificări, NeuroShell sugerează o arhitectură de rețea neuronală probabilistică (PNN).

1.3.5. Facilități „Runtime" ale mediului NeuroShell 2

Prin activarea modulului „**Runtime Facilities**", se genereză o icoană pentru activarea serverului DLL (Dynamic Link Library). Serverul DLL creează un fişier cu extensia .DEF, care face posibilă apelarea rețelei antrenate, cu NeuroShell 2, dintr-un program scris de programator care suportă Dynamic Link Libraries, cum este, de exemplu, Microsoft Excel, C, C++, Visual Basic, etc. De asemenea, acest fişier este utilizat şi pentru extragerea datelor în scopul trasării graficelor tridimensionale.

Dacă DLL este apelat dintr-un cod scris de programator, limbajul utilizat pentru scrierea programului trebuie să fie capabil să apeleze aceste librării prin pointeri de depărtare „**far pointers**".

Serverul DLL oferă şi câteva funcții care vor fi descrise în cele ce urmează:

- funcția „**OpenNet**"

Această funcție citeşte fişierul cu extensia .DEF şi realizează setările corespunzătoare. Ea returnează un număr pentru rețea, care pe tot parcursul, este o referință a acestei rețele. „**OpenNet**" precizează numărul de intrări aşteptate de rețea şi numărul de ieşiri prin care aceasta răspunde, programatorul cunoscând aceste valori.

- funcția „**FireNet**"

Odată rețeaua deschisă cu „**OpenNet**", se utilizează „**FireNet**" pentru a se indica rețelei intrările şi respective pentru a primi ieşirile, ambele variabile fiind incluse într-o martice „**DOUBLE PRECISION FLOATING POINT**", inițializată în programul sursă.

- funcția „**CloseNet**"

Această funcție se apelează când datele necesare procesate de rețeaua neuronală au fost preluate şi „serviciile” acesteia nu mai sunt necesare. „**CloseNet**” eliberează spațiul din memorie alocat pentru rețea, iar la următoarea apelare a rețelei se va activa funcția „**OpenNet**” din nou.

1.3.6. Moduri de îmbunătățire a rezultatelor

Câteva soluții de îmbunătățire a performanțelor rețelei sunt prezentate în continuare :

- Calibrarea rețelei
- Stabilirea unor variabile mai eficiente pentru predicția dorită, şi/sau determinarea unor metode mai bune de reprezentare decât cele actuale.
- Ajustarea ratei de învățare, a inerției sau a numărului de neuroni ascunşi, pentru un nou model de antrenare.

NeuroShell utilizează calibrarea pentru a optimiza ieşirile, prin aplicarea rețelei curente a unui set de teste independente în timpul antrenării. Acest set de date de test se crează automat utilizând modulul „Test Set Extract”. Prin calibrare se obține rețeaua optimă pentru date, în setul de test, ceea ce înseamnă că rețeaua este capabilă să generalizeze cu succes noile date şi să returneze rezultate foarte bune.

Calibrarea realizează acest lucru, prin calcularea mediei erorii pătratice, pentru toate ieşirile actuale şi prezise.

Pentru rețele de tip „Backprpagation”, rețeaua salvează de fiecare dată, când se realizează, un nou minim pentru media erorii (sau media erorii pătratice). Pentru a folosi calibrarea, trebuie setat câmpul Calibration Test Interval, având semnificația perioadei de evaluare a setului de test. Practica a demonstrat că rețelele neuronale realizează bune predicții, dacă acest câmp setat cu valori între 50 şi 200. Secretul construirii inuei rețele neuronale este cel al identificării momentului de intrerupere a antrenării. Dacă se antrenează prea puțin, rețeaua nu poate învăța toate modelele. Dacă se antrenează prea mult, rețeaua va învăța şi zgomote. Programul NeuroShell atenuează aceste erori prin intermediul calibrării.

1.4. Aplicație - Antrenarea unei RNA pentru identificarea satisfacției clienților în firma “X” cu ajutorul mediului NeuroShell

Domeniul în care firma “X” îşi desfăşoară activitatea este producția şi vânzarea de calculatoare şi respectiv de componente adiacente acestora (imprimante, scanere etc.). Dotarea cu calculatoare în România este încă mult sub media de pe Europa de Est. Deşi vânzările au crescut, ele nu se ridică la nivelul aşteptat de firmă.

Elementul cheie în decizia de achiziționare a unui calculator personal a fost şi rămâne prețul. Deşi într-o perioadă de timp mulți consumatori s-au orientat spre prețuri mai mici, adică calculatoarele „no name” care invadaseră piața, această mentalitate s-a schimbat, cumpărătorii constatând din practica curentă importanța service-ului şi garanției care au devenit indispensabile.

Una dintre problemele cu care firma se confruntă este aceea de măsurare a satisfacției clientului. Această problemă a apărut odată cu implementarea noii politici de calitate, adică a standardului ISO 9001-2000, a cărui obiectiv principal îl constituie satisfacția clientului.

S-a pus problema creării unei Rețele Neuronale Artificiale care să măsoare gradul de satisfacție a clienților în funcție de toți parametrii care influențează acest proces. În urma unui studiu care a durat 3 luni, au fost identificați câțivai parametrii care influențează puternic gradul de satisfacție a clientului. Acesti parametrii depind de tipul următoarelor componente, care influențează major funcția de întrebuințare a unui sistem de calcul: memoria, procesoarele, placa video; HDD-uri; placa de sunet. Aceste componente au fost cuantificate în cadrul unor anumite clase, în scopul codificării lor ca date de intrare în cadrul Rețelei Neuronale Artificiale.

Utilitatea creării rețelei neuronale este dată de furnizarea informațiilor necesare firmei în negocierile cu clienții şi respectiv de creştere a calității produselor.

Având în vedere faptul că numărul de variabile aplicate în scopul rezolvării problemei este relativ mic, s-a utilizat modulul aferent rețelelor neuronale pentru începători „**Biginner's Neural Networks**". Modulul selectat pentru rezolvarea problemei nu este mai puțin performant decât cel al modulului rețelelor neuronale avansate, ci este cel mai potrivit pentru rezolvarea problemei definite.

