گزارش تمرین شبکه عصبی سوال 1

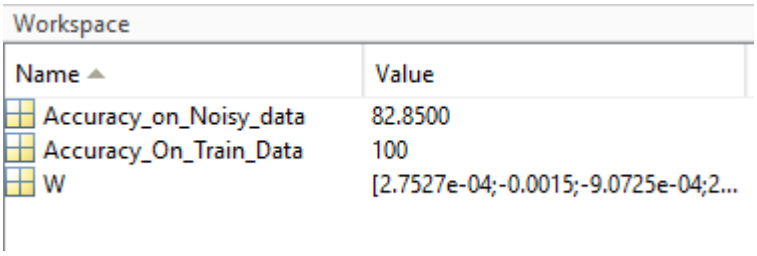
تمامی دستورات دارای کامنت میباشند.

ابتدا هر دو ورودی را به صورت ماتریس برای شبکه تعریف میکنیم و خروجی مطلوب متناظر با آنها را نیز تعریف میکنیم . سپس ماتریس وزن را با مقادیر تصادفی بسیار کوچک تولید میکنیم(هم دارای درایه های مثبت و هم دارای درایه های منفی) . سپس شبکه را طبق قانون یادگیری پرسپترون تا جایی که خطا برابر صفر شود آموزش میدهیم(به علت تابع فعالساز سخت ، خطا میتواند دقیقا صفر شود) . اینگونه ماتریس وزن ها به طور کامل پس از چند ایپاک با دقت 100 درصد بر روی دادگان آموزش بدست می آید.

همچنین تابعی جهت محاسبه دقت بر روی دادگان آموزش نوشته شده است(Accuracy_Fcn)

سپس شروع به تست شبکه بر روی دادگان نویزی میکنیم. به طور مثال ، یکی از ورودی ها را به طور تصادفی not میکنیم و دقت را برای هزار داده نویزی که هر بار I و L نویزی میشوند حساب میکنیم.

دقت بر روی دادگان نویزی متغیر است ولی دقتی به طور متوسط حدود 80 درصد دارد.



Editor - B:\ML_HW_2\Part 1\Train_Code.m

Train_Code.m | Accuracy_Fcn.m

```
%% Start

close all ; clear ; clc ; % Clear Everything

%% Initial Patameters

Etha = 0.001 ; % Learning Rate

%% Inputs

N = 2 ; % all possible choose of inputs

I = [ 0 1 0
      0 1 0
      0 1 0 ] ; % I input

I_Num = 0 ; % Corresponding Output for input I

L = [ 1 0 0
      1 0 0
      1 1 1 ] ; % L input

L_Num = 1 ; % Corresponding Output for input L

[R , C] = size(I) ; % Size of Inputs

%% Initial Weights Generation

W = 0.001*2*(rand( R*C + 1 , 1 )-0.5) ; % Random weights Vector(10*1)

%% Learning Program

Accuracy_On_Train_Data = Accuracy_Fcn(W) ; % Accuracy on Random Weights

while Accuracy_On_Train_Data ~= 100 % Because the Activation Function is Hard, there is No need
                        % to define epochs and can make an auto-repeater
                        % until Zero Error

    for i = 0 : N-1 % Two kinds of Inputs
```

