# 基于pytorch的相关可视化工具 网络结构的可视化

import torch

import torch.nn as nn

import torchvision

import torchvision.utils as vutils

from torch.optim import SGD

import torch.utils.data as Data

import matplotlib.pyplot as plt

from torchvision.datasets import FashionMNIST

import torchvision.transforms as transforms

import numpy as np

train\_data = FashionMNIST(root="./deep Learning/",

train=True,

transform=transforms.ToTensor(),

download=True)

train\_loader = Data.DataLoader(dataset=train\_data,

batch\_size=128,

shuffle=True,

)

print(len(train\_loader))

# 469

test\_data = FashionMNIST(root='./deep Learning/',

train=False,

download=False)

print(len(test\_data))

# 10000

test\_data\_x = test\_data.data.type(torch.FloatTensor) / 255.0

test\_data\_x = torch.unsqueeze(test\_data\_x, dim=1)

test\_data\_y = test\_data.targets

print(test\_data\_x.shape)

# torch.Size([10000, 1, 28, 28])

print(test\_data\_y.shape)

# torch.Size([10000])

class ConvNet(nn.Module):

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

super(ConvNet, self).\_\_init\_\_()

self.conv1 = nn.Sequential(

nn.Conv2d(in\_channels=1, out\_channels=16, kernel\_size=5, stride=1, padding=1),

nn.ReLU(),

nn.AvgPool2d(kernel\_size=2, stride=2))

self.conv2 = nn.Sequential(nn.Conv2d(16, 32, 3, 1, 1),

nn.ReLU(),

nn.MaxPool2d(2, 2))

self.fc = nn.Sequential(nn.Linear(in\_features=32 \* 7 \* 7, out\_features=128), nn.ReLU(), nn.Linear(128, 64),

nn.ReLU())

self.out = nn.Linear(in\_features=64, out\_features=10)

def forward(self, x):

x = self.conv1(x)

x = self.conv2(x)

x = x.view(x.size(0), -1)

x = self.fc(x)

output = self.out(x)

return output

MyConvnet = ConvNet()

print(MyConvnet)

# ConvNet(

# (conv2): Sequential(

# (0): Conv2d(16, 32, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

# (1): ReLU()

# (2): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

# )

# (fc): Sequential(

# (0): Linear(in\_features=1568, out\_features=128, bias=True)

# (1): ReLU()

# (2): Linear(in\_features=128, out\_features=64, bias=True)

# (3): ReLU()

# )

# (out): Linear(in\_features=64, out\_features=10, bias=True)

# )

import hiddenlayer as hl

# hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1, 1, 32, 32]))

hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1, 1, 30, 30]))

hl\_graph.theme = hl.graph.THEMES['blue'].copy()

# hl\_graph.save("deep Learning/MyConvnet\_hl.png", format='png')

from torchviz import make\_dot

x = torch.randn(1, 1, 30, 30).requires\_grad\_(True)

