Assignment 12

# Output

A picture containing diagram

Description automatically generated

# Code

# DLApr1C.py CS5173/6073 cheng 2022  
# load saved LeNet5 model from DLMar25B.py  
# as the file DLMar25Bstate.pt  
# gradient ascent on input for the 16 channels output by cnn2  
# Usage: python DLApr1C.py  
  
import torch  
import torchvision  
import torch.nn as nn  
import torch.nn.functional as F  
import numpy as np  
import matplotlib.pyplot as plt  
  
inputsize = 1  
hiddensize1 = 6  
hiddensize2 = 16  
hiddensize3 = 400  
hiddensize4 = 120  
hiddensize5 = 84  
outputsize = 10  
kernelsize = 5  
poolsize = 2  
  
class Model(nn.Module):  
 def \_\_init\_\_(self, inputsize, hiddensize1, hiddensize2, hiddensize3,   
 hiddensize4, hiddensize5, outputsize, kernelsize, poolsize):  
 super(Model, self).\_\_init\_\_()  
 self.cnn1 = nn.Conv2d(inputsize, hiddensize1, kernelsize, padding=2)  
 self.cnn2 = nn.Conv2d(hiddensize1, hiddensize2, kernelsize)  
 self.linear1 = nn.Linear(hiddensize3, hiddensize4)  
 self.linear2 = nn.Linear(hiddensize4, hiddensize5)  
 self.linear3 = nn.Linear(hiddensize5, outputsize)  
  
 def forward(self, x):  
 x = torch.sigmoid(self.cnn1(x))  
 x = F.avg\_pool2d(x, poolsize)  
 x = torch.sigmoid(self.cnn2(x))  
 x = F.avg\_pool2d(x, poolsize)  
 return x  
  
LeNet5 = Model(inputsize, hiddensize1, hiddensize2, hiddensize3,   
 hiddensize4, hiddensize5, outputsize, kernelsize, poolsize)  
  
LeNet5.load\_state\_dict(torch.load("DLMar25Bstate.pt"))  
  
plt.figure(figsize=(10, 10))  
for j in range(16):  
 s = torch.zeros((1, 1, 84, 84), dtype=torch.float32, requires\_grad=True)  
 for i in range(100):  
 s.retain\_grad()  
 LeNet5.zero\_grad()  
 p = torch.sum(LeNet5(s)[0][j])  
 p.backward(retain\_graph=True)  
 s = s + 0.1 \* s.grad # gradient ascent  
 s[s < 0] = 0  
 s[s > 255] = 255.0  
  
 plt.subplot(4, 4, j + 1)  
 plt.imshow(s.detach().numpy()[0][0], cmap='gray')  
 plt.xticks([])  
 plt.yticks([])  
plt.show()