1.4.1. Datele problemei

Datele de ieşire

Ieşirea rețelei reprezintă predicția indicelui de satisfacție a clientului asupra produselor firmei. Pentru aceasta s-au analizat mai multe tipuri de componente care au fost clasificate în clase de cuantificare (**vezi Anexa 1**), şi s-a creat o rețea neuronală care să aibă ca dată de ieşire coeficientul SCVP – satisfacția clientului post vânzare.

Datele de intrare

Datele de intrare pentru antrenarea rețelei sunt considerate ca cei mai importanți factori care influențează coeficientul SCVP. Aceştia sunt prezentați în lista care urmează:

- **clasa de cuantificare**, obținută prin încadrarea componentelor pe clase;
- perioada de garanție acordată de către furnizori pentru componenta respectivă (**garanție**);
- **numărul de intervenții** care s-au realizat pe parcursul a 3 luni de zile la fiecare tip de componentă analizată şi vândută (**nr. intervenții**);
- durata parcursă din momentul sesizării şi momentul rezolvării problemei (**DR**);
- timpul contractat de rezolvare a sesizării (**DRC**),
- nivelul defecțiunii echipamentului - care este 1 dacă prin defecțiune clientul a pierdut informațiile, este ½ dacă clientul nu a pierdut informațiile dar nu poate utiliza produsul şi este ¼ dacă clientul nu a pierdut informațiile şi poate utiliza produsul dar nu la performanțele oferite (**Pi**);

Există şi alți factori care influențează satisfacția clienților, însă în scopul simplificării aplicației s-a stabilit că factorii prezentați mai sus sunt cei mai importanți factori de influență.

1.4.2. Selectarea modului de lucru

După determinarea datele de intrare, de ieşire şi respectiv selectarea modelului rețelei neuronale, s-a trecut la următorul pas, cel al introducerii datelor de antrenare. În acest scop a fost lansat programul NeuroShell 2, respectiv a fost selectat modulul **Beginner's Neural Network**. fig. 1.16.

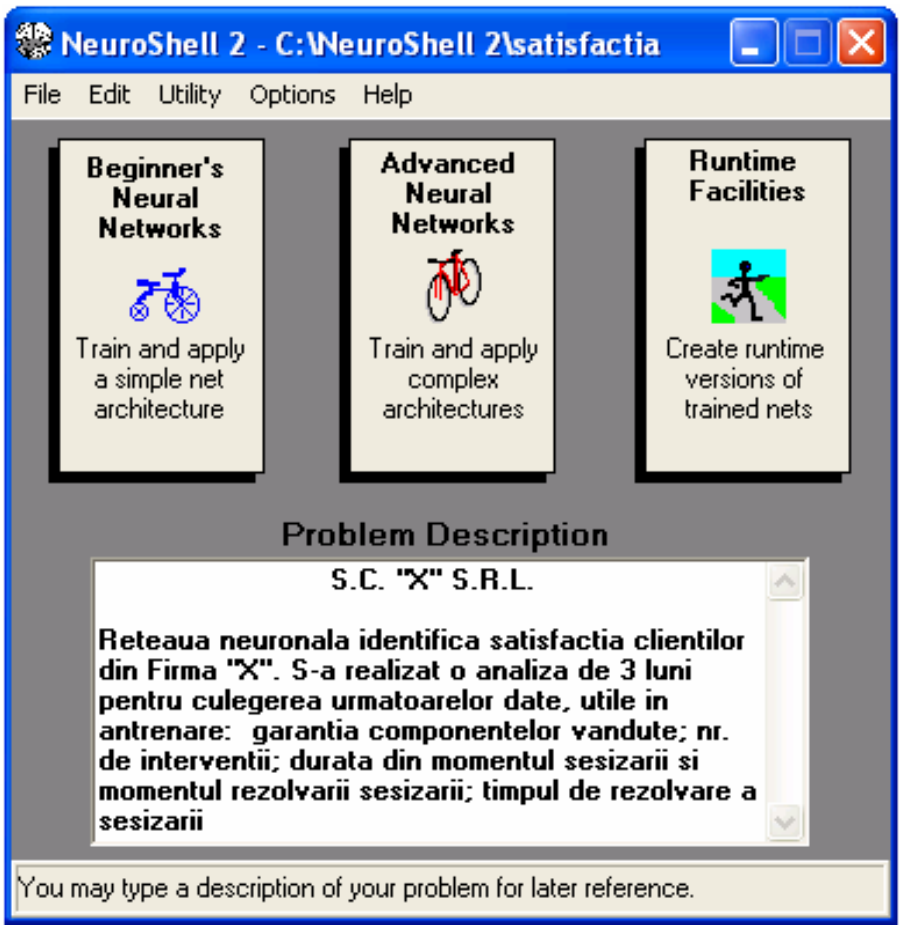
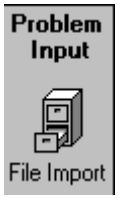


Fig. 1.16. Prezentarea meniului aferent modulului Beginner's

După selectarea modului **Beginner's Neural** Networks se deschide o fereastră, care conține patru coloane, reprezentând module functionale pentru crearea rețelei neuronale (fig. 1.17), care sunt prezentate în continuare.

1.4.3. Importarea fişierului de antrenare a RNA



Prima icoană activează modulul de importare a fişierului de antrenare a RNA. Fişierul de antrenare a rețelei conține toate datele de intrare şi ieşire pe care rețeaua va trebui să le învețe. Pentru activarea acestui modul se realizează "dublu clic" pe icoana „**File Import**".