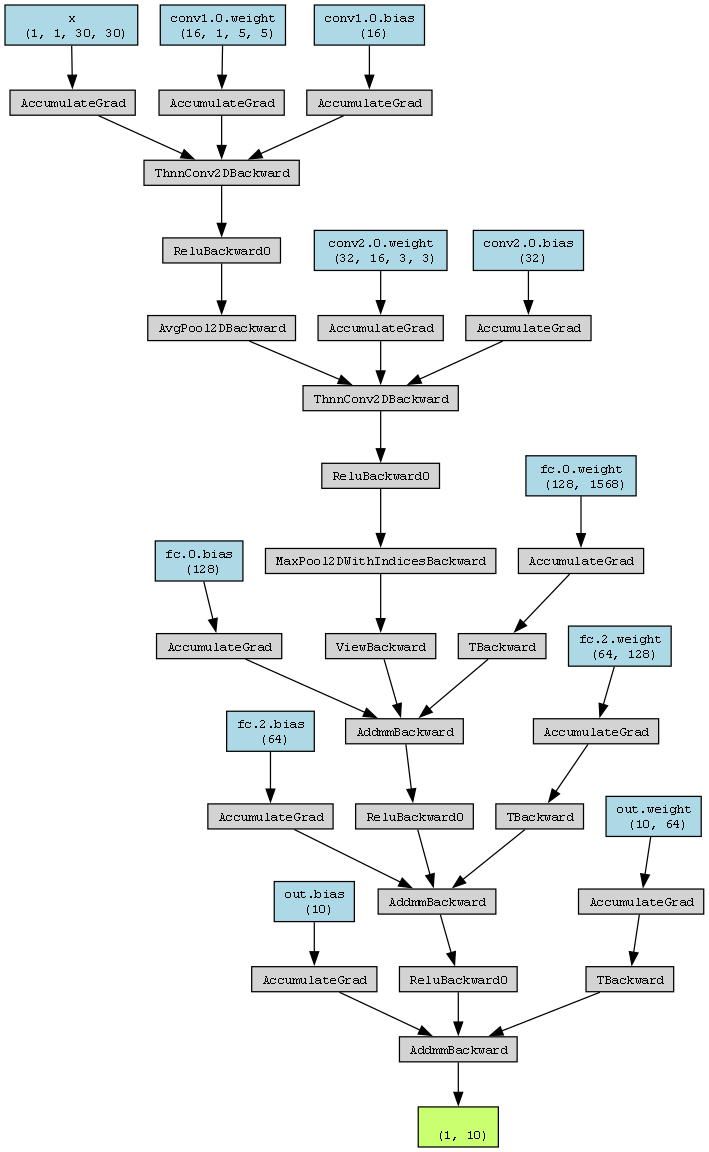
y = MyConvnet(x)

MyConvnetvis = make\_dot(y, params=dict(list(MyConvnet.named\_parameters()) + [('x', x)]))

MyConvnetvis.format = 'png'

MyConvnetvis.render("deep Learning/MyConvnet\_vis")

MyConvnetvis.view()



# 输入的张量维度有问题，修改

# 修改前 课本的 hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1, 1, 28, 28]))

# 修改后编译通过 hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1, 1, 32, 32]))

# Traceback (most recent call last):

# File "D:\pythoncode\learn\a\deep\_learning4.1.py", line 85, in <module>

# hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1,1,28,28]))

# File "D:\anaconda3\envs\portfolio\lib\site-packages\hiddenlayer\graph.py", line 143, in build\_graph

# import\_graph(g, model, args)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\hiddenlayer\pytorch\_builder.py", line 70, in import\_graph

# trace, out = torch.jit.\_get\_trace\_graph(model, args)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\jit\\_trace.py", line 1310, in \_get\_trace\_graph

# outs = ONNXTracedModule(

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1532, in \_wrapped\_call\_impl

# return self.\_call\_impl(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1541, in \_call\_impl

# return forward\_call(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\jit\\_trace.py", line 138, in forward

# graph, out = torch.\_C.\_create\_graph\_by\_tracing(

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\jit\\_trace.py", line 129, in wrapper

# outs.append(self.inner(\*trace\_inputs))

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1532, in \_wrapped\_call\_impl

# return self.\_call\_impl(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1541, in \_call\_impl

# return forward\_call(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1522, in \_slow\_forward

# result = self.forward(\*input, \*\*kwargs)

# File "D:\pythoncode\learn\a\deep\_learning4.1.py", line 62, in forward

# x = self.fc(x)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1532, in \_wrapped\_call\_impl

# return self.\_call\_impl(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1541, in \_call\_impl

# return forward\_call(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1522, in \_slow\_forward

# result = self.forward(\*input, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\container.py", line 217, in forward

# input = module(input)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1532, in \_wrapped\_call\_impl

# return self.\_call\_impl(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1541, in \_call\_impl

# return forward\_call(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\module.py", line 1522, in \_slow\_forward

# result = self.forward(\*input, \*\*kwargs)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\torch\nn\modules\linear.py", line 116, in forward

