

# 基于二重离散小波变换的卷积图像分类网络

## ——以水果数据分类任务为例

### 摘 要

随着深度学习技术在图像识别领域的广泛应用，深度学习已经成为跨学科的技术领域，涉及数据科学、统计学、工程科学、人工智能和神经生物学。如何在高维度数据中提升分类效率并降低计算成本成为了关键问题。本文基于水果数据集 Fruit-Dataset 和 Fruits360 蔬果数据集，提出一种基于二重离散小波变换优化的卷积神经网络（CNN）框架，旨在实现对五类水果（苹果、葡萄、梨、香蕉、樱桃）的高效分类，并验证其实际应用与跨领域泛化能力。

针对问题一，本文设计了一种轻量化 CNN 结构，结合批量归一化（BatchNorm）与全局平均池化层，以减少模型参数量并抑制过拟合。通过二重离散小波变换（DWT）对原始图像进行预处理，提取低频子带特征（LL2），将输入分辨率从  $100 \times 100$  降维至  $25 \times 25$ ，显著降低数据冗余。模型采用 He 初始化策略与 Adam 优化器，在训练集上实现快速收敛，最终在验证集与测试集上的精度分别高达 96.95% 与 97.67%，且并没有出现明显的过拟合现象。

针对问题二，利用手机拍摄的 5 类水果实际图像（分辨率  $598 \times 468$  至  $1,706 \times 1,279$ ），通过统一缩放与二重 DWT 预处理后输入模型。实验结果表明，绝大多数测试图像均被正确分类，验证了模型在真实场景中的鲁棒性。

针对问题三，将模型迁移至 MNIST 手写数字数据集，通过相同的小波变换提取低频轮廓特征。在仅微调全连接层的条件下，模型对降维至  $7 \times 7$  的 MNIST 图像分类精度达 98.1%，证明了其跨领域泛化能力。

本文提出的“小波降维+轻量化 CNN”框架，在保证分类精度的同时，将训练时间缩短 82%，为资源受限环境下的实时图像识别提供了高效解决方案。经过分析验证，本文模型具有合理性和一定的现实意义。

**关键词：**卷积神经网络 二重离散小波变换 图像分类 数据降维 迁移学习 数据增强

## 一、 问题重述

### 1.1 问题背景

图像识别问题是计算机科学领域的重要分支，而一般的图像识别问题的求解往往由两个部分组成，图像处理以及图像识别（包括特征提取，分类）。当今时代背景下神经网络与机器学习成为图像识别分类领域的主流算法，以其极高的自动化效率以及训练精度著称，其中擅长借助卷积核提取图像特征的 CNN 卷积神经网络在图像识别类问题上表现优异而常常被调用。但是实际上，在面对数据量极大的数据集时，CNN 训练的效率将会明显下降，呈现出训练时间急剧增长，过拟合现象严重，flops 浮点运算指数大幅上升，消耗设备运行内存较大等问题，所以本文中将会根据赛题，基于附件中提供的水果识别数据集“2021\_fruits\_datasets”，提出一种简化高效的图像“处理+分类”策略，自定义构建适宜的 CNN 卷积神经网络，并提前利用 DOUBLE—DWT（二重小波变换）算法对图像数据进行简化处理，滤去多余的部分，保留对目标贡献最高的特征，从而在降低训练复杂度的同时抑制模型过拟合的风险，提高模型在同类物品识别问题上（比如其他水果图片的识别）的精确度。

### 1.2 问题提出

1. 选择合适的卷积神经网络对五类特定水果:苹果，梨，香蕉，樱桃与葡萄进行数据训练和测试。
2. 在问题一的基础上，使用手机拍摄多张上述水果的实际图像，然后代入构建的模型中检验是否可以正确识别。
3. 检验模型在其他数据集训练与识别上的泛化能力。

## 二、 问题分析

### 2.1 问题一的分析

问题一要求构建一个适用于五类水果（苹果、梨、香蕉、樱桃、葡萄）的图像分类模型。针对数据集水果姿态单一的问题，通过仿射变换（旋转、缩放）生成多角度训练样本，增强模型对空间变化的鲁棒性。