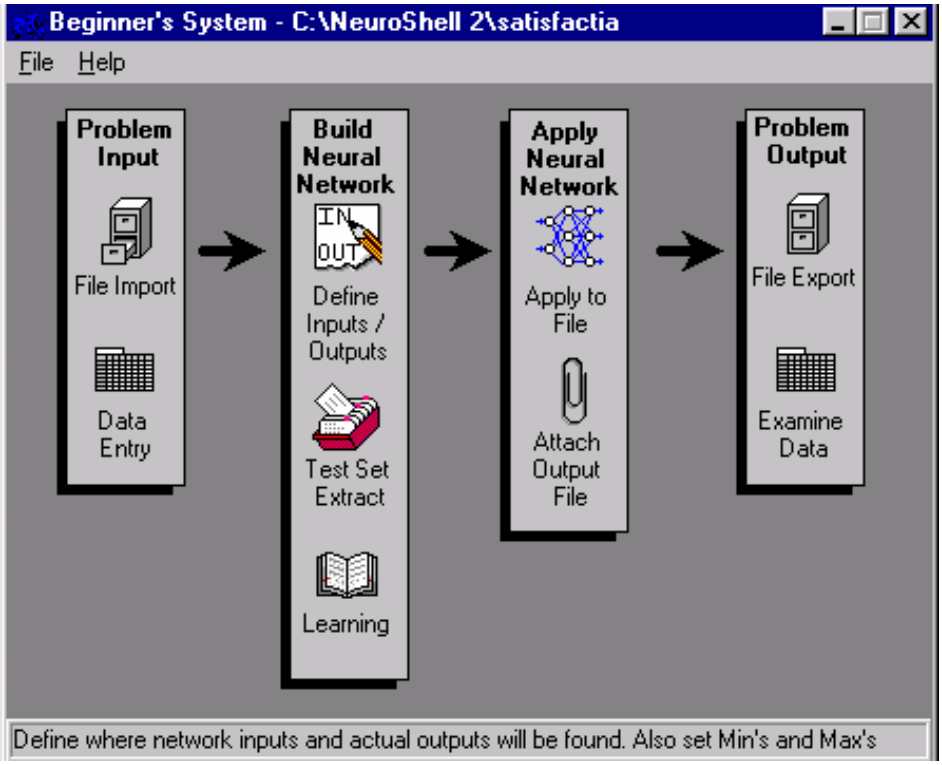


Fig. 1.17. Prezentarea meniului aferent modulului Beginner's

Se deschide caseta de dialog din care se selectează „**Spreadsheet Files**" pentru importarea datelor din programul Microsoft Excel. (**fig. 1.18 şi Anexa 1**)

În continuare se selectează fişierul corespunzător datelor, dar având în vedere că un document din Excel are mai multe foi (Sheet) se va selecta Sheet-ul corespunzător datelor. În cazul aplicaţiei prezentate se selectează Sheet 8. (**fig. 1.18**), după care se generează importul datelor (**fig. 1.19**) prin selectarea meniului **Import**, respectiv al comenzii **Begin Import.**

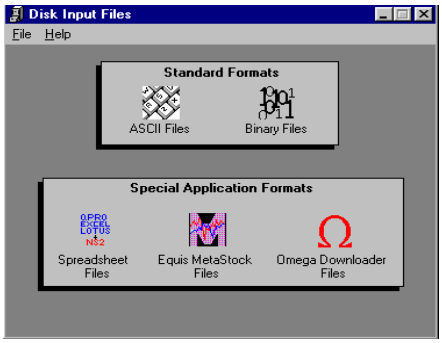


Fig. 1.18. Caseta pentru generarea importului de date

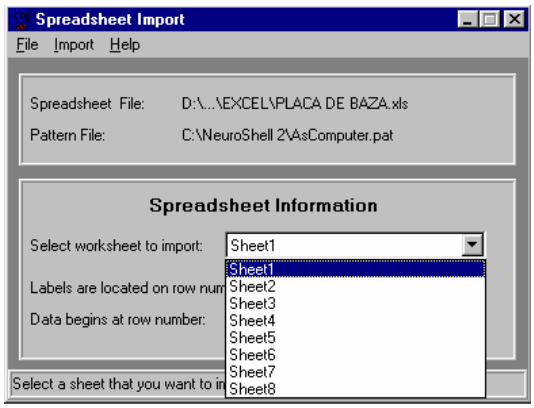


Fig. 1.19. Selectarea foii de calcul

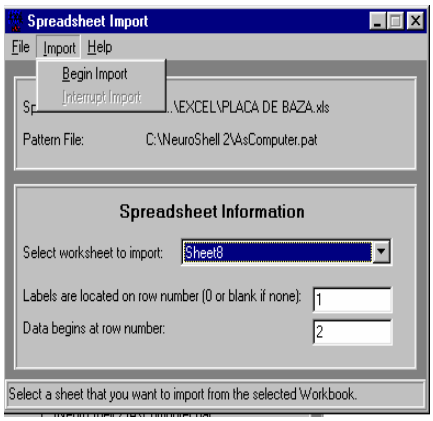
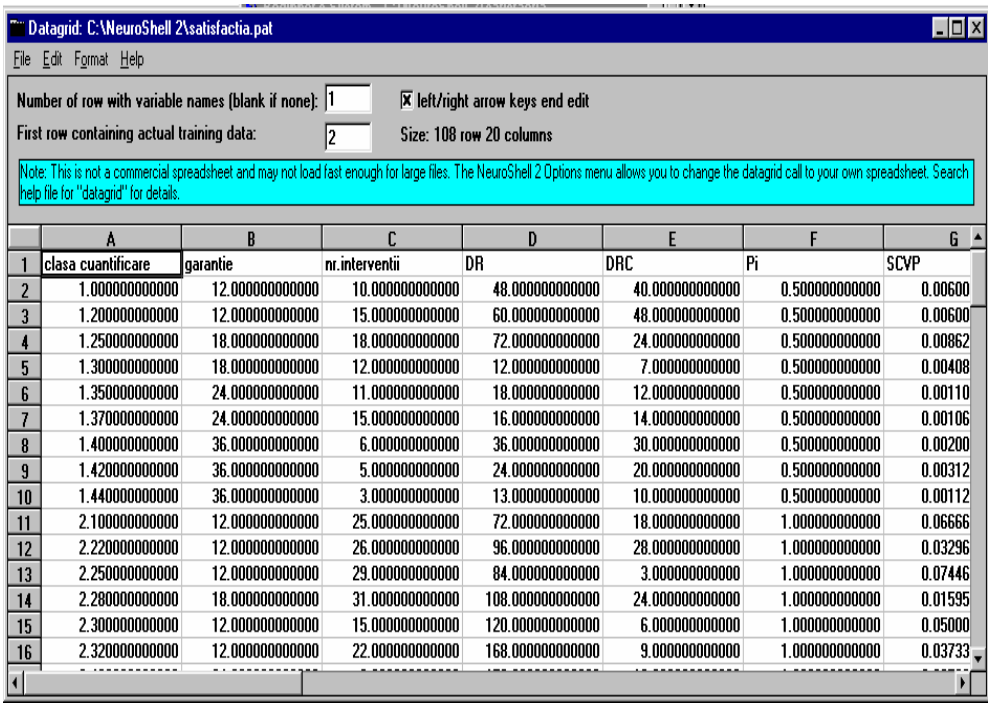