Editor - B:\ML_HW_2\Part 1\Train_Code.m

Train_Code.m | Accuracy_Fcn.m

```
40
41 -            switch i
42 -                case I_Num % I input
43 -                    X = [reshape(I',1,R*C) , 1] ; % Reshape Train Date into Vector(1*10)
44 -                case L_Num % L input
45 -                    X = [reshape(L',1,R*C) , 1] ; % Reshape Train Date into Vector(1*10)
46 -            end
47
48 -            Y = X * W ; % Calculate the Output
49
50 -            if Y >= 0 % Passing through Activation Function
51 -               Y = 1 ;
52 -            else
53 -               Y = 0 ;
54 -            end
55
56 -            D = i ; % Desired Output
57 -            E = D - Y ; % Error Calculation
58 -            W = W + ( Etha .* X' * E ) ; % Weights Correction
59
60 -        end
61
62 -        Accuracy_On_Train_Data = Accuracy_Fcn(W) ; % Accuracy Calculation
63
64 -    end
65
66      %% Test on Noisy Data with One noisy input
67
68 -    Correct = 0 ;
69 -    Wrong = 0 ;
70
71 -    for i = 1 : 1000 % Test on 2*1000 Noisy Data
72
73          % Make Noise For I
74
75 -        random_index = randi([1,9],1,1) ; % Random Position to change
76 -        X = [reshape(I',1,R*C) , 1] ; % Reshape Noisy Date into Vector(1*10)
77 -        X(1,random_index) = not(X(1,random_index)) ; % Change a position into its not
78
79 -        Y = X * W ; % Calculate the Output
```

Editor - B:\ML_HW_2\Part 1\Train_Code.m

Train_Code.m | Accuracy_Fcn.m

```
79 -        Y = X * W ; % Calculate the Output
80 -        if Y >= 0   % Passing through Activation Function
81 -           Y = 1 ;
82 -        else
83 -           Y = 0 ;
84 -        end
85
86 -        if Y == 0 % Decide if it is True or Wrong
87 -           Correct = Correct + 1 ;
88 -        else
89 -           Wrong   = Wrong   + 1 ;
90 -        end
91
92          % Make Noise For L
93
94 -        random_index = randi([1,9],1,1) ; % Random Position to change
95 -        X = [reshape(L',1,R*C) , 1] ; % Reshape Noisy Date into Vector(1*10)
96 -        X(1,random_index) = not(X(1,random_index)) ; % Change a position into its not
97
98 -        Y = X * W ; % Calculate the Output
99 -        if Y >= 0   % Passing through Activation Function
100 -           Y = 1 ;
101 -       else
102 -          Y = 0 ;
103 -       end
104
105 -       if Y == 1 % Decide if it is True or not
106 -          Correct = Correct + 1 ;
107 -       else
108 -          Wrong   = Wrong   + 1 ;
109 -       end
110
111 -   end
112
113 -   Accuracy_on_Noisy_data = 100*(Correct)/(Correct + Wrong) ; % Accuracy on Noisy Data
114
115     %% Finish
116
117 -   clear C Correct D E Etha i I I_Num L L_Num N R random_index Wrong X Y
```

تابع Accuracy_Fcn

Editor - B:\ML_HW_2\Part 1\Accuracy_Fcn.m

Train_Code.m | Accuracy_Fcn.m

```
function [Accuracy] = Accuracy_Fcn(W)
%% Inputs

Correct = 0 ;
Wrong   = 0 ;

N = 2 ; % all possible kinds of input model

I = [ 0 1 0
      0 1 0
      0 1 0 ] ; % I input

I_Num = 0 ; % Corresponding Output for input I

L = [ 1 0 0
      1 0 0
      1 1 1 ] ; % L input

L_Num = 1 ; % Corresponding Output for input L

[R , C] = size(I) ; % Size of Inputs

%% Testing Program

for i = 0 : N-1 % Test on two choose for inputs

    switch i
        case I_Num
            X = [reshape(I',1,R*C) , 1] ; % % Reshape Train Date into Vector(1*10)
        case L_Num
            X = [reshape(L',1,R*C) , 1] ; % % Reshape Train Date into Vector(1*10)
    end

    Y = X * W ; % Calculate the Output

    if Y >= 0 % Passing through Activation Function
       Y = 1 ;
    else
       Y = 0 ;
    end

    if Y == i % Decide if it is True or not
        Correct = Correct + 1 ;
    else
        Wrong   = Wrong   + 1 ;
    end

end

Accuracy = 100*(Correct)/(Correct + Wrong) ; % Accuracy on Test Data

end
```