dl-lab-assignment-3

December 13, 2023

Q1.Create tensor using pytorch and do basic functions on them.

[1]:
```
import torch
```

[8]:
```
z0=torch.zeros(7,7)
print(z0)
print(z0.dtype)
```

```
tensor([[0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.]])
torch.float32
```

[9]:
```
z1=torch.ones(7,7)
print(z1)
```

```
tensor([[1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.]])
```

[10]:
```
z1_int=torch.ones((7,7),dtype=torch.int16)
print(z1_int)
```

```
tensor([[1, 1, 1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1, 1, 1]], dtype=torch.int16)
```

[11]:
```
zr=torch.randint(10,(7,7))
print(zr)
```

```
tensor([[3, 3, 5, 9, 5, 9, 3],
        [2, 6, 6, 1, 1, 3, 1],
        [9, 5, 9, 2, 4, 1, 5],
        [7, 0, 7, 6, 5, 1, 9],
        [7, 5, 1, 1, 7, 4, 9],
        [7, 6, 6, 9, 4, 7, 9],
        [5, 1, 6, 1, 7, 4, 7]])
```

[12]:
```
zr1=torch.rand(7,7)
print(zr1)
```

```
tensor([[0.7583, 0.5850, 0.1650, 0.9816, 0.4828, 0.3852, 0.3278],
        [0.3647, 0.1675, 0.3154, 0.3839, 0.6774, 0.5821, 0.3673],
        [0.6737, 0.7830, 0.3551, 0.8629, 0.7017, 0.0444, 0.2416],
        [0.1052, 0.4512, 0.3675, 0.4032, 0.6829, 0.7504, 0.5412],
        [0.9762, 0.8610, 0.0015, 0.0591, 0.8063, 0.9350, 0.2667],
        [0.2659, 0.2807, 0.5368, 0.6223, 0.1263, 0.5683, 0.4982],
        [0.2490, 0.2566, 0.9340, 0.5286, 0.3366, 0.9112, 0.6110]])
```

[18]:
```
za=torch.arange(3,30,3)
print(za)
```

```
tensor([ 3,  6,  9, 12, 15, 18, 21, 24, 27])
```

[19]:
```
zlin=torch.linspace(3,30,3)
print(zlin)
```

```
tensor([ 3.0000, 16.5000, 30.0000])
```

[20]:
```
zfull=torch.full((5,5),7)
print(zfull)
```

```
tensor([[7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7]])
```

[21]:
```
zeye=torch.eye(5,5)
print(zeye)
```

```
tensor([[1., 0., 0., 0., 0.],
        [0., 1., 0., 0., 0.],
        [0., 0., 1., 0., 0.],
```

```
        [0., 0., 0., 1., 0.],
        [0., 0., 0., 0., 1.]])
```

[22]:
```
zfull1=torch.full((5,5),1)
print(zfull1)
```

```
tensor([[1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1],
        [1, 1, 1, 1, 1]])
```

[23]:
```
print(zfull+zfull1)
```

```
tensor([[8, 8, 8, 8, 8],
        [8, 8, 8, 8, 8],
        [8, 8, 8, 8, 8],
        [8, 8, 8, 8, 8],
        [8, 8, 8, 8, 8]])
```

[24]:
```
print(zfull-zfull1)
```

```
tensor([[6, 6, 6, 6, 6],
        [6, 6, 6, 6, 6],
        [6, 6, 6, 6, 6],
        [6, 6, 6, 6, 6],
        [6, 6, 6, 6, 6]])
```

[25]:
```
print(zfull*zfull1)
```

```
tensor([[7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7],
        [7, 7, 7, 7, 7]])
```

[26]:
```
#Mathematical operations with tensor pytorch
a=torch.rand(5,7)
print("Common functions:")
print(a)
print(torch.abs(a))
print(torch.ceil(a))
print(torch.floor(a))
```

```
Common functions:
tensor([[0.0087, 0.3460, 0.9342, 0.9680, 0.9659, 0.3873, 0.2922],
        [0.1773, 0.4570, 0.3258, 0.0512, 0.5357, 0.7104, 0.6362],
        [0.7752, 0.9597, 0.9950, 0.7419, 0.5268, 0.5297, 0.3159],
```

```
        [0.0520, 0.7996, 0.2981, 0.6827, 0.3295, 0.2892, 0.2430],
        [0.3779, 0.3568, 0.5629, 0.9521, 0.0170, 0.4755, 0.9997]])
tensor([[0.0087, 0.3460, 0.9342, 0.9680, 0.9659, 0.3873, 0.2922],
        [0.1773, 0.4570, 0.3258, 0.0512, 0.5357, 0.7104, 0.6362],
        [0.7752, 0.9597, 0.9950, 0.7419, 0.5268, 0.5297, 0.3159],
        [0.0520, 0.7996, 0.2981, 0.6827, 0.3295, 0.2892, 0.2430],
        [0.3779, 0.3568, 0.5629, 0.9521, 0.0170, 0.4755, 0.9997]])
tensor([[1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.],
        [1., 1., 1., 1., 1., 1., 1.]])
tensor([[0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0.]])
```

