STUDIUL UNOR ALGORITMI PENTRU SELECTAREA ALEATOARE DE VALORI SIMULATE PENTRU ÎNVĂȚAREA UNEI REȚELE NEURONALE

ENACHE Andrei George
Facultatea: IMST, Specializarea: Informatică aplicată în inginerie industrială, Anul de studii: Licență I,
e-mail: enache.andrei.george@gmail.com

Conducător ştiinţific: Prof. Dr. Ing. **Tom SAVU**

REZUMAT: O rețea neuronală reprezintă o formă de procesare a informației care este inspirată de sistemul nervos uman, urmând modelul creierului unei persoane. Cheia centrală a acestui model este chiar sistemul inovativ de procesare a informațiilor primite. Este compus dintr-un număr foarte mare de elemente de procesare interconectate (neuroni) care lucrează în același timp pentru a rezolva o anumită problemă. O rețea neuronală, ca și omul, învață din exemple, acest proces cognitiv necesitând ajustarea conexiunilor sinaptice dintre neuroni. De obicei, rețelele neuronale sunt configurate pentru o aplicație specifică. În această lucrare este prezentată învațarea tiparului pentru recunoașterea tipurilor diferitelor structuri atomice în urma învățării valorilor simulate alese cu ajutorul a diferiți algoritmi de selectare aleatoare.

CUVINTE CHEIE: Rețea neuronală, neuroni, recunoașterea modelului.

1. Introducere

În ultimii 10 ani, cele mai bune sisteme de inteligență artificială din punct de vedere al performanței (cum ar fi recunoașterea vocală pe diferite aparaturi sau traducătorul automat de la Google) au rezultat în urma unei tehnici numite “deep learning”.

“Deep learning” este de fapt un nou nume pentru o abordare a inteligenței artificiale, numită rețea neuronală. Aceste rețele au capacitatea de a găsi o logica utilizând date complicate sau imprecise, pot fi folosite pentru a extrage tipare si a detecta tendințe care sunt prea complexe pentru a putea fi recunoscute de oameni sau alte tehnici informatice. O rețea neuronală antrenată poate fi considerată un “expert” in categoria de informații care i-a fost oferită pentru analizare. Alte avantaje ale acestora includ:

- Învățarea adaptivă: Abilitatea de a învăța cum să facă anumite sarcini, bazându-se pe datele primite la antrenarea sau experiența inițială;
- Auto-Organizarea: O RN își poate crea propria organizare sau reprezentare a informației pe care o primește în timpul invățării;
- Operații în timp real: Calculele pot fi făcute in paralel și dispozitive hardware speciale sunt făcute pentru a profita de acest lucru;
- Toleranța defecțiunilor prin codificarea informațiilor redundante: Distrugerea parțială a unei rețele duce la degradarea corespunzătoare a performanței, dar anumite capabilități pot fi reobținute chiar și cu defecțiuni majore.

RN au alte modalități de rezolvare a problemelor față de calculatoarele obișnuite. În timp ce calculatoarele folosesc o abordare algoritmică (folosește un set de reguli, trebuind să le urmeze pentru a reusi, asta restricționând capacitatea de rezolvare a problemelor pentru care avem deja o rezolvare prestabilita), dar inteligenta artificială procesează informatia într-un mod similar cu al creierului uman.
În această lucrare se urmărește crearea unei rețele pentru detectarea și afișarea tipului de particule din aer, având în vedere modul de constituție al acesteia, folosind elemente prestbilite în ordine aleatoare.

Elementele de noutate prezentate în lucrarea de cercetare prezentată sunt:

- Crearea și implementarea unor diferiți algoritmi de sortare a unor valori prestabilite
- Extragereaa unei anumite cantitști din aceste valori, păstrând un nivel de proporționalitate ale elementelor diferite cât mai mare, comparativ cu setul inițial;
- Învățarea unei rețele neuronale folosind aceste seturi de valori, urmând a fi testate pe diferite cazuri reale, încercand să se ajungă la o precizie de constatare cât mai mare, acestea având o flexibilitate mult mai mare decât formulele statistice obișnuite.

