AIM: Write a program in Python to implement Hopfield neural network

SOURCE CODE:

importnumpy as np

x=np.array([[1,1,1,1,1],[1,-1,-1,1,-1],[-1,1,-1,-1,-1]])

x1=np.transpose(x)

t1=np.array([[1,1,1,-1,1]])

t2=np.array([[1,-1,-1,-1,-1]])

t3=np.array([[1,1,-1,-1,-1]])

w=np.zeros((5,5))

i=0

j=0

k=0

for i in range(len(x1)):

for j in range(len(x[0])):

for k in range(len(x)):

w[i][j] += x1[i][k] \* x[k][j]

print('Weight Matrix:\n')

for r in w:

print(r)

print('\n\nWeight Matrix with no self connection:\n')

i=0

j=0

for i in range(int(5)):

for j in range(int(5)):

if(i==j):

w[i][j]=0

for r in w:

print(r)

E1=0

E2=0

E3=0

x11= x[0].reshape(5,1)

x12=x[1].reshape(5,1)

x13=x[2].reshape(5,1)

E1= -0.5 \* np.matmul(x[0],np.matmul(w,x11))

print('\n\nEnergy Calculations for pattern [1,1,1,1,1]:',E1)

E2= -0.5 \* np.matmul(x[1],np.matmul(w,x12))

print('\n\nEnergy Calculations for pattern [1,-1,-1,1,-1]:',E2)

E3= -0.5 \* np.matmul(x[2],np.matmul(w,x13))

print('\n\nEnergy Calculations for pattern [-1,1,-1,1,-1]:',E3)

print('\n\nTESTING PHASE')

w\_dash=np.transpose(w)

Yin1=t1[0][3]+ np.matmul(x[0],w\_dash[3])

if(Yin1>0):

t1[0][3]=1

else:

t1[0][3]=-1

if((t1==x).any()):

print('\nPattern [1,1,1,-1,1] Recognized ')

else:

print('\nPattern [1,1,1,-1,1] not Recognized ')

Yin2=t2[0][3]+ np.matmul(x[1],w\_dash[3])

if(Yin2>0):

t2[0][3]=1

else:

t2[0][3]=-1

if((t2==x).any()):

print('\nPattern [1,-1,-1,-1,-1] Recognized ')

else:

print('\nPattern [1,-1,-1,-1,-1] not Recognized ')

Yin3=t3[0][0]+ np.matmul(x[2],w\_dash[0])

if(Yin3>0):

t3[0][0]=1

else:

t3[0][0]=-1

if((t3==x).any()):

print('\nPattern [1,1,-1,-1,-1] Recognized ')

else:

print('\nPattern [1,1,-1,-1,-1] not Recognized ')

OUTPUT:

