《深度学习平台与应用》作业三（20241018）

1、分别用简短的语言描述一下单帧 CNN 模型，Early Fusion，Late Fusion，3D CNN的做法以及区别。

2、如何使用RNN（如LSTM）建模视频中的长时间依赖？相比于3D CNN，这种方法有何特点？

3、结合课件中Spatio-Temporal Self-Attention (Nonlocal Block)部分的介绍，请补充下面的代码，

```python
import torch
import torch.nn as nn

class NonLocalBlock(nn.Module):
    def __init__(self, in_channels):
        super(NonLocalBlock, self).__init__()
        self.in_channels = in_channels
        self.inter_channels = in_channels // 2  # 缩减通道数，节约计算

        # 1x1x1 卷积层用于生成 Query, Key, Value
        self.query_conv = nn.Conv3d(in_channels, self.inter_channels, kernel_size=1)
        self.key_conv = nn.Conv3d(in_channels, self.inter_channels, kernel_size=1)
        self.value_conv = nn.Conv3d(in_channels, self.inter_channels, kernel_size=1)

        # 最终的输出通道映射回原始的 in_channels
        self.out_conv = nn.Conv3d(self.inter_channels, in_channels, kernel_size=1)

        # 用于归一化的 softmax
        self.softmax = nn.Softmax(dim=-1)

    def forward(self, x):
        # 输入维度: (N, C, T, H, W)
        batch_size, C, T, H, W = x.size()

        # 生成 Query, Key, Value
        query = self.query_conv(x).view(batch_size, self.inter_channels, -1)  # (N, C', T*H*W)
        key = self.key_conv(x).view(batch_size, self.inter_channels, -1)     # (N, C', T*H*W)
        value = self.value_conv(x).view(batch_size, self.inter_channels, -1) # (N, C', T*H*W)

        ####请在下面补充完整 Spatio-Temporal Self-Attention 计算过程






        return out

# 测试模块
if __name__ == "__main__":
    # 输入示例: Batch = 2, Channels = 64, Time = 8, Height = 32, Width = 32
    x = torch.rand(2, 64, 8, 32, 32)
    nonlocal_block = NonLocalBlock(in_channels=64)
    out = nonlocal_block(x)
    print("输入维度:", x.shape)
    print("输出维度:", out.shape)

```

4、 在风格迁移中，Gram矩阵的作用是什么？

5、 风格迁移和快速风格迁移的区别？

6、显著性图是怎么计算的（Saliency Map），它的用途是什么？

7、请补充完整下面显著性可视化的代码

```python
import torch
import torchvision
import torchvision.transforms as T
import numpy as np
import matplotlib.pyplot as plt
import requests
from PIL import Image

def download(url,fname):
    response = requests.get(url)
    with open(fname,"wb") as f:
        f.write(response.content)

def preprocess(image, size=224):
    transform = T.Compose([
        T.Resize((size,size)),
        T.ToTensor(),
        T.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
        T.Lambda(lambda x: x[None]),
    ])
    return transform(image)
def deprocess(image):
    transform = T.Compose([
        T.Lambda(lambda x: x[0]),
        T.Normalize(mean=[0, 0, 0], std=[4.3668, 4.4643, 4.4444]),
        T.Normalize(mean=[-0.485, -0.456, -0.406], std=[1, 1, 1]),
        T.ToPILImage(),
    ])
    return transform(image)

def show_img(PIL_IMG):
    plt.imshow(np.asarray(PIL_IMG))

if __name__ =='__main__':
    model = torchvision.models.vgg19(pretrained=True)
    for param in model.parameters():
        param.requires_grad = False


    download("https://bkimg.cdn.bcebos.com/pic/3bf33a87e950352ac65cae81db13ecf2b21192131da3?x-bce-process=image/format,f_auto/quality,Q_70/resize,m_lfit,limit_1,w_536","input.jpg")
    img = Image.open('input.jpg') # 这里可以替换为自己的图片

    X = preprocess(img) # X.shape: 1, 3, 224, 224
    model.eval()
    X.requires_grad_()
    output = model(X) # 1, 1000
    print(output.shape)
    score_max_index = output.argmax()
    score_max = output[0, score_max_index]

    #####
    # 需要实现课件上显著性可视化部分，具体的步骤为：
    # a) 通过反向传播获取梯度
    # b) 在RGB通道上取梯度绝对值的最大值
    # 请在下面补充代码：




    #####


    plt.imshow(saliency[0], cmap=plt.cm.hot)
    plt.axis('off')
    plt.savefig("output.png")
```