Fig. 1.20. Activarea importului



	A	B	C	D	E	F	G
1	clasa cuantificare	garantie	nr.interventii	DR	DRC	Pi	SCVP
2	1.000000000000	12.000000000000	10.000000000000	48.000000000000	40.000000000000	0.500000000000	0.00600
3	1.200000000000	12.000000000000	15.000000000000	60.000000000000	48.000000000000	0.500000000000	0.00600
4	1.250000000000	18.000000000000	18.000000000000	72.000000000000	24.000000000000	0.500000000000	0.00862
5	1.300000000000	18.000000000000	12.000000000000	12.000000000000	7.000000000000	0.500000000000	0.00408
6	1.350000000000	24.000000000000	11.000000000000	18.000000000000	12.000000000000	0.500000000000	0.00110
7	1.370000000000	24.000000000000	15.000000000000	16.000000000000	14.000000000000	0.500000000000	0.00106
8	1.400000000000	36.000000000000	6.000000000000	36.000000000000	30.000000000000	0.500000000000	0.00200
9	1.420000000000	36.000000000000	5.000000000000	24.000000000000	20.000000000000	0.500000000000	0.00312
10	1.440000000000	36.000000000000	3.000000000000	13.000000000000	10.000000000000	0.500000000000	0.00112
11	2.100000000000	12.000000000000	25.000000000000	72.000000000000	18.000000000000	1.000000000000	0.06666
12	2.220000000000	12.000000000000	26.000000000000	96.000000000000	28.000000000000	1.000000000000	0.03296
13	2.250000000000	12.000000000000	29.000000000000	84.000000000000	3.000000000000	1.000000000000	0.07446
14	2.280000000000	18.000000000000	31.000000000000	108.000000000000	24.000000000000	1.000000000000	0.01595
15	2.300000000000	12.000000000000	15.000000000000	120.000000000000	6.000000000000	1.000000000000	0.05000
16	2.320000000000	12.000000000000	22.000000000000	168.000000000000	9.000000000000	1.000000000000	0.03733

Fig. 1.21. Introducerea datelor în Datagrid

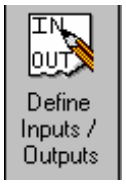
1.4.4. Introducerea datelor

După importarea fiţierului de antrenare se activeză următoarea iconă din Fig.1.21, "**Data Entry**" . Se deschide un tabel asemănător cu foaia de lucru a programului Excel. În cadrul acestui modul se stabilesc denumirile variabilelor şi se introduc pe primul rând al coloanelor destinate acestora.

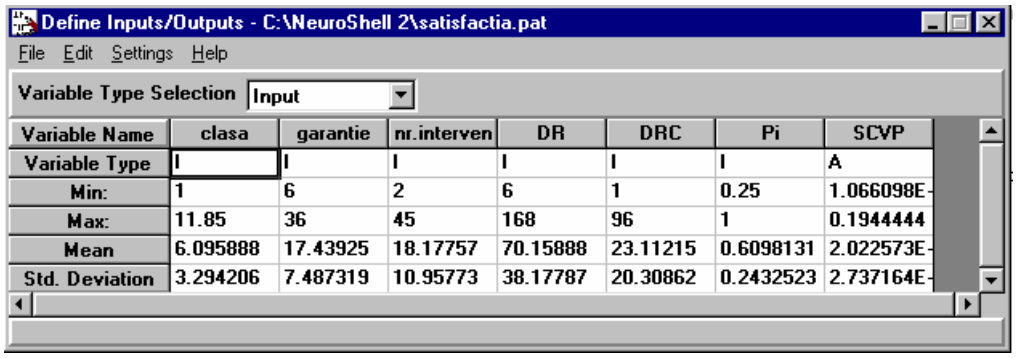
Se setează **„1"** în căseta corespunzătoare numărului de rânduri utilizate pentru numele variabilelor şi **„2"** pentru primul rând care conţine date active.

Pentru aplicația prezentată, în primul rând s-a tipărit „**clasă cuantificare", „garanție", „nr. intervenții", „DR", „DRC", "Pi" şi „SCVP"**. Începând cu rândul doi s-au introdus datele măsurate în perioada analizată de 3 luni şi sortate în funcție de clasele de cuantificare. În final, lista conține un rând pentru numele variabilelor şi **107** rânduri pentru date. Pentru salvarea datelor se selectează meniul **File**, respectiv opțiunea **Save.** Fişierul a fost denumit „**satisfactia.pat".**

1.4.5. Definirea intrărilor şi ieşirilor



În cadrul sistemului informatic NeuroShell 2, utilizatorul trebuie să specifice care dintre coloanele de date introduse sunt variabile de intrări şi care sunt variabile de ieşiri. Se activează modulul **„Define Inputs and Outputs"**, operație în urma căreia se afişează toate denumirile coloanelor. În această fereastră se stabileşte, respective se setază prin intermediul abrevierilor **I** sau **A** dacă o coloană conține variabile de intrare, respective de ieşire. pentru fiecare coloană. Fig. 1.22.



Variable Name	clasa	garantie	nr.interven	DR	DRC	Pi	SCVP
Variable Type	I	I	I	I	I	I	A
Min:	1	6	2	6	1	0.25	1.066098E-
Max:	11.85	36	45	168	96	1	0.1944444
Mean	6.095888	17.43925	18.17757	70.15888	23.11215	0.6098131	2.022573E-
Std. Deviation	3.294206	7.487319	10.95773	38.17787	20.30862	0.2432523	2.737164E-

Fig. 1.22. Caseta de dialog Input/Output

Tot în acestă fereastră se stabilesc şi se înregistrează valorile de minim şi maxim pentru fiecare variabilă în parte. Deoarece pentru rețelele neuronale variabilele trebuie determinate într-un **interval de la 0 la 1 sau –1 la 1**, rețeaua trebuie să cunoască aceste minime şi maxime reale, pentru a atribui valori din aceste intervale. Acest lucru se poate raliza manual, sau se determină automat selectând opțiunea „**Compute mins/maxes**". Prin această operație se determină implicit valoarea medie **„Mean"**, şi deviația standard **„Std. Deviation"**, pentru fiecare dintre variabile.