# return F.linear(input, self.weight, self.bias)

# RuntimeError: mat1 and mat2 shapes cannot be multiplied (1x1152 and 1568x128)

#

# 进程已结束,退出代码1

# 版本变化问题 ，修改这里 \_optimize\_trace变成 \_optimize\_graph

#

# File "D:\anaconda3\envs\portfolio\lib\site-packages\hiddenlayer\pytorch\_builder.py", line 71, in import\_graph

# torch\_graph = torch.onnx.\_optimize\_trace(trace, torch.onnx.OperatorExportTypes.ONNX)

# Traceback (most recent call last):

# File "D:\pythoncode\learn\a\deep\_learning4.1.py", line 88, in <module>

# hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1, 1, 30, 30]))

# File "D:\anaconda3\envs\portfolio\lib\site-packages\hiddenlayer\graph.py", line 143, in build\_graph

# import\_graph(g, model, args)

# File "D:\anaconda3\envs\portfolio\lib\site-packages\hiddenlayer\pytorch\_builder.py", line 71, in import\_graph

# torch\_graph = torch.onnx.\_optimize\_trace(trace, torch.onnx.OperatorExportTypes.ONNX)

# AttributeError: module 'torch.onnx' has no attribute '\_optimize\_trace'. Did you mean: '\_optimize\_graph'?

# torch版本不适配问题，安装torch1.1 最终解决

# Traceback (most recent call last):

# File "D:\pythoncode\learn\a\deep\_learning4.1.py", line 87, in <module>

# hl\_graph = hl.build\_graph(MyConvnet, torch.zeros([1, 1, 30, 30]))

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\hiddenlayer\graph.py", line 136, in build\_graph

# import\_graph(g, model, args)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\hiddenlayer\pytorch\_builder.py", line 68, in import\_graph

# hl\_node = Node(uid=pytorch\_id(torch\_node), name=None, op=op, params=params)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\hiddenlayer\pytorch\_builder.py", line 43, in pytorch\_id

# return node.scopeName() + "/outputs/" + "/".join([o.uniqueName() for o in node.outputs()])

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\hiddenlayer\pytorch\_builder.py", line 43, in <listcomp>

# return node.scopeName() + "/outputs/" + "/".join([o.uniqueName() for o in node.outputs()])

# AttributeError: 'torch.\_C.Value' object has no attribute 'uniqueName'

#

# 进程已结束,退出代码1

# graphviz没添加到环境变量，去下载exe安装程序安装然后添加到环境变量

# Traceback (most recent call last):

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\graphviz\backend\execute.py", line 85, in run\_check

# proc = subprocess.run(cmd, \*\*kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\subprocess.py", line 423, in run

# with Popen(\*popenargs, \*\*kwargs) as process:

# File "D:\anaconda3\envs\deeplearning\lib\subprocess.py", line 729, in \_\_init\_\_

# restore\_signals, start\_new\_session)

# File "D:\anaconda3\envs\deeplearning\lib\subprocess.py", line 1017, in \_execute\_child

# startupinfo)

# FileNotFoundError: [WinError 2] 系统找不到指定的文件。

#

# The above exception was the direct cause of the following exception:

#

# Traceback (most recent call last):

# File "D:\pythoncode\learn\a\deep\_learning4.1.py", line 90, in <module>

# hl\_graph.save("deep Learning/MyConvnet\_hl.png", format='png')

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\hiddenlayer\graph.py", line 356, in save

# dot.render(file\_name, directory=directory, cleanup=True)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\graphviz\\_tools.py", line 172, in wrapper

# return func(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\graphviz\rendering.py", line 119, in render

# rendered = self.\_render(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\graphviz\\_tools.py", line 172, in wrapper

# return func(\*args, \*\*kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\graphviz\backend\rendering.py", line 320, in render

# capture\_output=True)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\graphviz\backend\execute.py", line 88, in run\_check

# raise ExecutableNotFound(cmd) from e

# graphviz.backend.execute.ExecutableNotFound: failed to execute 'dot', make sure the Graphviz executables are on your systems' PATH

#

# 进程已结束,退出代码1

# 训练过程的可视化

import torch.nn as nn

class ConvNet(nn.Module):

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

super(ConvNet, self).\_\_init\_\_()

self.conv1 = nn.Sequential(

nn.Conv2d(in\_channels=1, out\_channels=16, kernel\_size=5, stride=1, padding=1),

nn.ReLU(),

nn.AvgPool2d(kernel\_size=2, stride=2))

self.conv2 = nn.Sequential(nn.Conv2d(16, 32, 3, 1, 1),

nn.ReLU(),

nn.MaxPool2d(2, 2))

self.fc = nn.Sequential(nn.Linear(in\_features=32 \* 7 \* 7, out\_features=128), nn.ReLU(), nn.Linear(128, 64),

nn.ReLU())

self.out = nn.Linear(in\_features=64, out\_features=10)

def forward(self, x):

x = self.conv1(x)

x = self.conv2(x)

x = x.view(x.size(0), -1)

x = self.fc(x)

output = self.out(x)

return output

MyConvnet = ConvNet()

from tensorboardX import SummaryWriter

import torch

SumWriter = SummaryWriter(log\_dir='deep Learning/log')

optimizer = torch.optim.Adam(MyConvnet.parameters(), lr=0.0003)

import hiddenlayer as hl

import time

import torch.nn as nn

import torch

from sklearn.metrics import accuracy\_score

from torchvision.datasets import FashionMNIST

import torch.utils.data as Data

import torchvision.transforms as transforms

import numpy as np

train\_data = FashionMNIST(root="./deep Learning/",

train=True,

transform=transforms.ToTensor(),

download=True)

train\_loader = Data.DataLoader(dataset=train\_data,

batch\_size=128,

shuffle=True,

)

print(len(train\_loader))

# 469

test\_data = FashionMNIST(root='./deep Learning/',

train=False,

download=False)

print(len(test\_data))

# 10000

test\_data\_x = test\_data.data.type(torch.FloatTensor) / 255.0

test\_data\_x = torch.unsqueeze(test\_data\_x, dim=1)

test\_data\_y = test\_data.targets

print(test\_data\_x.shape)

# torch.Size([10000, 1, 28, 28])

print(test\_data\_y.shape)

# torch.Size([10000])

class ConvNet(nn.Module):

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

super(ConvNet, self).\_\_init\_\_()

self.conv1 = nn.Sequential(

nn.Conv2d(in\_channels=1, out\_channels=16, kernel\_size=5, stride=1, padding=1),

nn.ReLU(),

nn.AvgPool2d(kernel\_size=2, stride=2))

self.conv2 = nn.Sequential(nn.Conv2d(16, 32, 3, 1, 1),

nn.ReLU(),

nn.MaxPool2d(2, 2))

# self.fc = nn.Sequential(nn.Linear(in\_features=32 \* 7 \* 7, out\_features=128), nn.ReLU(), nn.Linear(128, 64),

# nn.ReLU())

self.fc = nn.Sequential(nn.Linear(in\_features=32 \* 6 \* 6, out\_features=128), nn.ReLU(), nn.Linear(128, 64),

nn.ReLU())

self.out = nn.Linear(in\_features=64, out\_features=10)

def forward(self, x):

x = self.conv1(x)

x = self.conv2(x)

x = x.view(x.size(0), -1)

x = self.fc(x)

output = self.out(x)

return output

MyConvnet = ConvNet()

optimizer = torch.optim.Adam(MyConvnet.parameters(), lr=0.0003)

loss\_func = nn.CrossEntropyLoss()

history1 = hl.History()

canvas1 = hl.Canvas()

print\_step = 100

print(MyConvnet.fc[2].weight)

print(MyConvnet.fc[2].weight.shape)

# Parameter containing:

# tensor([[ 0.0166, 0.0044, 0.0586, ..., -0.0650, -0.0125, -0.0475],

# [ 0.0823, 0.0646, -0.0560, ..., 0.0881, 0.0576, -0.0146],

# [-0.0416, 0.0772, -0.0110, ..., 0.0261, 0.0081, 0.0283],

# ...,

# [-0.0356, 0.0712, -0.0448, ..., -0.0462, -0.0544, 0.0026],

# [-0.0192, 0.0113, -0.0848, ..., -0.0070, 0.0719, 0.0529],

# [-0.0513, -0.0832, -0.0089, ..., 0.0436, -0.0239, -0.0772]],

# requires\_grad=True)

# torch.Size([64, 128])

for epoch in range(5):

for step, (b\_x, b\_y) in enumerate(train\_loader):

output = MyConvnet(b\_x)

loss = loss\_func(output, b\_y)

optimizer.zero\_grad()

loss.backward()

optimizer.step()

if step % print\_step == 0:

# 记录训练集上的精度

output = MyConvnet(test\_data\_x)