2. Stadiul actual

Încă de la primele rețele neuronale în anul 1944, acestea au constituit un element de cercetare care a fost supus mai mult sau mai puțin diverselor analize. Cele mai active perioade de cercetare în acest domeniu au fost la implementarea ideii unei astfel de creații a cercetătorilor McCulloch si Pitts, urmând o perioadă de pauză când Minsky si Papert [1] au creat o lucrare în care se evidențiau limitările unui perceptron simplu (algoritm de învățare supravegheată folosind clasificatori binari), micșorând entuziasmul majorității cercetătorilor, mai ales a celor din domeniul informatic. Această pauză a dezvoltării a durat aproximativ douăzeci de ani, dar la începutul anilor 1980, RN au căpătat din nou atenție. Pe parcursul evoluției acestora s-au constatat anumite caracteristici speciale[6]:

- Pot generaliza: După procesarea datelor primite, RN pot de obicei deduce corect părtile ascunse ale unor mulțimi chiar dacă datele de intrare nu au continut doar informații precise, recurgând la modele anterioare [2];
- Pot aproxima orice funcție continuă cu orice număr de zecimale [3],[4],[5] (având mai multe funcții generale și flexibile decât metodele statistice tradiționale). Fiecare model predictibil asumă că exista o relație logică ascunsa între datele de intrare si de ieșire.

De la prima apariție a acestora, utilizarea rețelor pentru anticiparea anumitor fenomene a evoluat, în special dupș introducerea "algoritmului de învățare profundș" ("backpropagation algorithm").

Aplicațiile rețelelor neuronale au atins foarte multe arii de acoperire, cum ar fi sfera financiară (prezicerea falimentului și eșuării firmelor, prețurile acțiunilor), dar și prognoza consumului de electricitate (folosind ca date de intrare căldura) și învatarea mașinăriilor cu diverse procedee în cadrul inteligenței artificiale.

3. Enunțul problemei

Obiectivul cercetării a fost acela de a găsi un algoritm optim pentru selectarea aleatoare a unui număr prestabilit de valori simulate în scopul învățării unei rețele neuronale.

Setul de date inițial este grupat sub forma unui vector alcătuit din 359 000 de linii și 6 coloane. Pe prima coloană a acestuia se regăsesc 6 valori distincte în proporții variate, urmând ca la finalul întregului proces, matricea finală să conțină 70% din numărul total de valori, proporția celor distincte rămânand cât mai apropiată de cea inițială.

4. Descrierea algoritmilor testați

Cei patru algoritmi folosiți în crearea acestei lucrări au majoritatea componentelor identice, diferența fiind făcută de modul în care sunt alese elementele în ordine aleatoare.

Părțile utilizate, constante ca mod de lucru în toate cele patru programe sunt reprezentate de:

- Citirea din fișier a datelor de intrare (Fig. 1);
- Evidențierea diferitelor elemente de pe prima coloană prin extragerea unei valori din fiecare tip și scrierea acestora sub forma unei matrice (Fig. 2);

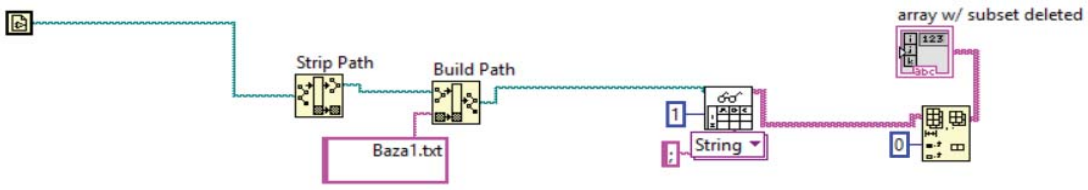

Fig. 1 Citirea din fișier a datelor de intrare

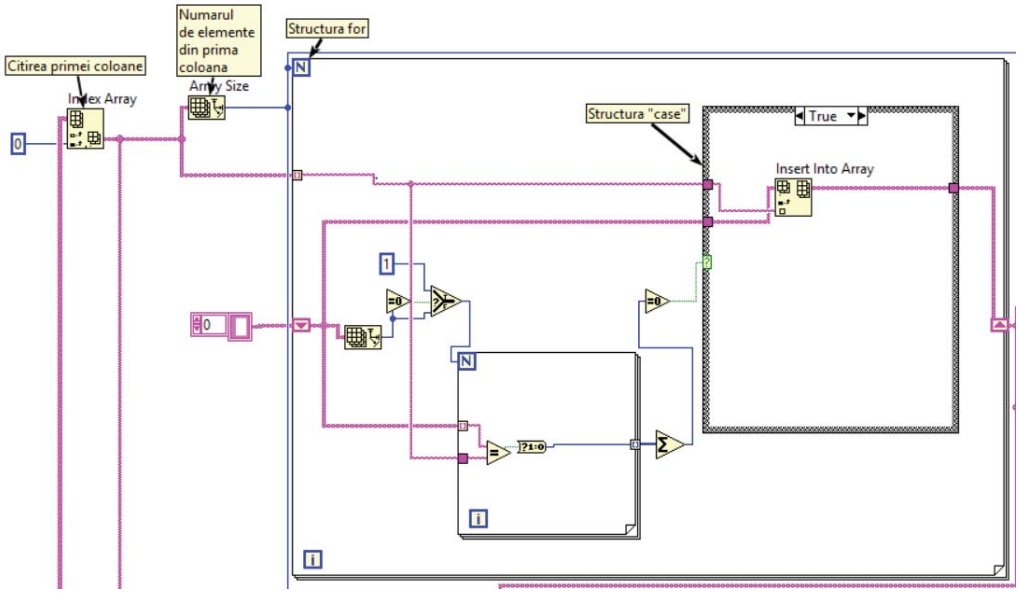

Fig. 2 Evidențierea elementelor diferite de pe prima coloană

- Utilizarea structurilor “for” (repetarea de un număr N a unei execuții) și “case”("în caz că", permite formarea a doua tipuri de execuție diferită, în funcție de valoarea primită (Adevarat sau Fals), pentru alegerea a cca. 70% din valorile ordonate aleator, in funcție de prima coloană (procentul variind în funcție de valorile introduse si de numărul acestora);

- Elidarea rândurilor goale cauzate de alegerea procentului de valori prin căutarea termenilori nuli de pe prima coloană și ștergerea acestora pentru a calcula acuratețea fiecărui algoritm unic în scopul alegerii celui optim, programul este introdus într-o buclă “for” având 1000 de repetări, calculându-se suma pătratelor abaterilor după fiecare iteratie, astfel putând fi observată precizia programului, dar și media de timp pentru fiecare rulare (Fig. 3).

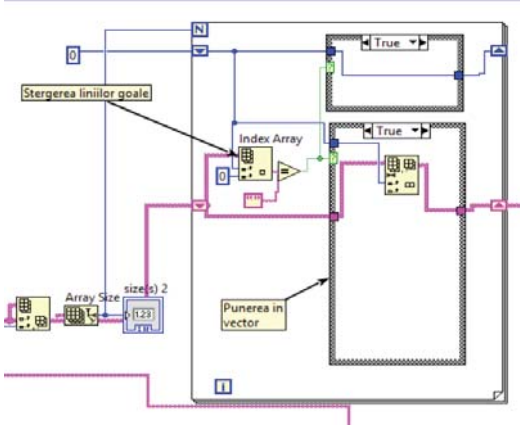

Fig. 3 Elidarea rândurilor goale

Modalitățile de autoaranjare concepute sunt executate în modul următor:

1) Numărul total de rânduri N este înmulțit cu o valoare aleatoare generată în intervalul [0,1) , aceasta reprezentând poziția de pe care va fi stearșa o linie a matricei inițiale. Repetând încă o dată procesul de înmulțire a unei valori situate în intervalul [0,1) cu numărul inițial de rânduri, se formează un număr care va reprezenta poziția unde linia ștearsă anterior va fi adăugată în matricea inițială. Acest proces se repetă de N ori pentru crearea unui vector de valori cât mai variate (Fig. 4) .

2) Crearea unui șir de numere naturale crescătoare, având valoarea maximă egală cu mărimea primei coloane a matricei în care se află datele de intrare si rearanjarea acestui șir în ordine aleatoare folosind funcția “riffle” (rearanjarea aleatoare a unui vector de numere) . Pentru crearea unei noi matrice care să respecte ipoteza problemei, fiecare dintre numerele existente în șirul anterior va constitui, pe rând, poziția liniei ce va fi extrasă din matricea inițiaăa și transmisa celei finale (Fig. 5) ;

3) Se creează o structură “for” având N repetări, introducându-se în aceasta o funcție “cluster” ce poate construi un grup de 2 sau mai multe elemente. Prin intermediul acestei functii se formează un mănunchi (cluster) între un număr aleator situat în interval [0,1) și numărul iterației curente, formându-se astfel un vector de N elemente a câte 2 valori fiecare. Cu o funcție “Sort 1D array” (sortarea unui vector cu o singura coloană în ordinea crescătoarea a valorilor conținute) , acest vector se ordonează crescător în funcție de valoarea primului termen, al celui aflat între 0 si 1. Extrăgând pe rând fiecare valoare aflată pe poziția a doua, se formează un vector constituit din numere naturale diferite, având valoare maximă numărul N. Aceste numere vor reprezenta numărul de ordine al liniei extrase din vectorul de valori prestabilit, urmând să fie pus în matricea finală (Fig. 6) ;