În general se definește un interval foarte strâns în jurul datelor reale de minim şi maxim. Se pot specifica valori minime şi maxime care sunt uşor deasupra sau sub valorile reale, pentru a aloca un interval mai larg pentru predicțiile următoare. Dacă valorile de minim şi maxim nu sunt în jurul valorilor reale, rețeaua îşi pierde abilitatea de a recunoaşte diferențele mici între datele de interes.

1.4.6. Antrenarea rețelei

Antrenarea rețelei se realizează prin activarea icoanei **„Learning"**. Prin activarea modulului de antrenare (învățare) se inserează automat modulul „Define **parametrii** ai intrări şi cel de ieşiri, din fişierul cu extensia „.mmx", creat de Inputs/Outputs". Înainte de a începe antrenarea se mai setează câțiva parametrii ai modulului „Learning". Aceştia sunt :

- Specificarea **complexității problemei**, prin activarea unei boxe corespunzătoare nivelului de complexitate a problemei. Datorită numărului mai redus de variabile se consideră că rețeaua neuronală

are un nivel de complexitate scăzut ţi s-a activat boxa **„Very Simple”.** Prin selectarea acestei opţiuni se activează automat factorii „**Learning rate**” (rata de învăţare) şi „**Momentum**” (inerţia) la valorile **0,6** respectiv **0,9.**

- **Numărul de neuroni ascunşi** ai reţelei se poate seta atât manual, cât şi automat, activând caseta „**Set Number of Hidden Neurons to Default**”. Numărul de neuroni ascunşi, a fost calculat după formula :

$$N = \frac{1}{2}(IN + OUT) + \sqrt{N_1} \quad ,$$

unde :N – numărul de neuroni ascunşi,
IN – numărul de intrări,
OUT – numărul de ieşiri,
N_1 – numărul a câte linii de date s-au introdus pentru variabilele modelului

În cazul aplicaţiei prezentate s-a optat pentru setarea automată, obţinându-se un număr de **13** neuroni ascunşi.

- Datorită faptului că datele sunt sortate crescător conform convenţiilor de cuantificare, s-a optat pentru o compilaţie **random** (pe sărite) a acestor valori, pentru ca reţeaua să fie aptă de a prezice indiferent de diferenţele de clasă a componentelor apărute.
- Pentru această sesiune de antrenare opţiunea **Calibration Interval** (intervalul de calibrare)a fost setată la **0**.

În urma setării parametrilor prezentaţi mai sus s-a pornit antrenarea reţelei neuronale din meniul **Train** prin selectarea opţiunii **Start Training**. Fig. 1.23.

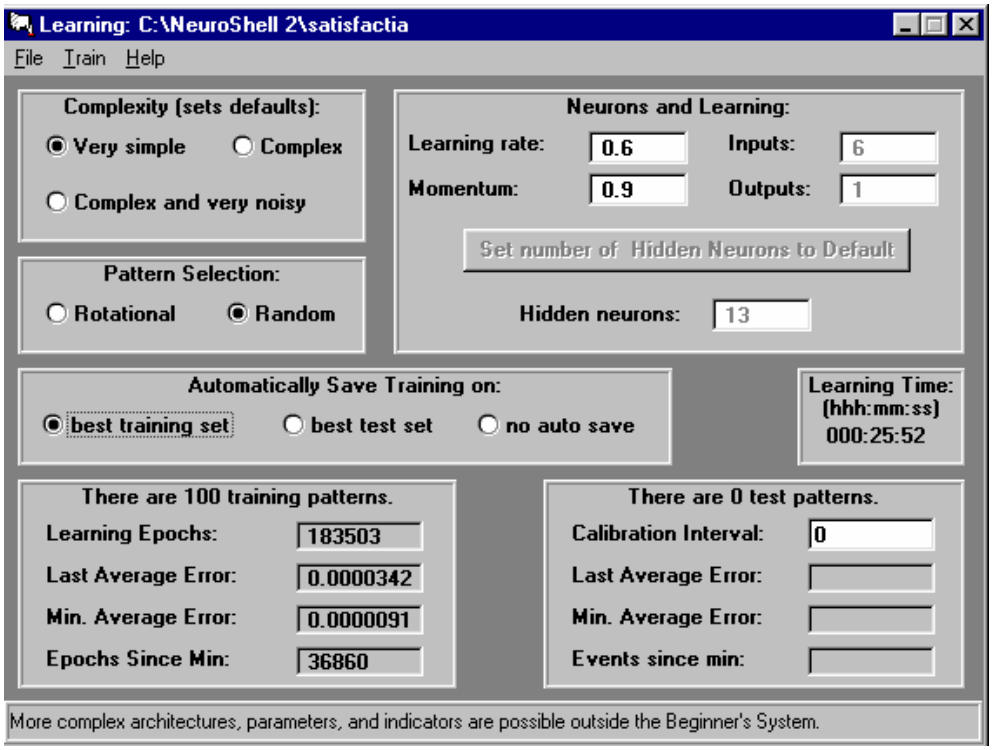


Fig. 1.23. Tabloul de antrenare a reţelei neuronale

Momentul opririi rețelei s-a hotărât de către utilizator, pe parcursul evoluției antrenării, observându-se că după **2568 de epoci** media erorii minime a devenit aproximativ **0.0000013** şi s-a menținut de peste **400 de epoci.** Din acest punct rețeaua neuronală părea că nu mai face progrese în ceea ce priveşte media erorii minime şi, deci, antrenarea s-a oprit prin selectarea opțiunii **Intrerrupt Training** a meniului **Train**.

