用 pytorch 实现 CNN，对 cifar10 数据集进行分类

一、文件描述：

examples.py: 为深度学习第二课课上配套代码，仅供参考学习，不涉及作业

cifar10_cnn_torch_template.py: 作业代码，先将其复制为 cifar10_cnn.py，然后再 cifar10_cnn.py 中完成作业。

data：预先下载好的数据集，不需要提交给助教。

Elephant.jpg: 样例图片，在 example.py 中有用到。

Imagenet_class_index.json: 文件类别标签对应文件，在 example.py 中有用到。

二、任务描述：

本次作业旨在通过 PyTorch 框架实现一个卷积神经网络（CNN），以实现对 CIFAR-10 数据集的图像分类。CIFAR-10 数据集包含 60,000 张 32x32 彩色图像，分为 10 个类别，每个类别有 6,000 张图像。其中 50,000 张图像用于训练，10,000 张用于测试。

三、任务要求：

1. **网络实现**：使用 PyTorch 的 nn.Module 基类创建一个自定义的卷积神经网络。网络应至少包含一个卷积层（nn.Conv2d）和一个池化层（如 nn.MaxPool2d）。你可以根据自己的需要添加更多的卷积层、池化层、全连接层（nn.Linear）以及激活函数（如 ReLU）。（2 分）
2. **数据加载**：使用 PyTorch 的 DataLoader 和 Dataset 相关的 API 来实现 CIFAR-10 数据集的加载。确保数据被正确地归一化并划分为训练集和测试集。
3. **网络结构与参数调整**：修改网络结构和参数，观察不同结构对训练效果的影响。你可以尝试不同的激活函数、卷积核大小、步长、填充等。但请注意，你不能使用任何预训练的模型、接口或代码。（1 分）

4. **数据增强**：为了提高模型的泛化能力，使用数据增强技术，如随机裁剪、水平翻转、色彩抖动等。这些技术可以在训练过程中为模型提供更多样化的数据样本。（1 分）
5. **训练与评估**：训练网络至少 100 个 epoch，并在每个 epoch 结束后计算并打印训练集和测试集的准确率。确保你的模型在 100 个 epoch 后测试集的准确率能够达到 80%以上。（1 分）

四、提交要求：

1. **代码文件**：提交一个名为 cifar10_cnn.py 的 Python 代码文件，其中包含你的网络定义、数据加载、训练循环和评估逻辑。
2. **训练日志**：提交一个包含训练过程中每个 epoch 结束时的训练集和测试集准确率的文本文件或截图。这个日志应该清晰地展示模型性能随着训练的进行而提高的过程。（1 分）
3. **详细报告**：提交一份详细的报告（4 分），内容应包括：
    - 网络结构的设计及其背后的理由。
    - 你在训练过程中所做的任何修改或调整，以及这些修改如何影响模型的性能。
    - 数据增强技术的使用及其对模型泛化能力的影响。
    - 训练过程中遇到的任何挑战以及你是如何解决的。
    - 最终的模型性能评估，包括训练集和测试集的准确率。
4. **提交：**在 pack.py 中添加你的姓名学号并运行 python pack.py 压缩，提交得到的 zip 文件即可。