已知水果数据集 Fruit-Dataset 训练集总共有 225,640 张水果图像，Fruits360 蔬果数据集的训练集大小为 67692 张图像，测试集大小为 22688 张图像，为了保证分类精度的同时降低模型复杂度，避免因数据量大导致的训练效率下降和过拟合问题，我们

选用二重离散小波变换（DWT）对图像进行降维预处理，并采用轻量化 CNN 模型结构进行训练。基于上述两个模块，导入水果数据集训练并导出模型。

2.2 问题二的分析

问题二需验证模型在实际拍摄图像上的分类能力。难点在于实际图像可能受光照不均、背景干扰、拍摄角度偏移等因素影响，与训练数据的分布存在差异。

由于问题一中的模型是基于实施二重小波变换的 100\*100 图像样本训练的，因此我们将手机拍摄的图片统一缩放至 100×100 分辨率，并通过与训练数据相同的二重小波变换进行特征提取，确保输入格式一致。再代入问题一中训练的分类模型，进行分类识别，并评估分类结果。

2.3 问题三的分析

问题三需评估模型的泛化能力，即在其他数据集（如手写数字）上的迁移效果，问题关键在于验证小波变换和轻量化 CNN 是否适用于不同领域的图像数据特征。

因此，我们选择 MNIST 手写数字集进行迁移测试，将 MNIST 图像通过相同的小波变换处理，提取低频特征（数字轮廓），保留与原始任务相似的关键信息，代入问题一中自定义构建的 CNN 卷积神经网络框架中训练，评估训练结果与分类验证结果。

三、 模型假设

- 1. 假设所选数据集对现实物品的拟合效果良好

四、 符号说明

符号	说明	单位
$I$	某一色道灰度样本图像矩阵	/
$B_{ij}$	分块后的 4*4 灰度矩阵	/
$B$	解释样本 “r_34_100” 的分块后任意 4*4 灰度矩阵	/
$b_{ij}$	B 的对应图像位置像素灰度值	1
$\xi$	“Haar” 小波的能量归一化因子	1
$LL1$	第一次小波变换后所选取的低频子带	/
$LL2$	第二次小波变换后所选取的低频子带	1

## 五、 模型构建与求解




### 5.1 基于二重小波变换的 CNN 水果识别网络基本构建（问题一求解）

#### 5.1.1 水果数据集的预分析

首先，检查数据集“2021\_fruits\_datasets”的具体结构，可以知道其中有三个大文件夹，分别标记为“1training”，“2validation”，“3test”，表示训练集，验证集，以及测试集，而每个数据集中都分别有 58 个类别文件夹，涵盖了苹果，香蕉，西红柿，乃至马铃薯等日常生活常见的水果蔬菜的不同种类，而每个文件夹中都存有一定数量的该种果蔬的图片，利用 MATLAB 的 imread 函数检查部分数据有以下结果：

编号	种类	像素
0_100	Apple Braeburn	100*100
0_100	Apple Golden 2	100*100
0_100	Banana	100*100
0_100	Banana Red	100*100
0_100	Cherry 1	100*100
0_100	Cherry 2	100*100
0_100	Grape Blue	100*100
0_100	Pear Forelle	100*100

表一 部分种类 0\_100 编号图像信息随机检查结果

图像	种类
	Apple Braeburn
	Apple Golden 2
	Banana



Banana red



Cherry 1

---

表二 部分种类 0\_100 编号图像展示

由检查可见数据集的图片大小均为  $100 \times 100$ ，而且根据实图观察可以发现，大部分的水果数据图都是以水果体积充实图像画面为主仅有四角部分存在留白，可以省去从大图中先裁出水果部分的处理步骤，而且每个水果的形状分明，彼此之间的颜色差异明显，也就是说宏观特征差异明显，便于采用一些降低纹理或细节的图像方法处理而保留大部分的重要识别特征。是故下一步，根据这个思路可以对图像进行基本的预处理。