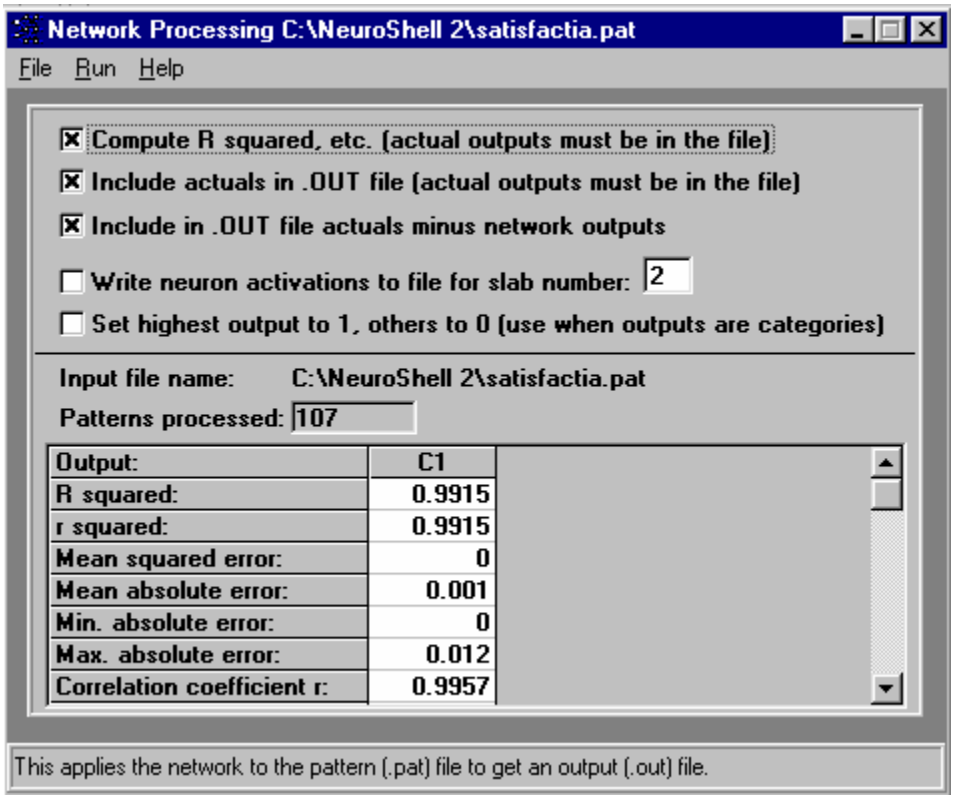
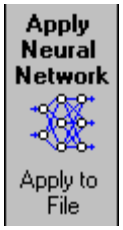


Fig. 1.24. Caseta de dialog Network Processing

1.4.7. Procesarea datelor antrenate

Procesarea datelor antrenate se realizează prin selectarea opțiunii modulului funcțional **Apply Neural Network,** în care se verifică dacă rezultate eficiente. Prin activarea acestui modul se deschide o casetă de **Processing.** Fig. 1.24. Din meniul **Run** se selectează **Start** Modulul **Apply to File** a procesat automat fişierul **satisfactia.pat**, care date introdus.



Pentru fişierul **satisfactia.pat**. **R squared** este egal cu **0.9915**. succesive a rețeaua a avut dialog **Network Processing.** a fost primul set de NeuroShell înregistrează paramentru **R squared** pentru a compara această rețea cu altele care ar putea fi create mai târziu.

R squared, coeficient de determinări multiple, este un indicator statistic aplicat de obicei unei analize regresive multiple. El compară acuratețea modelului creat cu acuratețea unui model benchmark ordinar. Un rezultat *perfect* al rețelei neuronale returnează valoarea **1** pentru **R squared**, iar un rezultat *bun* este în jurul acestei valori. Un rezultat în jurul valorii **0** este nesatisfăcător.

1.4.8. Ataşarea fişierelor pentru vizualizarea evoluției datelor

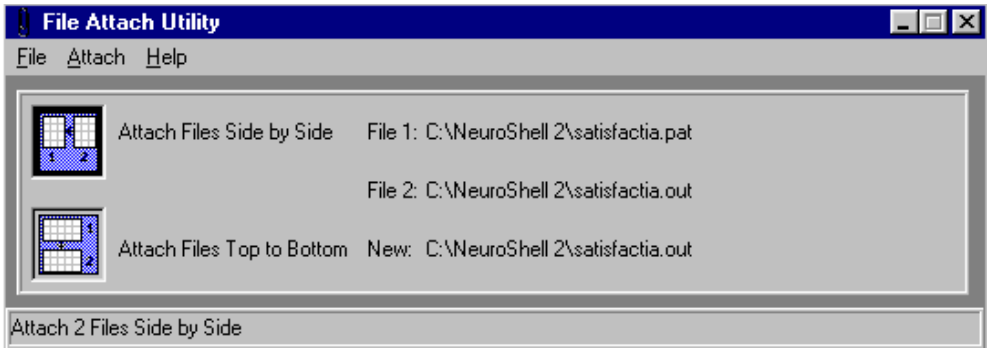


Fig.1.25. Ataşarea fişierului de ieşire de cel original

Activarea icoanei **Attach Output File** etalează un tablou numit **File Attach Utility**, Fig.1.25, care oferă posibilitatea de a ataşa fişierul vechi, conținând datele introduse de către utilizator **satisfactia.pat** (nume_fişier.pat), de cel nou, în care avem datele procesate **satisfactia.out** (nume_fişier.out), în două moduri :

- unul lângă celălalt,
- unul sub celălalt,

Pentru aplicația identificării satisfacției clienților s-a optat pentru primul mod de comparare (fişierele sunt ataşate unul lângă celălalt) pentru o mai bună vizualizare, prin comparație, a datelor.

După selectarea variantei dorite, din meniul **Attach** se activează **Attach File**, operație care creează un nou fişier cu extensia **.out**. Fişierele care se compară sunt determinate automat de către NeuroShell 2, rezultând:

- **satisfactia.pat** – fişierul original,
- **satisfactia.out** – fişierul conținând răspunsul rețelei

Modulul funcțional **Attach Output File** se poate utiliza de asemenea pentru a combina oricare alte 2 fişiere interne de registru tabelar. Fişierele care trebuie comparate vor fi selectate din caseta de dialog **File Attach Utility**. Selecția se realizează din meniul **File** opțiunea **Select Output File.**