\_, pre\_lab = torch.max(output, 1)

acc = accuracy\_score(test\_data\_y, pre\_lab)

history1.log((epoch, step),

train\_loss=loss,

test\_\_acc=acc,

hidden\_weight=MyConvnet.fc[2].weight

#hidden\_weight=MyConvnet.fc[2].weight.squeeze(dim=0)

)

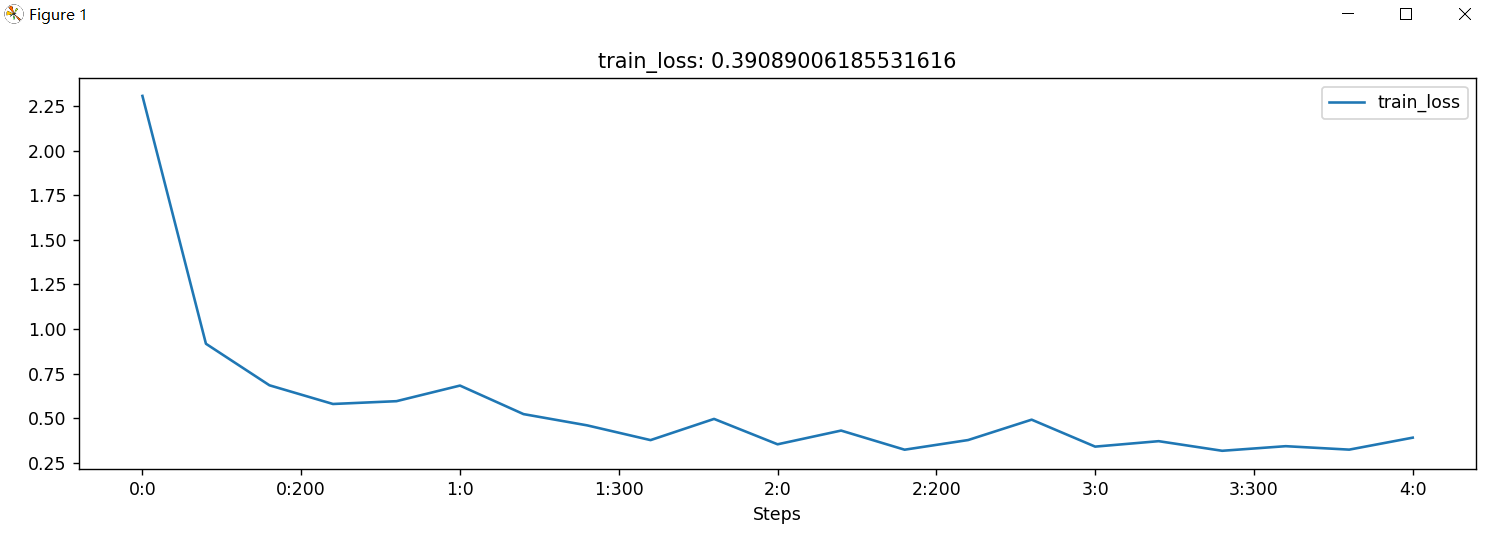
with canvas1:

canvas1.draw\_plot(history1['train\_loss'])

canvas1.draw\_plot(history1['test\_acc'])

#canvas1.draw\_plot(history1['hidden\_weight'])

# 尝试解决成功，换一个高版本的python

# protobuf requires Python '>=3.7' but the running Python is 3.6.13

# 使用Visdom进行可视化

import torch

from visdom import Visdom

from sklearn.datasets import load\_iris

import numpy as np

import torchvision.transforms as transforms

from torchvision.datasets import FashionMNIST

import torch.utils.data as Data

iris\_x, iris\_y = load\_iris(return\_X\_y=True)

print(iris\_x.shape)

print(iris\_y.shape)