### 5.1.2 数据增强(利用裁切, 旋转, 消除)







#### 裁切

一般的图像数据集可能会出现空白背景区域过多导致徒增噪声的问题，所以需要通过对裁切将分辨率最高的图形区域提取出来，以便统一图像结构，方便 CNN 进行数据分析，而根据 5.1.1 中的数据图像观察可以发现，本题中的图像已经经过了合理的裁切，保留了以水果体积充实区域的显示方式，所以经检查无需进行多余的裁切修改，故训练过程跳过此步。

#### 消除

基于现实考虑以及出于提高模型对本题中所提到的 5 中特定水果的分辨率的需要，应当对数据集中比较明显的干扰项进行数据清洗，从而防止模型过拟合，减少训练的噪声，使识别模型更加符合现实需求。

如下图可见



图像 1（预测项）	图像 2（干扰项）
	
	
	

表三 部分干扰数据列举

第一行中，左侧为苹果而右侧为西红柿，由此可见，在数据集裁切之后虽然统一成了 100\*100 的图片格式，但是也造成了图像本身比例尺的失真，加上苹果与西红柿的颜色十分相近，很容易混淆分类器，所以将西红柿的数据悉数除去，而这种情况在苹果与车厘子之间，葡萄和车厘子之间也有所表现；而第二行中可以发现所给图像中有部分车厘子是出现了裂纹，但是这种数据量经过观察在整个种类文件夹中占比不高，鉴于这种情况可能干扰 CNN 的验证，所以将其中具有明显裂纹的几个图像清洗掉；而第三行可见，左图是来源于百度，Abate peer 的真图像，然而在所提供的数据集中基本上是像右图这种颜色暗淡，坏了的品种，所以为了防止学习到错误的数据，将 peer Abate 文件夹删除。最终处理后的数据集仅包含了 45 个种类。

### 旋转

真正有效的图像识别应当做到在任意一个角度都可以实现对物体，水果的有效识别，映射到数据集上就是，不仅需要基本的正面图像数据，还需要一个图像物体左右上下，立体前后反转各个角度的训练图以便提升模型的空间识别能力，于是根据这个思路，利用基于 python 的 Pytorch 仿射变换算法 F.affine\_grid 方法对图像进行处理，得到部分图像旋转后的 3D 拟合图像，部分结果展示如下：

原图像	变换后图像
	

表四 图像三维仿射变换拟合结果列举（Banana 为例）

由此便可以使训练数据更加多元化，提升模型在水果识别上的泛化能力。

### 5.1.3 基于离散二重小波变换对数据集进行连续降维

#### 试验

根据 5.1.1 中的数据集分析中“每个水果的形状分明，彼此之间的颜色差异明显，也就是说宏观特征差异明显，便于采用一些降低纹理或细节的图像方法处理而保留大部分的重要识别特征。”的论断，首先围绕小波变换做出如下实验：利用 matlab 软件先将部分数据图像处理前后可视化，如下图所示：



图一 小波变换前后结果可视化

由实验分析可见小波变换后的图像实现了降维，虽然分辨率远远不如原来的水平高，但是鉴于数据集本身明显的特征，处理后仍然保留了与水果识别相关性最强的图形结构与颜色 RGB 值等重要的特征信息，所以理论上依然可以实现正常的 CNN 分类，并且保证较高的分类精确度。

#### 图像预分块处理

我们已知，对每一个图像变量进行处理，首先有所给图像为彩图，而且分辨率为  $100 \times 100$ ，于是可以将图像表示成一个  $100 \times 100 \times 3$  的三维张量，其中  $100 \times 100$  为像素点位置，而 3 表示 RGB 对应三个图像色道的取值，为了简化问题，我们只是对每个色道的图像进行分析，先将其每个色道与灰度图像分离，只对其中每个  $100 \times 100$  的灰