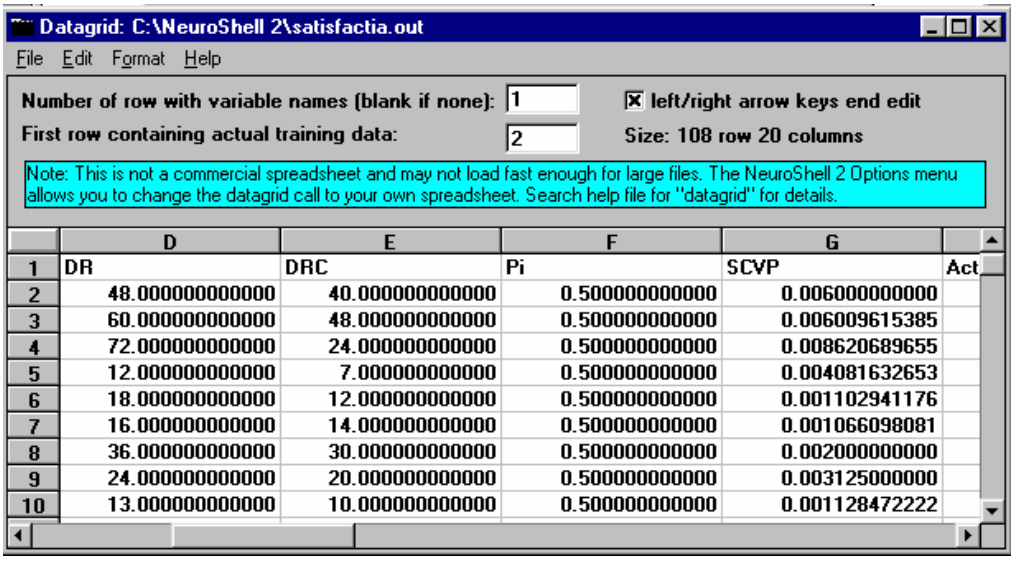
1.4.9. Examinarea datelor

Pentru a vizualiza fişierul **satisfactia.out** se activează icoana modulului **Examine Data**. Se deschide un registru tabelar, Fig.1.26, stângă coloanele inițiale ale fişierului **satisfaction.pat** iar în partea corespunzătoare fişierului **satisfaction.out.** Coloanele din fişierul de prezentate în lista următoare:

- coloana "**Actual"** (ieşire actuală) conține valorile pentru variabilele setului de antrenare pe care trebuie să le prezică rețeaua;
- Coloana "**Network**" (rețea) conține valorile pentru variabilele de ieşire prezise de rețea;

corespunzătoare având în partea dreaptă coloanele ieşire sunt de ieşire ale

- Coloana "**Act-Net**" (Actual – Network) reprezintă diferența dintre coloanele "Actual" şi "Network"; cu cât aceste valori sunt mai mici, cu atât valorile scoase de rețea sunt mai corecte şi rețeaua este mai bine antrenată.



	D	E	F	G	
1	DR	DRC	Pi	SCVP	Act
2	48.000000000000	40.000000000000	0.500000000000	0.006000000000	
3	60.000000000000	48.000000000000	0.500000000000	0.006009615385	
4	72.000000000000	24.000000000000	0.500000000000	0.008620689655	
5	12.000000000000	7.000000000000	0.500000000000	0.004081632653	
6	18.000000000000	12.000000000000	0.500000000000	0.001102941176	
7	16.000000000000	14.000000000000	0.500000000000	0.001066098081	
8	36.000000000000	30.000000000000	0.500000000000	0.002000000000	
9	24.000000000000	20.000000000000	0.500000000000	0.003125000000	
10	13.000000000000	10.000000000000	0.500000000000	0.001128472222	

Fig. 1.26. Examinarea fişierului de ieşire

După vizualizarea comparației dintre fişierul de intrare **satisfaction.pat** şi fişierul de ieşire **satisfaction.out** se va ataşa de asemenea fişierul creat la începutul aplicației de către programul NeuroShel **satisfaction.pro**. În acest sens se reia pasul procesarea datelor antrenate (§ 1.4.7) şi în modulul funcțional **Network Processing** se selectează din meniul **File** comanda **Select Alternate Pattern File**. Fişierul alternativ este **satisfaction.pro.** S-a pornit procesarea şi s-a evaluat valoarea **R squared**, în cazul aplicației **satisfaction.pro** a rezultat **0,98**, fiind considerată o valoare bună.

Pentru evaluarea rețelei mai există o variantă alternativă, care parcurge din nou paşii prezentați încă de la fişierul de introducere a datelor, creat în modulul **Data Entry**. În continuare sunt prezentați paşii percurşi pentru aplicația **satisfaction.**

- Pas 1: crearea unui nou fişier în modulul **Data Entry** (cca. 20% din datele de test) asemănătoare cu primele, pentru a verifica corectitudinea rețelei. Aceste date se regăsesc în **Anexa 2.** Noului fişier de date trebuie să I se atribuie aceeaşi denumre cu cea a aplicației, însă cu o altă extensie aleatoare introdusă de către utilizator. (ex. **satisfaction.xxx**)
- Pas 2: se trece la modulul de procesare a datelor, activat prin icoana **Apply Neural Network**. După deschiderea casetei de dialog **Network Processing** s-au deselectat 3 opțiuni (**Compute R squared, Include actuals in .OUT file**, rspectiv **Include in .OUT file actuals minus network outputs**), rămânând în final doar informația **Patterns Processed**.(**vezi fig.1.24**)
- Pas 3: se parcurg procedurile de *ataşare a fişierelor pentru vizualizarea evoluției datelor* (§.1.4.8), respectiv *examinarea datelor* (§.1.4.9).

În scopul analizei comparative s-au luat în considerare câteva date din rețeaua antrenata, pentru a vedea dacă se ajunge la un răspuns satisfăcător, atunci când în rețea se introduc noi date. În acest sens au fost selectate două date în mod aleator, şi s-a observat că eroarea este mică. Tabelul 1.3, fig. 1.27, fig. 1.28, fig.1.29.

