信息工程学院

数据结构综合训练

(2024~2025 学年第二学期)

报告题目：简易神经网络的实现

姓名：	曾瑞安
专业：	计算机科学与技术
年级班级：	2023 级计算机科学与技术 2 班
学号：	2023013332
指导教师：	耿耀君
完成日期：	2025 年 07 月 16 日

1 综合训练目的和要求

本综合训练是计算机科学与技术、信息管理与信息系统、软件工程、电子商务专业重要的实践性环节之一，是在学生学习完《面向对象程序设计 (C++)》、《数据结构》课程后进行的一次全面的综合练习。本课综合训练的目的和任务：

1. 巩固和加深学生对 C++ 语言、数据结构课程的基本知识的理解和掌握
2. 掌握 C++ 语言编程和程序调试的基本技能
3. 利用 C++ 语言进行基本的软件设计
4. 掌握书写程序设计说明文档的能力
5. 提高运用 C++ 语言、数据结构解决实际问题的能力

2 综合训练具体内容

2.1 神经网络核心框架设计与构建：

计算图的抽象化设计：项目的核心是将神经网络的运算流程抽象为一个有向无环图（DAG）。我们设计了 Node（节点）与 Graph（图）两个核心类。Node 封装了 Matrix（矩阵）作为其运算值，并存储了指向其父节点的 weak_ptr，以此构建出完整的计算依赖关系。这种设计使得网络结构的定义极为灵活，能够动态地构建任意复杂度的网络模型。

正向传播的实现：通过将加法、乘法、激励函数（如 Sigmoid）等操作均封装为建立新节点的函数 (ops)，我们实现了直观的正向计算流程。每一步运算都会在计算图中产生一个新的 Node，并记录其与父节点的依赖关系，为后续的反向传播奠定基础。

2.2 反向传播与拓扑排序的实现：

拓扑排序的应用：为了实现自动微分（Auto-differentiation），我们在 Node 的 backward() 方法中，首先对整个计算图进行拓扑排序。通过深度优先搜索（DFS）从损失节点开始遍历所有父节点，我们得到了一个反向的节点计算序列。这个序列确保了在计算任何节点的梯度之前，其所有后续节点的梯度都已经计算完毕，完美地解决了链式法则中的依赖问题。

自动梯度计算：每个运算操作在建立 Node 的同时，都会通过 std::function 动态地为其绑定一个 _backward 函数。在拓扑排序后的反向传播过程中，每个节点会依序执行自己的 _backward 函数，将上游传来的梯度（存储在自身的 grad 矩阵中）根据该节点的运算类型（如加法、乘法）正确地传递给其父节点，从而实现了全自动的梯度计算与分发。

2.3 模型训练与数据增强实践：

利用构建好的框架搭建一个用于 MNIST 手写数字识别的 MLP 模型。在训练过程中，为了提升模型的泛化能力以适应真实、多样的手写笔迹，我们引入了数据增强技术。在每个训练批次中，对输入的 28x28 图像矩阵进行随机的空间变换（平移、旋转）与像素级干扰（高斯噪声），极大地丰富了训练样本，使模型学习到更鲁棒的数字特征。

2.4 端到端系统集成与部署：

将训练好的模型参数存储，并在一个基于 Qt 的图形化桌面应用中进行集成。应用程序内部实现了一套完整的图像预处理管线，能将用户在画布上书写的任意笔迹，标准化为与训练时一致的 28x28 矩阵格式。最终，应用程序直接在内部调用神经网络框架执行预测，并实时反馈识别结果，完成了从底层算法到上层应用的完整技术闭环，验证了整个框架的实用性与正确性。

3 总体设计

3.1 项目组成

本项目框架主要可分为核心类、核心计算、模型训练、前端展示.