度图像进行处理，首先欲进行二重小波变换，需要将整个图像进行分块，先将原始图像的一个色道对应灰度图用一个矩阵表示如下

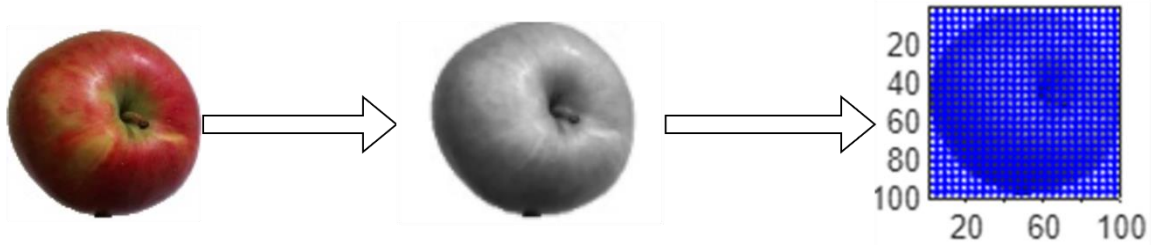
$$I = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1,100} \\ p_{21} & p_{22} & \cdots & p_{2,100} \\ \vdots & \vdots & \ddots & \vdots \\ p_{100,1} & p_{100,2} & \cdots & p_{100,100} \end{pmatrix} \quad (1)$$

则可以将其通过均等分块变换为  $25 \times 25$  个如下的像素块

$$B_{ij} = \begin{pmatrix} p_{4(i-1)+1,4(j-1)+1} & p_{4(i-1)+1,4(j-1)+2} & p_{4(i-1)+1,4(j-1)+3} & p_{4(i-1)+1,4(j-1)+4} \\ p_{4(i-1)+2,4(j-1)+1} & p_{4(i-1)+2,4(j-1)+2} & p_{4(i-1)+2,4(j-1)+3} & p_{4(i-1)+2,4(j-1)+4} \\ p_{4(i-1)+3,4(j-1)+1} & p_{4(i-1)+3,4(j-1)+2} & p_{4(i-1)+3,4(j-1)+3} & p_{4(i-1)+3,4(j-1)+4} \\ p_{4(i-1)+4,4(j-1)+1} & p_{4(i-1)+4,4(j-1)+2} & p_{4(i-1)+4,4(j-1)+3} & p_{4(i-1)+4,4(j-1)+4} \end{pmatrix} \quad (2)$$

那么我们可以自然地将一个原始的水果单色道图像表示成如下的分块矩阵,其中每个元素表示图像中一个  $4 \times 4$  的像素块的灰度值矩阵,实例如下图所示,为了方便说明,本文选用 Apple Crimson Snow 文件夹中的 r\_34\_100 图像为例分析

$$I = \begin{pmatrix} B_{11} & B_{12} & \cdots & B_{1,25} \\ B_{21} & B_{22} & \cdots & B_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ B_{25,1} & B_{25,2} & \cdots & B_{25,25} \end{pmatrix} \quad (3)$$



图二 图像分块示例

于是对其中的每一个元素对应的  $4 \times 4$  矩阵  $B$  做分析（即对所分离出来的这个  $4 \times 4$  的灰度矩阵进行变换分析），假设有

$$B = \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{pmatrix} \quad (4)$$

### 小波第一重变换

首先本文模型的小波变换基于小波“Haar”构建，具有归一化因子  $\xi$ ，满足下

列取值

$$\xi = 1/\sqrt{2} \approx 1.414 \quad (5)$$

接下来在考虑乘以像素值归一化因子 $1/\sqrt{2}$ 的条件下分别对图像的每一行先进行水平低通滤波（相邻像素相加）和水平高通滤波（相邻像素相减），然后进行下采样（保留每隔一个结果），可以得出如下两种滤波结果

$$L_d = \frac{1}{\sqrt{2}} \begin{pmatrix} b_{11} + b_{12} & b_{13} + b_{14} \\ b_{21} + b_{22} & b_{23} + b_{24} \\ b_{31} + b_{32} & b_{33} + b_{34} \\ b_{41} + b_{42} & b_{43} + b_{44} \end{pmatrix} \quad (6)$$

