Prac 12

import numpy as np

class HopfieldNetwork:

def \_\_init\_\_(self,size):

self.size=size

self.weights=np.zeros((size,size))

def train(self,patterns):

for p in patterns:

p=np.array(p)

self.weights+=np.outer(p,p)

np.fill\_diagonal(self.weights,0)

self.weights/=len(patterns)

def recall(self,pattern,steps):

pattern=np.array(pattern)

for \_ in range(steps):

for i in range(self.size):

raw=np.dot(self.weights[i],pattern)

pattern[i]=1 if raw>=0 else -1

return pattern

if \_\_name\_\_=="\_\_main\_\_":

patterns=[[1,-1,1,-1],

[-1,-1,-1,1],

[1,1,-1,-1],

[-1,1,1,1]]

steps=4

hf=HopfieldNetwork(steps)

hf.train(patterns)

pattern=[1,1,1,-1]

recalled=hf.recall(pattern,steps)

print("Recalled pattern", recalled.tolist())