4) Amplificarea numărului curent de valori prezente în prima coloană cu un număr din intervalul [0,1) generat aleator, rezultatul urmând a fi rotunjit la cea mai apropiată valoare naturală, extragerea liniei aflată pe poziția cu acest număr si punerea acestuia într-o nouă matrice care va fi utilizată ulterior. După ce o linie este extrasă, aceasta este ștearsă din matricea în care se afla inițial (Fig. 7) ;

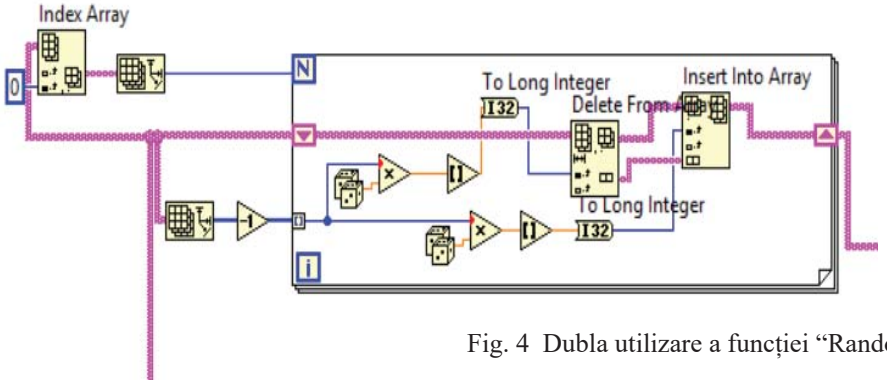

Fig. 4 Dubla utilizare a funcției “Random number (0-1)”

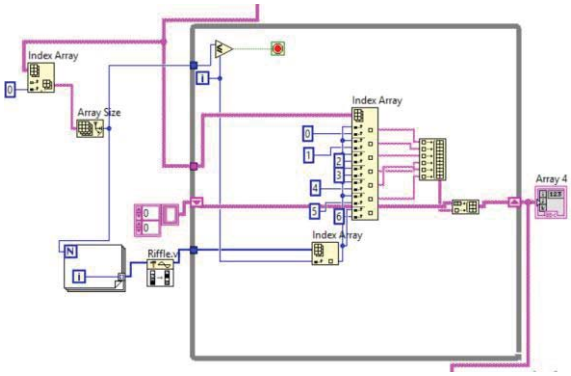

Fig. 5 Utilizarea funcției “Riffle”

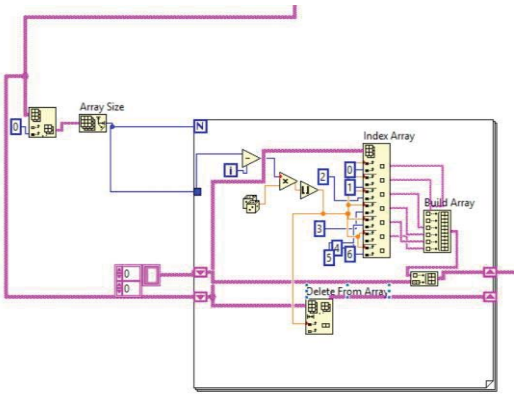

Fig. 6 Ștergerea termenului mai apropiat

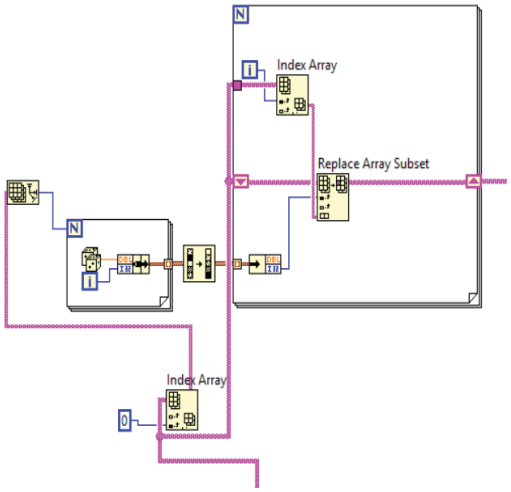

Fig. 7 Utilizarea funcției “Bundle”

5. Testarea algoritmilor

Pentru a vedea care dintre cei patru algoritmi este cel mai bine optimizat pentru scopul dorit, aceștia se testeaza pe rând, utilizând același set de valori de intrare și se calculează timpul mediu necesar unei rulări complete, adunând la fiecare repetare timpul necesar rulării cu cele precedente, suma rezultată fiind împărțită la numărul total de iterații pentru a afla valoarea medie (Fig. 8).