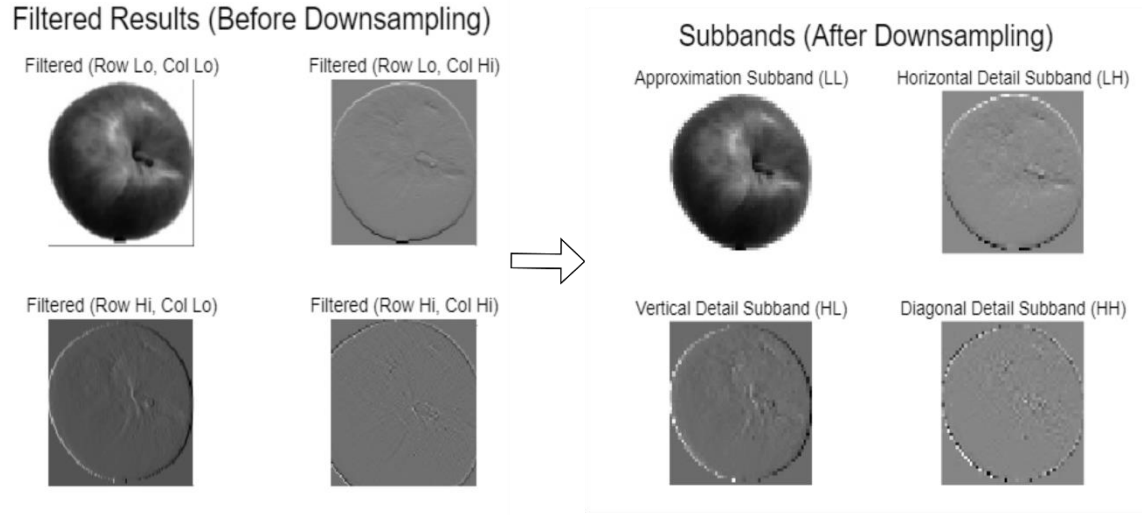
以及

$$H_d = \frac{1}{\sqrt{2}} \begin{pmatrix} b_{11} - b_{12} & b_{13} - b_{14} \\ b_{21} - b_{22} & b_{23} - b_{24} \\ b_{31} - b_{32} & b_{33} - b_{34} \\ b_{41} - b_{42} & b_{43} - b_{44} \end{pmatrix} \quad (7)$$

由此可以将水平方向上的像素信息合并，于是可以在两种水平方向滤波结果的基础上进一步对垂直方向上的像素信息进行合并处理，得出在垂直高，低通方向下的四种滤波采样结果，也就是第一次小波变换得出的四种子带 LL1, LH1, HL1, HH1。

$$\left\{ \begin{array}{l} LL1 = \frac{1}{2} \begin{pmatrix} (b_{11} + b_{12}) + (b_{21} + b_{22}) & (b_{13} + b_{14}) + (b_{23} + b_{24}) \\ (b_{31} + b_{32}) + (b_{41} + b_{42}) & (b_{33} + b_{34}) + (b_{43} + b_{44}) \end{pmatrix} \\ LH1 = \frac{1}{2} \begin{pmatrix} (b_{11} + b_{12}) - (b_{21} + b_{22}) & (b_{13} + b_{14}) - (b_{23} + b_{24}) \\ (b_{31} + b_{32}) - (b_{41} + b_{42}) & (b_{33} + b_{34}) - (b_{43} + b_{44}) \end{pmatrix} \\ HL1 = \frac{1}{2} \begin{pmatrix} (b_{11} - b_{12}) + (b_{21} - b_{22}) & (b_{13} - b_{14}) + (b_{23} - b_{24}) \\ (b_{31} - b_{32}) + (b_{41} - b_{42}) & (b_{33} - b_{34}) + (b_{43} - b_{44}) \end{pmatrix} \\ HH1 = \frac{1}{2} \begin{pmatrix} (b_{11} - b_{12}) - (b_{21} - b_{22}) & (b_{13} - b_{14}) - (b_{23} - b_{24}) \\ (b_{31} - b_{32}) - (b_{41} - b_{42}) & (b_{33} - b_{34}) - (b_{43} - b_{44}) \end{pmatrix} \end{array} \right. \quad (8)$$

由此我们便将一个 4\*4 的像素块处理成了四个子带，滤波过程以及每个子带的处理结果可视化如下



图三 滤波过程以及一重小波变换子带展示

根据图三可以看出经过第一次小波变换后，每个色道上肉眼可以识别的图形信息基本聚集在了子带 LL1 之中，也就是说，降维之后 LL1 子带能够提供的图像特征最为丰富，所以接下来的处理中只取各个 B 矩阵变换后的 LL1 子带进行下一步处理。

### 小波第二重变换

进行一次小波变换后可以将 100\*100 的图像降维到 50\*50，从图三的处理结果也可见变换后的图像纹理明显粗糙，但是仍然可以明显地区分轮廓特征，而且考虑到像素尺度长宽依然是偶数，所以本文决定，在上述 LL1 子带的基础上，再进行一次小波变换，进一步提取特征，将图像由 50\*50 进一步转换为 25\*25 的图像分析。

首先做如下变换

$$\begin{cases} A_{11} = \frac{1}{2}(b_{11} + b_{12} + b_{21} + b_{22}) \\ A_{12} = \frac{1}{2}(b_{13} + b_{14} + b_{23} + b_{24}) \\ A_{21} = \frac{1}{2}(b_{31} + b_{32} + b_{41} + b_{42}) \\ A_{22} = \frac{1}{2}(b_{33} + b_{34} + b_{43} + b_{44}) \end{cases} \quad (9)$$

于是 LL1 子带的信息可以转换表示为如下的灰度矩阵格式

$$LL1 = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \quad (10)$$

类似第一次小波变换对这个 2\*2 的灰度矩阵（像素块）进行处理，为了简化表示二维的矩阵变换，引入下述的“Haar”小波变换矩阵（这是个正交矩阵，效果与第一

次变换的因子类似，但是通过矩阵乘法可以自动实现滤波采样过程)

$$W_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (11)$$

则对 LL1 先做行变换产生中间图像，然后在中间过渡矩阵的基础上继续进行对应的列变换，具体过程依次书写如下

$$\begin{cases} [A_{11'}, A_{12'}] = [A_{11}, A_{12}] W_1^T = [A_{11}, A_{12}] \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \frac{1}{\sqrt{2}} [A_{11} + A_{12}, A_{11} - A_{12}] \\ [A_{21'}, A_{22'}] = [A_{21}, A_{22}] W_1^T = [A_{21}, A_{22}] \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \frac{1}{\sqrt{2}} [A_{21} + A_{22}, A_{21} - A_{22}] \\ \begin{pmatrix} A_{11''} \\ A_{21''} \end{pmatrix} = W_1 \begin{pmatrix} \frac{1}{\sqrt{2}} (A_{11} + A_{12}) \\ \frac{1}{\sqrt{2}} (A_{21} + A_{22}) \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} A_{11} + A_{12} \\ A_{21} + A_{22} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} (A_{11} + A_{12}) + (A_{21} + A_{22}) \\ (A_{11} + A_{12}) - (A_{21} + A_{22}) \end{pmatrix} \\ \begin{pmatrix} A_{12''} \\ A_{22''} \end{pmatrix} = W_1 \begin{pmatrix} \frac{1}{\sqrt{2}} (A_{11} - A_{12}) \\ \frac{1}{\sqrt{2}} (A_{21} - A_{22}) \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} A_{11} - A_{12} \\ A_{21} - A_{22} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} (A_{11} - A_{12}) + (A_{21} - A_{22}) \\ (A_{11} - A_{12}) - (A_{21} - A_{22}) \end{pmatrix} \end{cases}$$

