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

12.3 深度学习核心模型

北京石油化工学院 人工智能研究院

刘 强

本节概述

深度学习的能力来源于其多样化的网络架构

- 本节介绍神经网络的基础概念
- 重点讲解两种最基础也是最重要的深度学习模型
  - 全连接神经网络
  - 卷积神经网络（CNN）

12.3.1 卷积神经网络基本结构

学习内容：

- 从神经网络到CNN的设计动机
- 卷积层、池化层、激活函数的原理与实现
- 简单CNN架构的完整数据流

从神经网络到CNN

神经网络基础:

- 传统神经网络由神经元组成
- 每个神经元的输出 = 激活函数(权重 × 输入 + 偏置)
- 神经元通过权重学习输入特征的重要性
- 通过激活函数（如ReLU）引入非线性

CNN的设计动机

传统全连接网络处理图像的问题:

- 参数过多
- 忽略空间结构

卷积神经网络（CNN）的解决方案:

- 专门为处理具有网格结构数据（如图像）而设计
- 通过卷积操作保留空间结构
- 用局部连接和参数共享大幅减少参数数量

卷积操作原理

卷积层的工作方式:

- 使用多个卷积核（滤波器）在输入数据上滑动
- 计算局部区域的加权和
- 每个卷积核专门检测特定的特征模式（如边缘、纹理或形状）

层次化特征学习:

- 浅层卷积层提取基本特征（边缘、角点）
- 深层卷积层组合形成更复杂的特征表示（形状、物体部件）
- 最终识别完整的物体

CNN的基本组件

组件	功能
卷积层	特征提取，通过卷积核与输入进行卷积运算
池化层	降维和提供平移不变性，减少计算量
激活函数	引入非线性，增强网络的表达能力

简单CNN架构：卷积层

卷积层：特征提取器

卷积层通过卷积核在输入图像上滑动，提取局部特征如边缘、纹理。

```
conv = nn.Conv2d(1, 16, kernel_size=3, padding=1)  # 1通道输入，16通道输出
```

参数说明：

- **1**：输入通道数（灰度图）
- **16**：输出通道数（特征图数量）
- **kernel_size=3**：使用3×3卷积核
- **padding=1**：保持特征图尺寸

简单CNN架构：激活函数

激活函数：非线性变换

激活函数引入非线性变换，使网络能够学习复杂关系。

```
x = torch.relu(conv(x))  # ReLU激活函数
```

ReLU函数：

- 计算公式：f(x) = max(0, x)
- 计算简单且梯度稳定
- 是CNN中最常用的激活函数

简单CNN架构：池化层

池化层：降维与不变性

通过2×2窗口的最大池化操作，将特征图尺寸减半。

```
pool = nn.MaxPool2d(2, 2)  # 2×2最大池化
x = pool(x)                # 特征图尺寸减半
```

池化的作用：

- 降低计算量
- 提供平移不变性
- 使网络对输入的小幅移动不敏感

简单CNN架构：全连接层

全连接层：分类决策

将2D特征图展平为1D向量，然后映射到分类结果。

```
fc = nn.Linear(16 * 14 * 14, 10)  # 展平后的特征到类别映射
x = x.view(-1, 16 * 14 * 14)      # 展平操作
x = fc(x)                         # 分类输出
```

参数说明：

- **16 * 14 * 14**：展平后的特征维度
- **10**：分类类别数

完整数据流

简单CNN的数据处理流程：

输入图像 → 卷积提取特征 → ReLU激活 → 池化降维 → 展平 → 全连接分类 → 输出结果

这种结构实现了从像素级输入到语义级输出的自动特征学习。

12.3.2 深度卷积神经网络：AlexNet

学习内容：

- AlexNet的历史意义和技术创新
- 多尺度卷积层设计：从11×11到3×3
- 三层递进式全连接层设计

AlexNet的历史意义

AlexNet是深度学习发展史上的里程碑式架构：

- 由Alex Krizhevsky等人在2012年提出
- 在ImageNet图像识别竞赛中取得压倒性胜利
- 将错误率从26%降低到15%
- 标志着深度学习时代的正式到来

AlexNet的技术创新

突破性贡献:

- 首次证明深度卷积网络在大规模图像识别任务中的能力
- 推动GPU在深度学习中的广泛应用

技术创新:

- 使用ReLU激活函数解决梯度消失问题
- 采用Dropout防止过拟合
- 使用数据增强技术扩充训练数据
- 利用GPU并行计算加速训练过程

AlexNet架构设计

网络结构相比简单CNN更加深层和复杂：

- 5个卷积层
- 3个全连接层

采用**多尺度特征提取**策略，通过不同大小的卷积核实现从粗糙到精细的层次化特征学习。

AlexNet卷积层设计（一）

第一卷积块：使用11×11大卷积核

- 步长为4，快速降维：224×224 → 55×55
- 从3个颜色通道扩展到96个特征通道
- 3×3最大池化进一步缩小到27×27
- 大卷积核能够快速捕获全局特征和粗粒度模式

