Laborator 7

Toolbox-ul ***nnet*** conține funcții care facilitează antrenarea rețelelor neuronale, reprezentate în Matlab ca obiecte de tip rețea (network). În acest laborator veți învăța cum să antrenați o rețea (prin specificarea unor parametri care definesc modul în care se realizează antrenarea) pe cazul particular în care rețeaua conține un singur perceptron. Vom denumi în cele ce urmează rețeaua cu ***net***.

În Matlab folosim funcția generică ***train*** pentru antrenarea rețelelor neuronale. Această funcție poate fi folosită pentru antrenarea tuturor tipurilor de rețele neuronale (rețele feedforward, rețele recurente) cu diverse arhitecturi (cu un singur nivel (layer) având unul sau mai mulți perceptroni, cu mai multe nivele). Vom folosi de acum înainte această funcție pentru antrenarea oricărei rețele la laborator.

Materialul pentru acest laborator poate fi consultat și în limba engleză aici: https://www.mathworks.com/help/pdf_doc/nnet/nnet_ug.pdf (întreg pdf-ul conține manualul pentru folosirea toolbox-ului Neural Networks), capitolul 11 (de la pagina 476 la pagina 488 corespunzător intervalului 11-6 până la 11-18).

În laboratorul precedent, funcția ***algoritmRosenblattOnline.m*** implementa algoritmul de învățare/antrenare al lui Rosenblatt (varianta online) pentru un perceptron. În cazul unei mulțimi de antrenare liniar separabile, algoritmul de învățare a lui Rosenblatt conduce într-un număr finit de pași la parametri $(\mathbf{w}^*, b^*)$ care definesc hiperplanul de separare ale celor două clase $\{-1, +1\}$.

Funcția ***train*** învață parametri unei rețele ***net*** folosind algoritmul de învățare specificat de ***net.trainFcn***. În varianta de învățare online (incrementală), vectorul de ponderi **w** și biasul b se modifică (se updatează) de fiecare dată când un exemplu de antrenare este misclasificat. Exemplele de antrenare pot fi prezentate în ordine ciclică (în aceeași ordine pentru fiecare epocă – opțiunea *'trainc'*, setată implicit) sau în ordine aleatoare (în altă ordine pentru fiecare epocă – opțiunea *'trainr'*). În varianta offline (sau batch – opțiunea *'trainb'*) vectorul de ponderi **w** și biasul b se modifică pe baza tuturor exemplelor misclasificate. În acest caz nu contează ordinea exemplelor.

Regulile pentru modificarea (updatarea) vectorului de ponderi **w** și a biasului b trebuiesc specificate. Regulile de update în cazul unei rețele ***net*** cu un perceptron se specifică astfel: (i) pentru vectorul de ponderi **w** se setează net.inputWeights{1}.learnFcn (valoarea implicită este *'learnp'* – implementează regula de update din algoritmul lui Rosenblatt); (ii) pentru biasul b se setează net.biases{1}.learnFcn (valoarea implicită este *'learnp'* – implementează regula de update din algorimtul lui Rosenblatt). O altă regulă de update este dată de valoarea *'learnpn'*, ce specifică că update-ul se face în funcție de exemplele de antrenare normalizate (de lungime 1).

Alți parametri ai antrenării sunt specificați în ***net.trainParam***. Însemnătatea fiecărui parametru o puteți afla consultând help-ul din Matlab.

Pentru punctele *a* și *b* din laboratorul trecut, codul Matlab care realizează antrenarea online a perceptronului (rețea cu un neuron) este următorul:

```
clear all, close all
%datele: exemplele + etichetele
X = [0 0 0 0.5 0.5 0.5 1 1;0 0.5 1 0 0.5 1 0 0.5];
T = [1 1 1 1 -1 -1 -1 -1];
%reprezentare grafica a datelor
figure(1), hold on;
eticheta1 = find(T==1);
etichetaMinus1 = find(T==-1);
plot(X(1,eticheta1),X(2,eticheta1),'or');
plot(X(1,etichetaMinus1),X(2,etichetaMinus1),'xg');
axis([-2 2 -2 2]);
%pune pauza 2 secunde
pause(2);
%creeaza perceptron
net = perceptron;
%modifica functia de transfer – din hardlim in hardlims
net.layers{1}.transferFcn = 'hardlims';
%configureaza perceptron pe baza datelor de intrare si iesire
net = configure(net,X,T);
view(net);
%afiseaza proprietatile perceptronului legate de vectorul de ponderi + bias
disp('Proprietati legate de vectorul de ponderi:');
disp(net.inputWeights{1});
disp('Proprietati legate de bias:');
disp(net.biases{1});
%initializeaza parametri retelei
net = init(net);
% implicit ponderile + bias sunt nule (de la initzero)
```

```
disp(net.IW{1}), disp(net.b{1})
%seteaza numarul de epoci pentru antrenare
net.trainParam.epochs = 100;
%antreneaza reteaua
[net,antrenare] = train(net,X,T);
figure(1);
%ploteaza dreapta de separare
plotpc(net.IW{1},net.b{1})
title('Reprezentarea grafica a exemplelor de antrenare si a dreptei de separare');
%simuleaza reteaua pentru datele de intrare
etichetePrezise = sim(net,X);
isequal(etichetePrezise,T)
```

Realizați următoarele:

1. verificați că parametri ($\mathbf{w}^*$,b^*) învățați folosind codul de mai sus (cu funcția ***train***) sunt similari cu cei învățați pe baza funcției algoritmRosenblattOnline.m implementată laboratorul trecut.
2. inițializați cu valori aleatoare vectorul de ponderi (net.IW{1}) și bias-ul (net.b{1}) modificând valoarea câmpului *'initFcn'* (pentru net.inputWeights{1} și net.biases{1} din *'initZero'* în *'rands'*). Antrenați rețeaua în acest caz. Cum se modifică numărul de epoci necesare convergenței antrenării?
3. modificați codul Matlab de mai sus astfel încât să plotați după fiecare iterație hiperplanul de separare.
4. adăugați punctul (-50,90) cu eticheta 1 mulțimii de antrenare. Cum se modifică timpul de convergență (=numărul de epoci) al algoritmului dacă porniți de la parametri (w, b) inițializați aleator? (păstrați aceeași inițilizare a parametrilor pentru o rețea antrenată pe mulțimea de antrenare inițială și pentru rețea antrenată pe mulțimea augmentată)
5. modificați funcțiile de învățare ale ponderilor și ale bias-ului din *'learnp'* în *'learnpn'* (folosiți help-ul și citiți despre deosebirea dintre ele). Cum se modifică timpul de convergență (= numărul de epoci) al algoritmului?
6. rezolvați punctele *c* și *d* din laboratorul 6 folosind o rețea antrenată online.
7. definiți o rețea cu un perceptron care să învețe funcțiile logice AND și OR. Reprezentați grafic punctele mulțimii de antrenare și dreapta de separare.
8. rulați punctele anterioare cu antrenarea de tip offline.