# (150, 4)

# (150,)

vis = Visdom()

vis.scatter(iris\_x[:, 0:2], Y=iris\_y + 1, win='windows1', env='main')

vis.scatter(iris\_x[:, 0:3], Y=iris\_y + 1, win='3D 散点图', env='main',

opts=dict(marksize=4, xlabel='特征1', ylabel='特征2'))

vis = Visdom()

x = torch.linspace(-6, 6, 100).view((-1, 1))

sigmoid = torch.nn.Sigmoid()

sigmoidy = sigmoid(x)

tanh = torch.nn.Tanh()

tanhy = tanh(x)

relu = torch.nn.ReLU()

reluy = relu(x)

ploty = torch.cat((sigmoidy, tanhy, reluy), dim=1)

plotx = torch.cat((x, x, x), dim=1)

vis.line(Y=ploty, X=plotx, win='line plot', env='main',

opts=dict(dash=np.array(['solid', 'dash', 'dashdot']), legend=['sigmoid', 'tanh', 'relu']))

x = torch.linspace(-6, 6, 100).view((-1, 1))

y1 = torch.sin(x)

y2 = torch.cos(x)

plotx = torch.cat((x, x), dim=1)

ploty = torch.cat((y1, y2), dim=1)

vis.stem(X=plotx, Y=ploty, win='stem plot', env='main',

opts=dict(legend=['sin(x)', 'cos(x)'],

title='茎叶图'))

iris\_corr = torch.from\_numpy(np.corrcoef(iris\_x, rowvar=False))

vis.heatmap(iris\_corr, win='heatmap', env='main',

opts=dict(rownames=['x1', 'x2', 'x3', 'x4'],

columnnames=['x1', 'x2', 'x3', 'x4'],

title='热力图'

)

)

train\_data = FashionMNIST(root="./deep Learning/",

train=True,

transform=transforms.ToTensor(),

download=True)

train\_loader = Data.DataLoader(dataset=train\_data,

batch\_size=128,

shuffle=True,

)

for step, (b\_x, b\_y) in enumerate(train\_loader):

if step > 0:

break

print(b\_x.shape)

print(b\_y.shape)

# torch.Size([128, 1, 28, 28])

# torch.Size([128])

vis.image(b\_x[0, :, :, :], win='one image', env='MyimagePlot',

opts=dict(title='一张图像')

)

vis.images(b\_x, win='my batch image ', env='MyimagePlot',

nrow=16,

opts=dict(title='一个batch的图像')

)

texts = 'A Flexible tool for creating, organizing, and sharing visualizations of live, rich data. Supports Torch and Numpy.'

vis.text(texts, win='text plot', env='My image plot', opts=dict(title='可视化文本'))

# urllib3.exceptions.MaxRetryError: HTTPConnectionPool(host='localhost', port=8097): Max retries exceeded with url: /events (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x0000027D7F91E080>: Failed to establish a new connection: [WinError 10061] 由于目标计算机积极拒绝，无法连接。',))

#

# During handling of the above exception, another exception occurred:

#

# Traceback (most recent call last):

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\visdom\\_\_init\_\_.py", line 760, in \_send

# data=json.dumps(msg),

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\visdom\\_\_init\_\_.py", line 720, in \_handle\_post

# r = self.session.post(url, data=data)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\requests\sessions.py", line 577, in post

# return self.request('POST', url, data=data, json=json, \*\*kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\requests\sessions.py", line 529, in request

# resp = self.send(prep, \*\*send\_kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\requests\sessions.py", line 645, in send

# r = adapter.send(request, \*\*kwargs)

# File "D:\anaconda3\envs\deeplearning\lib\site-packages\requests\adapters.py", line 519, in send

# raise ConnectionError(e, request=request)

# requests.exceptions.ConnectionError: HTTPConnectionPool(host='localhost', port=8097): Max retries exceeded with url: /events (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x0000027D7F91E080>: Failed to establish a new connection: [WinError 10061] 由于目标计算机积极拒绝，无法连接。',))

#

# 进程已结束,退出代码0

# 解决方法，先在命令行运行，然后运行py文件

# (deeplearning) C:\Users\黄天佑>python -m visdom.server -p 8091

# Checking for scripts.

# Downloading scripts, this may take a little while

# ERROR:root:Error [WinError 10060] 由于连接方在一段时间后没有正确答复或连接的主机没有反应，连接尝试失败。 while downloading https://cdn.plot.ly/plotly-2.11.1.min.js

# It's Alive!

# INFO:root:Application Started

# INFO:root:Working directory: C:\Users\黄天佑\.visdom

# You can navigate to <http://localhost:8091>