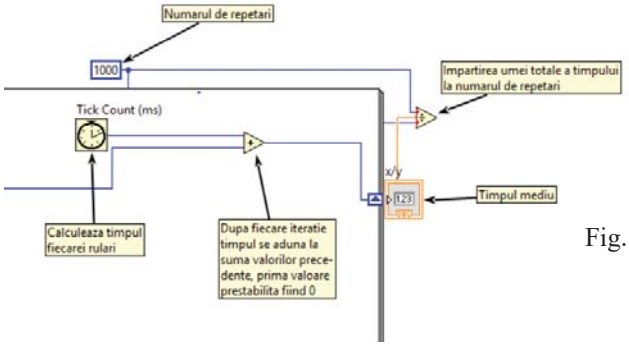

Fig. 8 Calculul mediu necesar unei rulări complete

A doua modalitate de comparație a rezultatelor este folosind media sumelor abaterilor, comparând rezultatele. În urma comparării acestor rezultate se poate constata care algoritm este cel mai bun din punct de vedere al performanțelor.

Tabelul 1. Calculul valorilor necesare aflării algoritmului optim

Termenul	A1	A2	A3	A4
%T1	20.892	20.892	20.892	20.892
%T2	16.3788	16.3788	16.3788	16.3788
%T3	22.9686	22.9686	22.9686	22.9686
%T4	21.4066	21.4066	21.4066	21.4066
%T5	3.9396	3.9396	3.9396	3.9396
%T6	14.414	14.414	14.414	14.414
%T1-%T1(f)	-0.0064	-0.014	0.001	-0.0026
%T2-%T2(f)	0.0028	0.0027	-0.0021	0.0032

%T3-%T3(f)	-0.0004	0.049	0.0045	-0.0011
%T4-%T4(f)	-0.0014	0.005	0.002	-0.0028
%T5-%T5(f)	0.0086	0.001	-0.0013	0.0065
%T6-%T6(f)	0.004	-0.0044	0.0032	0.003
Media diferentelor	0.0012	0.0065	0.00126	0.00103
Media totala		0.00249		
Diferenta fata de media totala	-0.00129	0.00401	-0.00123	-0.00146
Timpi rulare	148046558 ms	135616257 ms	155387279 ms	152015284

6. Concluzii si dezvoltări ulterioare

În urma obesrvațiilor putem constata că valoarea obținută pentru timpul mediu de rulare, dar și abaterile medii ale probabilității aparițiilor pot fi considerate suficient de reduse astfel încât algoritmul 3 să poată fi implementat ulterior în sistemul ce va fi dezvoltat.

În etapa a doua a lucrării, se dorește utilizarea programului ales la cazul anterior pentru învățarea unei rețele neuronale, pentru a diferenția și detecta tipurile diferite de particule atmosferice. Pentru aceasta, pe lângă acest algoritm ce va alege 70% dintre valorile inițiale și care va constitui proba rețelei, se va folosi și restul de 30% al eșantionului, acesta fiind proba de validare pentru testarea acesteia, urmând să se creeze un set de valori eronate pentru partea de erori, crescând acuratețea rezultatelor.

Utilitatea acestui program, în stadiul lui final, va reprezenta un real ajutor pentru instituțiile meteorologice bazate pe cercetarea concentrației si tipurilor particulelor din atmosferă, putând astfel să afle informații în timp real despre toate schimbările din aer.

7. Bibliografie

[1]. Marvin Minsky, Seymourr A. Papert (1987), *Perceptrons: An Introduction to Computational Geometry, Expanded Edition* Editura The MIT Press, ISBN 978-0262631112

[2]. Anil K. Jain, Jianchang Mao, K.M. Miohiuddin (1996), "*Artificial neural networks: a tutorial*", Computer, Volum: 29, Issue: 3

[3]. Hornik et al (1989), "*Multilayer feedforward networks are universal approximators*", volumul 2, issue:359-366

[4]. Cybenko(1989), "*Approximation by Superpositions of a Sigmoidal Function*", Math. Control Signal Systems 2: 303 - 314

[5]. Funahashi (1993), "*Prefrontal neuronal activity in rhesus monkeys performing a delayed anti-saccade task*" , Nature 365 753-756

[6]. Zhang G., Patuwo B., Hu M. (1997), Forecasting with artificial neural networks: The state of the art in International journal of Forecasting 14 (1998) 35-62

8. Notații

%T(1...6)= procentul inițial al termenilor unici 1-6 de pe prima coloană
%T(1...6)(f)=procentul final al termenilor 1-6
%T(1...6)-%T(1...6)(f)= diferența dintre procentul inițial și final al fiecărui termen
RN=Rețea neuronală