Tabelul 1.3 Datele comparate

Nr. Crt.	inițial / nou	clasă cuantificare	garanția	Nr.interv.	DR	DRC	Pi	SCVP
1	valoarea rețelei inițiale	8.89	18	7	72	16	0.5	0.0079
	valoarea rețelei antrenată cu noile date	8.89	18	5	48	13	0.5	0.0095
2	valoarea rețelei inițiale	8.97	18	3	56	36	0.5	0.0012
	valoarea rețelei antrenată cu noile date	8.97	18	5	48	30	0.5	0.0017

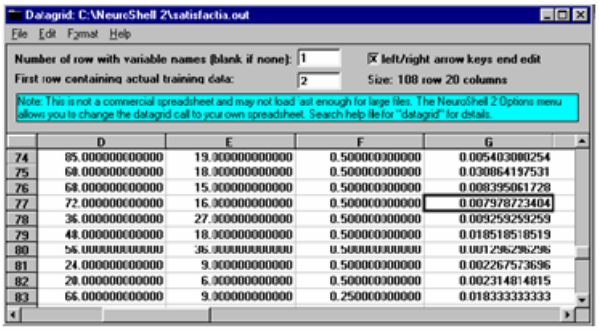


Fig. 1.26. Datele de ieşire iniţiale

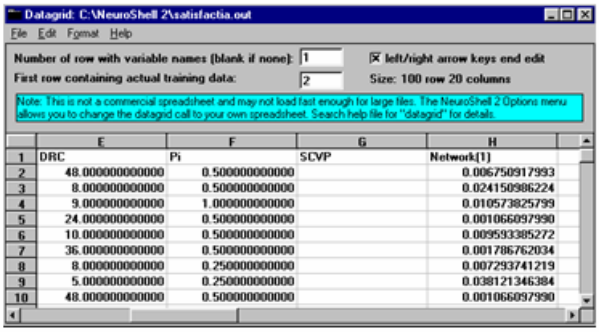


Fig. 1.27. Datele noi de ieşire

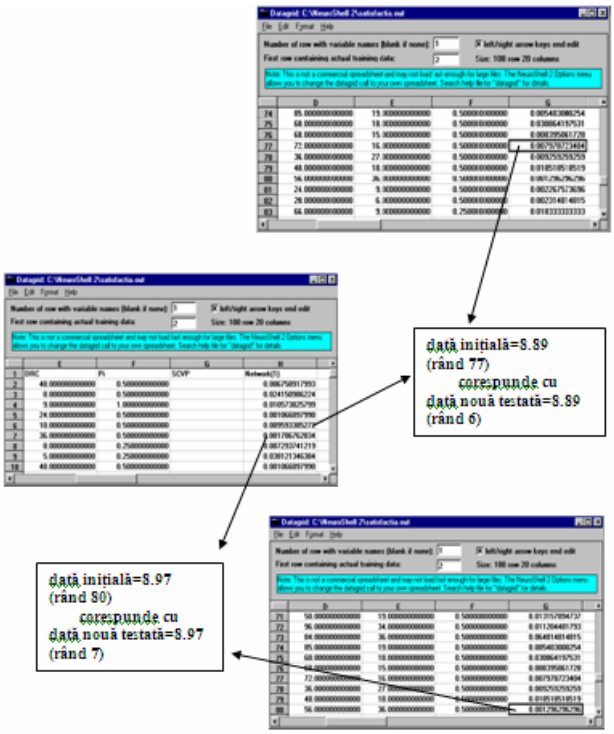


Fig. 1.29. Analiză comparativă între reţeaua iniţială şi reţeaua cu noile date

1.4.10. Moduri de îmbunătăţire a rezultatelor

Neuro Shell oferă posibilitatea îmbunătăţirii unei arhitecturi fară a-i schimba gradul de complexitate. Metodele cel mai des utilizate sunt prezentate în continuare:

1) *Utilizarea calibrării reţelei(calibration)*

Neuro Shell utilizează calibrarea pentru a optimiza reţeaua, prin aplicarea reţelei a unui set de teste independente în timpul antrenării. Setul de date de test se crează automat utilizând modulul **Test Set Extract.** Prin calibrare se „identifică" reţeaua optimă pentru date în setul de test, ceea ce înseamnă că reţeaua este capabilă să generalizeze cu succes noile date şi să returneze rezultate foarte bune.

Calibrarea realizează acest lucru prin calcularea mediei erorii pătratice pentru toate ieşirile actuale şi prezise. Pentru a apela facilitatea de calibrare trebuie setat câmpul **Calibration Test interval**, reprezentând frecvenţa evaluării setului de test. Cele mai bune predicţii se obţin dacă acest câmp se setează cu valori cuprinse între 50 şi 200. În cazul apelării la facilitatea de calibrare trebuie de asemenea validată căsuţa **Save network on the best test set.**

2) Ajustarea ratei de învăţare (learning rate), a inerţiei (momentum) şi a numărului de neuroni ascunşi pentru a-i crea reţelei un nou model de antrenare. Ajustările se realizează în modulul funcţional **learning.** Se poate de asemenea opta pentru activarea facilităţii **TurboPro**, care nu necesită setarea factorilor **learning rate** şi **momentum**. Aceată opţiune este inclusă în **Advanced System design** şi este valabilă pentru reţele de tip backpropagation.

3) Stabilirea unor variabile mai eficiente pentru predicţia pentru care este creată reţeaua.

Concluzii: Antrenarea Reţelie Neuronale Artificiale pentru identificarea satisfacţiei clienţilor în firma X reprezintă un real suport pentru rezolvarea problemelor legate de cuantificarea furnizorilor şi studierea

satisfacției clienților firmei, având în vedere dificultatea cuantificării aprecierilor clienților, a căror metalitate şi mod de acțiune diferă foarte mult. Rețeaua neuronală este capabilă să învețe singură, să dea răspunsuri pentru care un operator uman ar trebui să facă eforturi continue pentru a menține o bază informațională aşa de vastă.

Avantaje:

- În relația cu potențialii clienți se realizează o transparență mai mare asupra calității produselor pe care le va achziționa firma;
- Cel mai important aspect al realizării acestei rețele este cel al suportului informațional în procesul de negociere cu clienții, prin intermediul căruia se poate obține prețul optim pentru calitatea produselor selectate;
- Prin stabilirea nivelului de calitate al produselor vândute, al service-ului şi al consultanței oferite în procesul negocierii cu clienții, având ca suport informațional rețeaua neuronală artificială, s-a reuşit stabilirea configurației optime pentru aplicațiile necesare diferitelor tipuri de utilizatori;
- S-a reuşit ca în situațiile în care rețeaua a semnalat un grad scăzut a satisfacției clienților, firma să ia măsuri pentru a oferi servicii adiționale acestora.