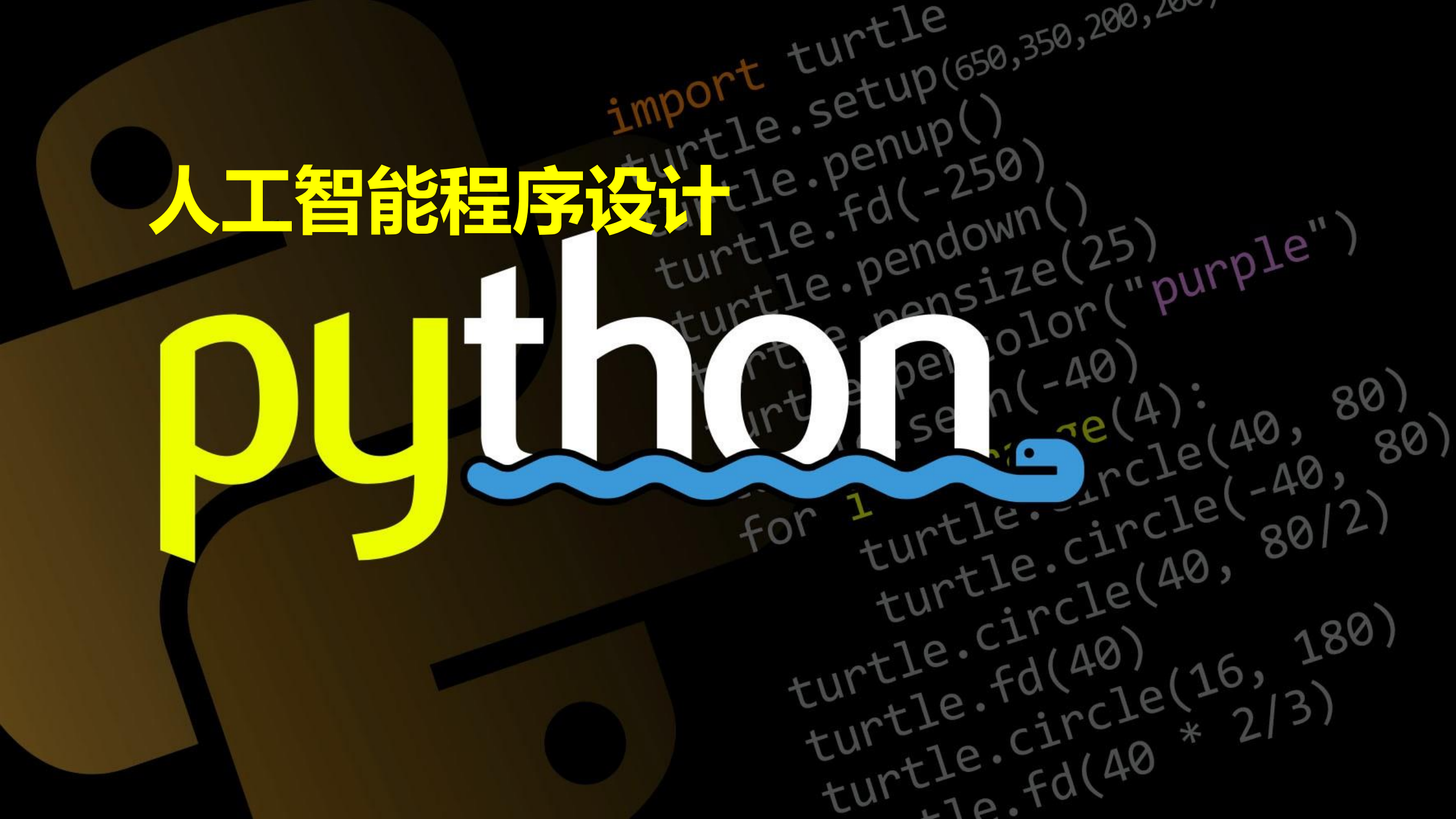
人工智能程序设计


python
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for 1 in range(4):
turtle.circle(40, 80)
turtle.circle(-40, 80)
turtle.fd(40)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)