[29]:
```
#Statistical functions,
b=torch.rand(2,20)
print(b)
print(torch.mean(b))
print(torch.std(b))
```

```
tensor([[0.2227, 0.6844, 0.7848, 0.3174, 0.5633, 0.6758, 0.9779, 0.3482, 0.4636,
         0.2856, 0.8398, 0.9345, 0.4934, 0.5826, 0.6220, 0.9495, 0.6979, 0.6156,
         0.8414, 0.2578],
        [0.6040, 0.6067, 0.3358, 0.9932, 0.3689, 0.2505, 0.7930, 0.0264, 0.8203,
         0.1783, 0.4820, 0.0167, 0.0839, 0.9766, 0.1220, 0.8389, 0.1813, 0.5472,
         0.1843, 0.3475]])
tensor(0.5229)
tensor(0.2909)
```

[30]:
```
#Reduction functions
c=torch.tensor([1,1,1,2,1,2])
print("Unique elements in the tensor is:",torch.unique(c))
```

```
Unique elements in the tensor is: tensor([1, 2])
```

[33]:
```
#Operations with vectors/matrices in pytorch
m1=torch.tensor([1,2,3,4,5,6,7,8,9])
m1=torch.reshape(m1,(3,3))
m2=torch.tensor([1,1,1,1,1,1,1,1,1])
m2=torch.reshape(m2,(3,3))
print(m1)
print(m2)
print(torch.cross(m1,m2))
print(torch.matmul(m1,m2))
```

```
tensor([[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]])
tensor([[1, 1, 1],
        [1, 1, 1],
        [1, 1, 1]])
tensor([[-3, -3, -3],
        [ 6,  6,  6],
        [-3, -3, -3]])
tensor([[ 6,  6,  6],
        [15, 15, 15],
        [24, 24, 24]])
```

[34]:
```
m3=torch.rand(3,3)
print(torch.svd(m3))
```

```
torch.return_types.svd(
U=tensor([[-0.3195,  0.7505, -0.5785],
        [-0.6430, -0.6202, -0.4494],
        [-0.6960,  0.2284,  0.6807]]),
S=tensor([1.4216, 0.3245, 0.0494]),
V=tensor([[-0.4835,  0.3583,  0.7986],
        [-0.6559,  0.4558, -0.6017],
        [-0.5796, -0.8147,  0.0146]]))
```

[35]:
```
from PIL import Image
from torchvision import transforms
img=Image.open('Tiger.jpg')
```

[36]:
```
tensor1=transforms.ToTensor()
t1=tensor1(img)
print(torch.max(t1))
print(torch.min(t1))
```

```
tensor(1.)
tensor(0.)
```

Q2.Implement CNN for mnist dataset.

[38]:
```
from keras import datasets # boston_housing, cifar10, cifar100, fashion_mnist,␣
  ↪imdb, mnist, reuters
(x_train, y_train), (x_test, y_test) = datasets.mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

[39]:
```
x_train1=x_train
x_test1=x_test
y_train1=y_train
y_test1=y_test
```

[40]:
```
# data converted from integer to float
x_train = x_train / 255.0
x_test = x_test / 255.0

print(x_train.shape)
print(x_test.shape)
```

```
(60000, 28, 28)
(10000, 28, 28)
```

[58]:
```
# reshape to add color channel into data
x_train = x_train.reshape(-1,28,28,1)
x_test = x_test.reshape(-1,28,28,1)

print(x_train.shape)
print(x_test.shape)
```

```
(60000, 28, 28, 1)
(10000, 28, 28, 1)
```

[59]:
```
from keras import utils
y_train = utils.to_categorical(y_train)
y_test = utils.to_categorical(y_test)

print(y_train.shape)
print(y_test.shape)
```

```
(60000, 10, 2, 2)
(10000, 10, 2, 2)
```

[60]:
```
input_shape = (28,28,1) # img_rows, img_colums, color_channels
num_classes = y_train.shape[1] # digit = 0~9
```

[61]:
```
from keras.models import Sequential
from keras.layers import Conv2D,MaxPooling2D,Flatten,Dense,Dropout
```

[62]:
```
model=Sequential()
#Convolution layer 1
model.
 ↪add(Conv2D(32,kernel_size=(3,3),padding='same',activation='relu',input_shape=(28,28,1)))
#Pooling layer1
model.add(MaxPooling2D(pool_size=(2,2)))
```

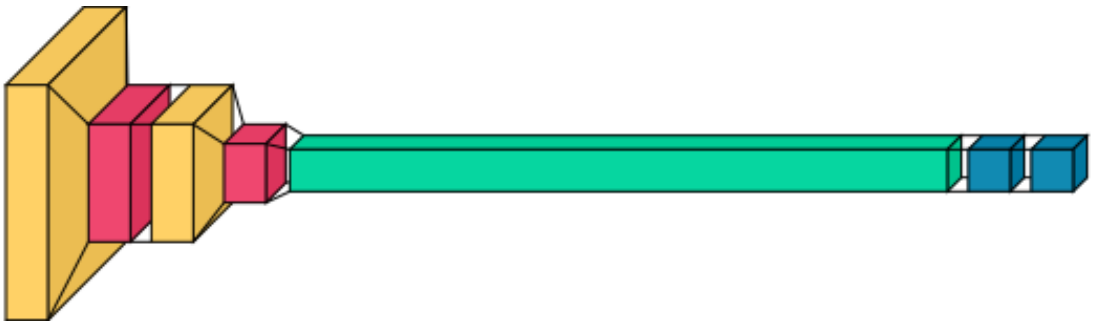
```
#Convolution layer2
model.add(Conv2D(64,kernel_size=(3,3),padding='same',activation='relu'))
#Pooling layer2
model.add(MaxPooling2D(pool_size=(2,2)))
#To reduce the dimension
model.add(Flatten())
#Dense layer
model.add(Dense(64,activation='relu'))
model.add(Dense(10,activation='softmax'))
```

[46]:
```
!pip install visualkeras
```

```
Collecting visualkeras
  Downloading visualkeras-0.0.2-py3-none-any.whl (12 kB)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from visualkeras) (9.4.0)
Requirement already satisfied: numpy>=1.18.1 in /usr/local/lib/python3.10/dist-
packages (from visualkeras) (1.23.5)
Collecting aggdraw>=1.3.11 (from visualkeras)
  Downloading
aggdraw-1.3.18-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (993
kB)
                           993.7/993.7
kB 11.2 MB/s eta 0:00:00
Installing collected packages: aggdraw, visualkeras
Successfully installed aggdraw-1.3.18 visualkeras-0.0.2
```

[63]:
```
import visualkeras
visualkeras.layered_view(model)
```

[63]:

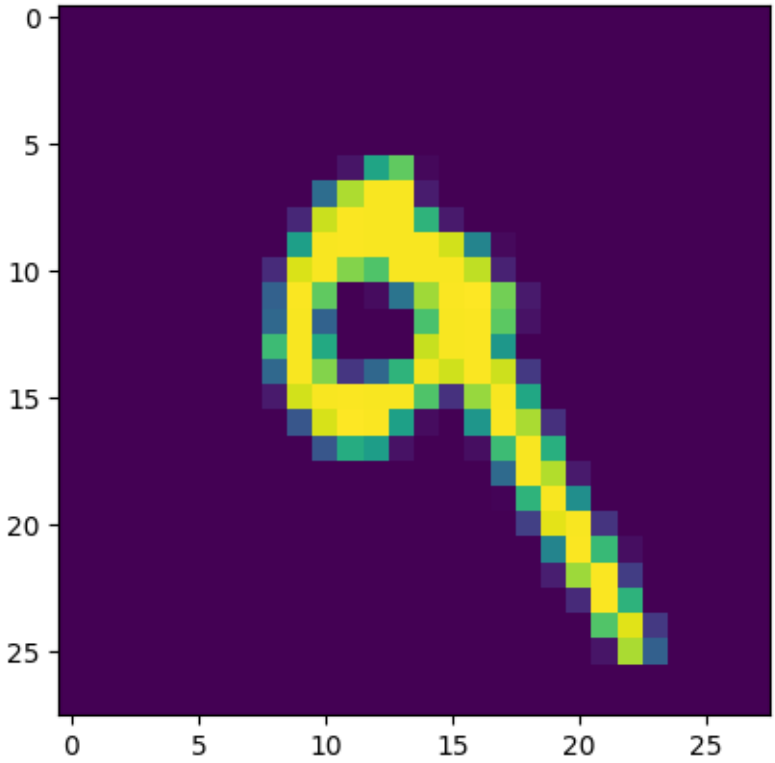


[48]:
```
model.
  ↪compile(loss='categorical_crossentropy',optimizer='adam',metrics='[accuracy]')
```

[69]:
```
import numpy as np
import matplotlib.pyplot as plt
preds = model.predict(x_test[7].reshape(-1,28,28,1))
print(int(np.argmax(preds)))
plt.imshow(x_test1[7])
```

```
1/1 [==============================] - 0s 84ms/step
0
```

[69]: <matplotlib.image.AxesImage at 0x78f784d57460>



Q3.Use the yolov8 and find the object detection for your favourite images.

[50]:
```
%pip install ultralytics
```

```
Collecting ultralytics
  Downloading ultralytics-8.0.227-py3-none-any.whl (660 kB)
                             660.5/660.5
kB 6.3 MB/s eta 0:00:00
Requirement already satisfied: matplotlib>=3.3.0 in
/usr/local/lib/python3.10/dist-packages (from ultralytics) (3.7.1)
Requirement already satisfied: numpy>=1.22.2 in /usr/local/lib/python3.10/dist-
```

```
packages (from ultralytics) (1.23.5)
Requirement already satisfied: opencv-python>=4.6.0 in
/usr/local/lib/python3.10/dist-packages (from ultralytics) (4.8.0.76)
Requirement already satisfied: pillow>=7.1.2 in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (9.4.0)
Requirement already satisfied: pyyaml>=5.3.1 in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (6.0.1)
Requirement already satisfied: requests>=2.23.0 in
/usr/local/lib/python3.10/dist-packages (from ultralytics) (2.31.0)
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (1.11.4)
Requirement already satisfied: torch>=1.8.0 in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (2.1.0+cu118)
Requirement already satisfied: torchvision>=0.9.0 in
/usr/local/lib/python3.10/dist-packages (from ultralytics) (0.16.0+cu118)
Requirement already satisfied: tqdm>=4.64.0 in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (4.66.1)
Requirement already satisfied: pandas>=1.1.4 in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (1.5.3)
Requirement already satisfied: seaborn>=0.11.0 in
/usr/local/lib/python3.10/dist-packages (from ultralytics) (0.12.2)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages
(from ultralytics) (5.9.5)
Requirement already satisfied: py-cpuinfo in /usr/local/lib/python3.10/dist-
packages (from ultralytics) (9.0.0)
Collecting thop>=0.1.1 (from ultralytics)
  Downloading thop-0.1.1.post2209072238-py3-none-any.whl (15 kB)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0->ultralytics)
(1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib>=3.3.0->ultralytics) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0->ultralytics)
(4.46.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0->ultralytics)
(1.4.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0->ultralytics)
(23.2)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0->ultralytics)
(3.1.1)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0->ultralytics)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
```

```
packages (from pandas>=1.1.4->ultralytics) (2023.3.post1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.23.0->ultralytics)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests>=2.23.0->ultralytics) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.23.0->ultralytics)
(2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.23.0->ultralytics)
(2023.11.17)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from torch>=1.8.0->ultralytics) (3.13.1)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.10/dist-packages (from torch>=1.8.0->ultralytics) (4.5.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages
(from torch>=1.8.0->ultralytics) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch>=1.8.0->ultralytics) (3.2.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch>=1.8.0->ultralytics) (3.1.2)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages
(from torch>=1.8.0->ultralytics) (2023.6.0)
Requirement already satisfied: triton==2.1.0 in /usr/local/lib/python3.10/dist-
packages (from torch>=1.8.0->ultralytics) (2.1.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.7->matplotlib>=3.3.0->ultralytics) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.8.0->ultralytics)
(2.1.3)
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-
packages (from sympy->torch>=1.8.0->ultralytics) (1.3.0)
Installing collected packages: thop, ultralytics
Successfully installed thop-0.1.1.post2209072238 ultralytics-8.0.227
```

[51]:
```
import ultralytics
ultralytics.checks()
```

```
Ultralytics YOLOv8.0.227  Python-3.10.12 torch-2.1.0+cu118 CPU (Intel Xeon
2.20GHz)
Setup complete  (2 CPUs, 12.7 GB RAM, 27.0/107.7 GB disk)
```

[52]:
```
!yolo predict model=yolov8n.pt source='Tiger.jpg'
```

```
Downloading
https://github.com/ultralytics/assets/releases/download/v0.0.0/yolov8n.pt to
'yolov8n.pt'…
```

```
100% 6.23M/6.23M [00:00<00:00, 58.4MB/s]
Ultralytics YOLOv8.0.227  Python-3.10.12 torch-2.1.0+cu118 CPU (Intel Xeon
2.20GHz)
YOLOv8n summary (fused): 168 layers, 3151904 parameters, 0 gradients, 8.7 GFLOPs

image 1/1 /content/Tiger.jpg: 384x640 1 dog, 201.2ms
Speed: 12.5ms preprocess, 201.2ms inference, 12.2ms postprocess per image at
shape (1, 3, 384, 640)
Results saved to runs/detect/predict
  Learn more at https://docs.ultralytics.com/modes/predict
```