3.1.1 核心类

在 core 中我们设计了 Matrix 作为矩阵管理类, 并封装相应的计算函数, 在设计时我们使用一维数组来模拟二维数组, 在内存读取方面更容易命中, 增加计算的速度.
我们设计了 Node 类作为图的节点类, 每个节点内部含有两个 Matrix 类, 分别作为前向计算值和反向计算梯度, 并使用 vector<weak<» 方式持有自己的父节点.
设计 Graph 类作为 Node 友元类, 并删除 Node 创造函数确保所有的 Node 节点都由 Graph 管理, 便于内存管理.

3.1.2 核心计算

核心计算代码位于 src/opts 文件夹中
我们设计了 ops 文件包含所有函数计算, 对于每个运算我们都会创建一个新的 Node 节点并设置对应的反向传播函数, 并确保由 Graph 管理.

3.1.3 模型训练

为了验证模型的可行性, 我们设计完成手写数字识别, 并将其作为我们的展示, 我们使用一个输入层、一个隐藏层、以及一个输出层, 由于我们设计的模型是动态生成节点因此在后续添加隐藏层较为容易, 亲自测试每天加一个隐藏层不超过 5 行代码.
为了读取文件我们设计了 DataLoader 与 DataSet 作为基类并封装了相应的方法, 并设计了 CsvDataLoader 作为本次手写数字识别的读取类.

3.1.4 前端展示

基于 QT 的图形化界面应用, 并能够让用户书写任何内容, 通关按钮将画布内容转为 28*28 矩阵并输出对应预测结果.