人工智能程序设计

13.2 物体识别：RESNET50

北京石油化工学院 人工智能研究院

刘 强

章节导入

图像分类是计算机视觉中最基础也是最重要的任务之一

- 目标是将输入图像分配到预定义的类别中
- ResNet50是图像分类领域的经典模型
- 预训练模型具有强大的特征提取能力

13.2.1 图像分类基础原理

学习内容：

- 图像分类任务定义
- 分类与检测的区别
- ImageNet数据集与预训练模型

图像分类任务

图像分类任务是计算机视觉中最基础的任务

- 目标是将输入图像分配到预定义的类别中
- 只需要识别整张图像的主要内容
- 不需要定位物体的具体位置

分类与检测的区别

图像分类通常作为目标检测和其他复杂视觉任务的基础

任务类型	输出结果
图像分类	类别标签 + 置信度分数
目标检测	边界框坐标 + 类别标签 + 置信度分数

ImageNet数据集

ImageNet数据集是图像分类领域最重要的基准数据集：

- 包含超过1400万张图像
- 1000个类别
- 涵盖日常物品、动物、植物等

预训练模型的优势：

- 无需从零开始训练
- 已经学习到丰富的视觉特征

13.2.2 ResNet50模型详解

学习内容：

- 残差网络架构的核心创新
- 跳跃连接机制
- ResNet50结构特点

残差网络架构

残差网络架构是ResNet的核心创新

- 通过引入跳跃连接解决深度网络训练中的梯度消失问题
- 传统深度网络随着层数增加，训练变得困难
- ResNet通过残差学习使得训练超深网络成为可能

跳跃连接机制

跳跃连接机制允许信息直接从前面的层传递到后面的层

优势:

- 解决梯度消失问题
- 使网络能够学习恒等映射
- 即使某些层没有学到有用特征，也不会影响整体性能

ResNet50残差块结构

残差块的核心思想是学习残差映射 F(x) + x：

```
## ResNet50核心残差块结构
## 残差连接：F(x) + x
conv_output = Conv2D(filters)(x)  # 卷积变换
shortcut = x                      # 跳跃连接
output = Add()([conv_output, shortcut])  # 残差相加
```

ResNet50结构特点

ResNet50包含50个层:

- 卷积层、批归一化层、激活函数层
- 网络分为5个阶段，每个阶段包含不同数量的残差块
- 输入图像尺寸为224×224
- 最终输出1000个类别的概率分布

ResNet50模型加载

通过PyTorch轻松加载预训练模型：

预训练模型已包含在ImageNet上训练好的权重

```
## ResNet50模型加载
import torchvision.models as models

model = models.resnet50(pretrained=True)  # 加载预训练模型
model.eval()  # 设置为评估模式
```

13.2.3 物体识别项目实现

学习内容：

- 图像预处理流程
- 单张图像识别实现
- 批量图像处理

图像预处理流程

图像预处理是确保模型正确工作的关键步骤

ResNet50要求:

- 输入图像尺寸为224×224像素
- 像素值需要进行标准化处理

预处理步骤:

- 尺寸调整
- 中心裁剪
- 张量转换
- 归一化

示例 13.2.1：物体识别分类器

基础识别功能实现对单张图像的物体识别：

```
## 单张图像识别核心流程
## 图像预处理
preprocess = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

## 加载图像并预测
image = Image.open(image_path).convert('RGB')
input_tensor = preprocess(image).unsqueeze(0)

with torch.no_grad():
    output = model(input_tensor)
    probabilities = torch.softmax(output, dim=1)
    top5_prob, top5_catid = torch.topk(probabilities, 5)
```

识别结果处理

输出内容:

- 最可能的类别及其置信度
- Top-5预测结果
- 处理各种格式的图像文件

实践练习

练习 13.2.1：基础图像分类

1. 使用ResNet50模型对提供的测试图像进行分类
2. 观察不同类型图像的识别效果
3. 分析模型在哪些类型的图像上表现较好，哪些类型识别困难

实践练习

练习 13.2.2：批量图像处理

1. 收集一组包含不同物体的图像
2. 使用批量识别功能进行处理
3. 统计识别准确率，分析错误识别的原因

实践练习

练习 13.2.3：Ask AI实现自定义分类器

1. 使用Ask AI工具，基于ResNet50实现一个针对特定领域的图像分类器
2. 可选领域：动物分类、食物分类等
3. 学习如何进行模型微调和迁移学习

实践练习

练习 13.2.4：性能优化实验

1. 比较不同预处理方法对识别效果的影响
2. 测试不同输入图像尺寸对模型性能和速度的影响
3. 探索模型优化的方法