2024《人工智能导论》大作业

任务名称：不良内容图像检测

完成组号：4

小组人员：李之遥 龚笑旸 彭剑峰 丁毅峰

完成时间：2024.6.19

1．任务目标

基于暴力图像检测数据集，构建一个检测模型。该模型可以对数据集的图像进行不良内容的检测与识别。

2．具体内容

（ 1）实施方案

1. 加载和预处理图像数据，并生成训练集和验证集的数据加载器

2. 构建卷积神经网络（ CNN）模型并训练和验证

3. 打印训练和验证过程中的损失和准确率，以监控模型的训练进度和性能。

（ 2）核心代码分析

A. data_loader 部分代码定义了一个自定义的数据集类，用于加载和预处理图像数据，并生成训练集和验证集的数据加载器。

代码还包含了一个简单的测试部分，用于验证数据加载器的功能。

get_loaders 函数

使用 transforms.Compose 来组合多个预处理步骤：

transforms.Resize((224, 224))将图像大小调整为 224x224 像素。

transforms.ToTensor()将图像转换为 torch.Tensor，并缩放到 [0.0, 1.0] 范围。

transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])对图像进行标准化，使用给定的均值和标准差。

B.　model 部分定义了一个简单的卷积神经网络模型，并在文件的末尾进行了测试。

其中 forward 方法

```
def forward(self, x):
    x = self.pool(F.relu(self.conv1(x)))
    x = self.pool(F.relu(self.conv2(x)))
    x = x.view(-1, 64 * 56 * 56)
    x = F.relu(self.fc1(x))
    x = self.fc2(x)
    return x
```

这是模型的前向传播方法，定义了输入数据通过网络的方式。

输入数据 x 首先经过第一个卷积层 self.conv1，然后应用 ReLU 激活函数，接着通过最大池化层 self.pool。

接着，数据通过第二个卷积层 self.conv2，同样应用 ReLU 激活函数和最大池化。

然后，使用 view 方法将特征图展平为一维向量，以便可以作为全连接层的输入。这里假设了池化后的特征图大小为 56x56。

接下来，数据通过第一个全连接层 self.fc1，并应用 ReLU 激活函数。

最后，数据通过第二个全连接层 self.fc2，得到最终的输出。

C.　train 部分从数据加载到模型定义，再到训练和验证，都通过函

数进行了模块化处理。这使得代码更易于理解和维护。

验证函数 def validate(model, val_loader, criterion, device）循环遍历验证集，计算损失和准确率。返回平均损失和准确率。用于验证模型在验证集上的性能。

训练与验证函数 train_and_validate 执行训练和验证的循环。在每个 epoch 中，先设置模型为训练模式，然后遍历训练集进行前向传播、计算损失、反向传播和参数更新。每个 epoch 结束后，调用 validate 函数来验证模型在验证集上的性能，并打印相关信息。

主函数 __main__使用 get_loaders 函数加载训练和验证数据集。实例化 SimpleCNN 模型。并调用 train_and_validate 函数开始训练和验证过程。

3．工作总结

（ 1）收获、心得

通过大作业的实践，我们加深了对人工智能基础知识的理解，在完成作业的过程中，我们复习了机器学习、神经网络等概念，并尝试将这些理论应用于实际问题中。这不仅巩固了我们的理论知识，还让我们更加明白这些知识的实际应用价值。

其次，我们编程和解决问题的能力提高了。在编写代码、调试程序的过程中，不断遇到各种问题和挑战，我们通过查阅资料、请教同学，逐渐找到了解决问题的方法。这些经历让我们更加熟悉编程语言和工具的使用。

最后，我们还学会了团队协作和沟通。在作业完成过程中，我和小组

成员分工合作，共同探讨问题、分享经验。这不仅提高了我们的工作效率，还增进了我们之间的友谊和信任。

（ 2）遇到问题及解决思路

模型实现时遇到代码错误：发现是文件中设定的一个 hue 值会导致溢出，将其删除

模型在测试集上的性能不理想：对代码进一步优化

代码运行耗时过长：发现是预处理函数运行较慢，对其进行修改，先得到一个.pt 形式文件，后续模型训练时直接使用