AlexNet卷积层设计（二）

第二卷积块：采用5×5卷积核

- 在27×27的特征图上进行卷积
- 96个通道扩展到256个通道
- 通过池化降维到13×13
- 中等尺寸卷积核关注更精细的特征组合

AlexNet卷积层设计（三）

第三、四、五卷积层：连续使用3×3小卷积核

- 在13×13的较小特征图上进行深层特征提取
- 通道数变化：256 → 384 → 384 → 256
- 第五层后通过3×3池化将特征图缩小到6×6

设计理念：从大到小的卷积核实现从粗糙到精细的渐进式特征学习

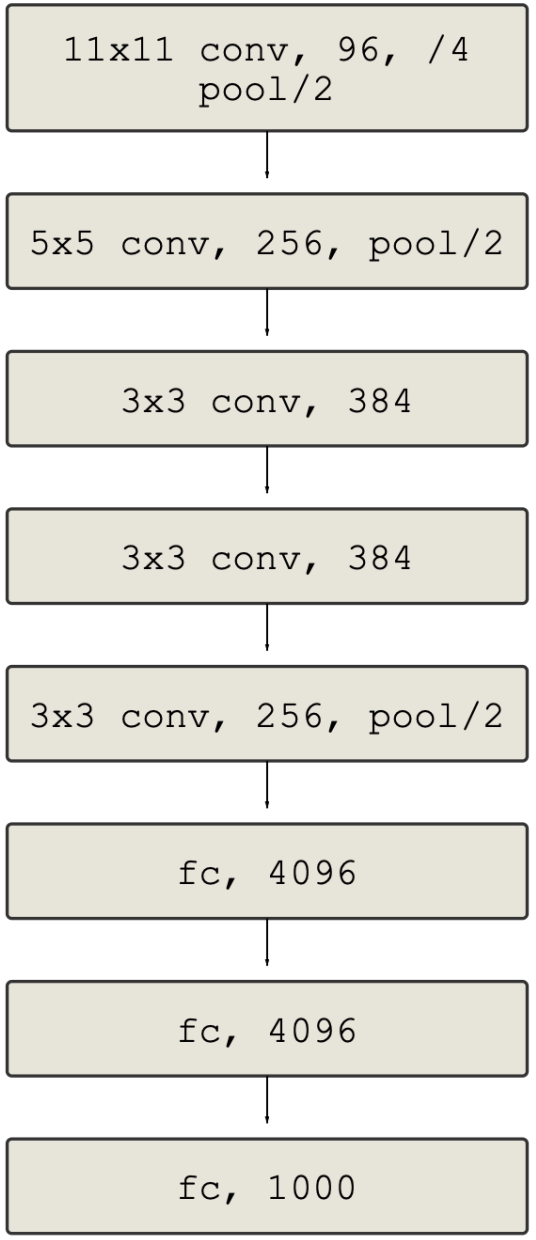
AlexNet全连接层设计

三层递进式设计实现从空间特征到语义决策的转换：

层	输入维度	输出维度	功能
第一全连接层	6×6×256=9216	4096	空间特征→语义空间
第二全连接层	4096	4096	特征组合学习
第三全连接层	4096	1000	分类输出

AlexNet网络结构图

图 12.3.1 AlexNet的网络结构



实践练习

练习 12.3.1：Ask AI实现简单CNN

1. 使用Ask AI帮助实现一个简单的CNN模型

2. 包含1个卷积层、1个池化层、1个全连接层

3. 理解每个层的参数设置和作用

提示：向AI描述："请帮我用PyTorch实现一个简单的CNN模型，包含卷积层、池化层和全连接层，并解释每个层的参数含义。"

实践练习

练习 12.3.2：Ask AI完善AlexNet结构

1. 使用Ask AI实现完整的AlexNet架构

2. 包含5个卷积层和3个全连接层

3. 正确设置卷积核大小、步长、Dropout和ReLU

提示：向AI说明："请帮我用PyTorch实现AlexNet网络结构，包含完整的特征提取部分和分类部分。"

实践练习

练习 12.3.3：Ask AI解释模型差异

1. 分析简单CNN与AlexNet的架构差异

2. 比较各自适用的数据集和任务类型

3. 对比参数数量和计算复杂度

提示：询问AI："请比较简单CNN和AlexNet的区别，包括网络深度、参数数量、适用场景等方面。"