3.2 框图与流程图

3.2.1 程序组成框图

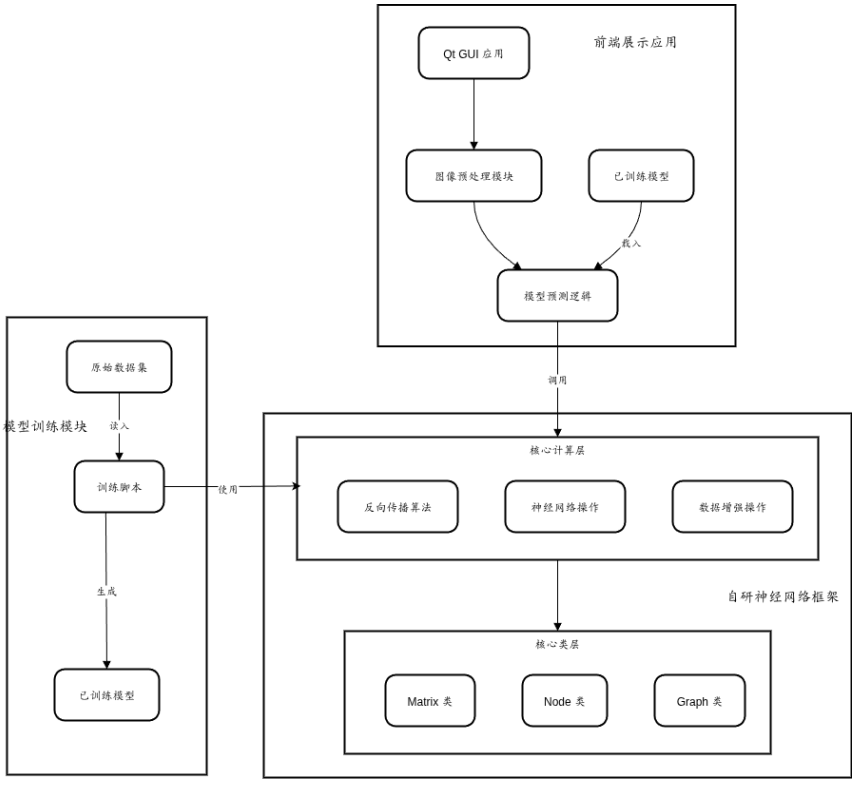

图 1: 框图

3.2.2 流程图

4 详细设计说明

整体代码分为四个模块, 核心类模块

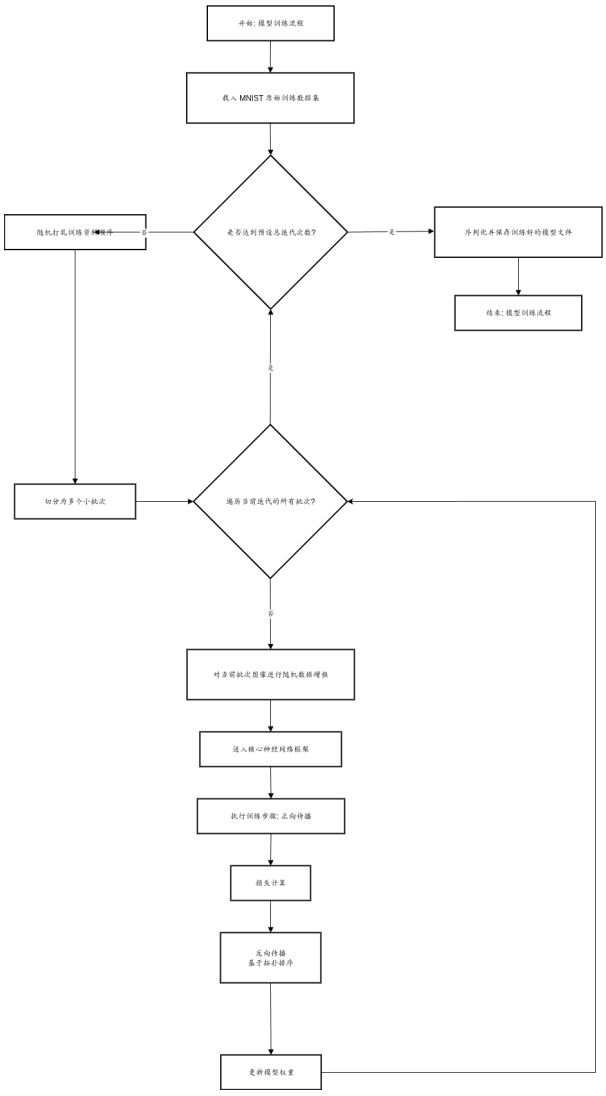

图 2: 模型训练流程图

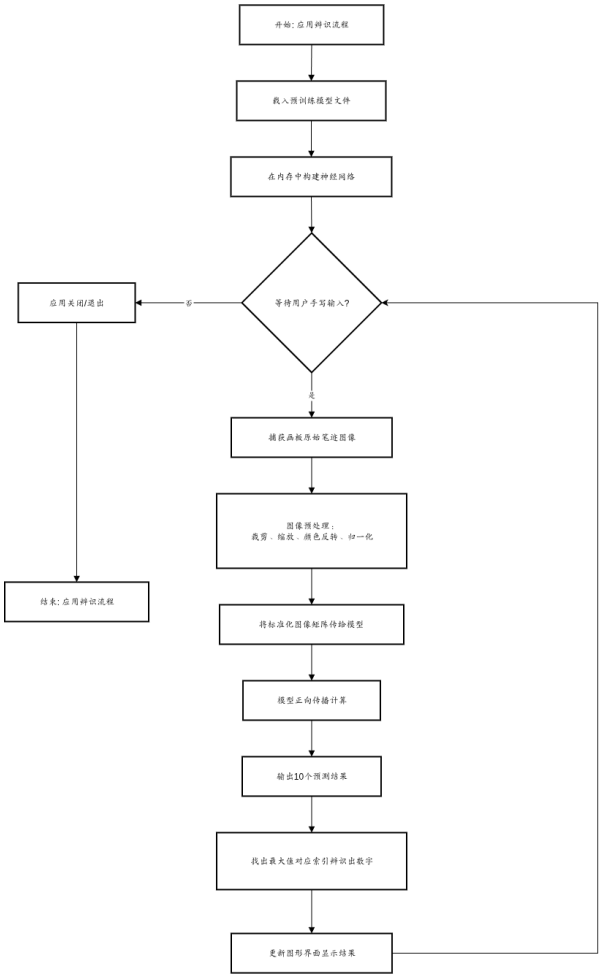

图 3: 手写数字识别