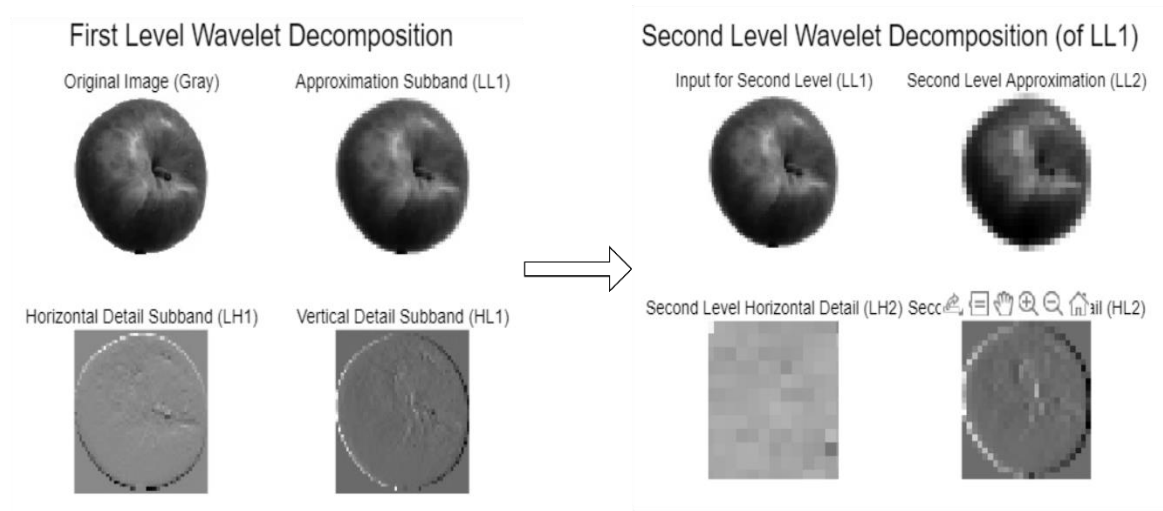
于是整理得第二重小波变换的总公式如下

$$\begin{pmatrix} \text{LL2} & \text{LH2} \\ \text{HL2} & \text{HH2} \end{pmatrix} = W_1 \text{LL1} W_1^T \quad (12)$$

其中

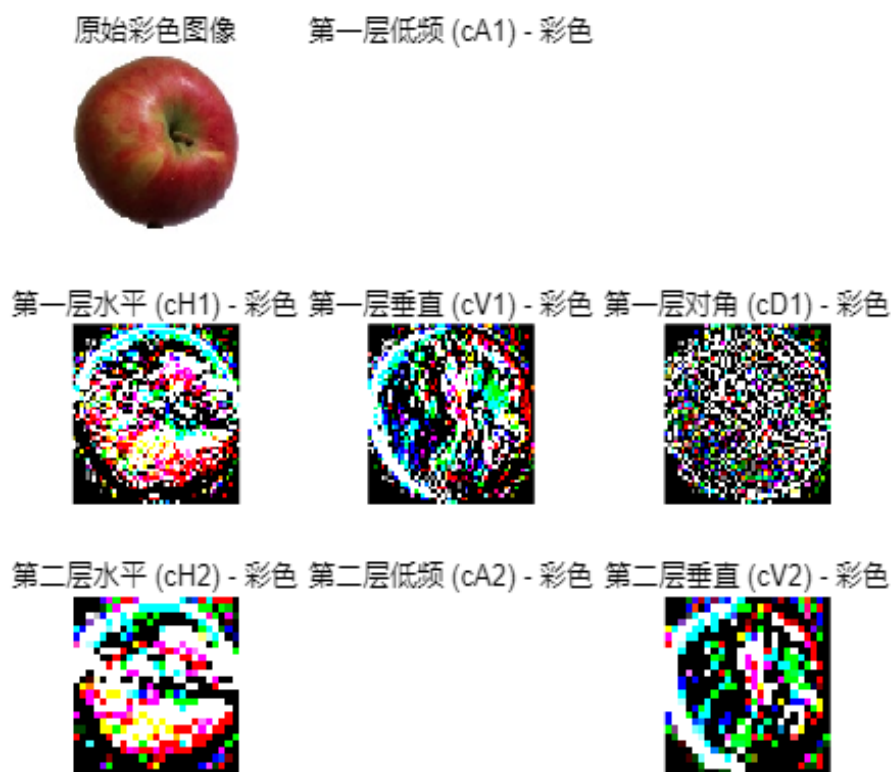
$$\begin{cases} \text{LL2} = \frac{1}{2} (A_{11} + A_{12} + A_{21} + A_{22}) \\ \text{LH2} = \frac{1}{2} (A_{11} - A_{12} + A_{21} - A_{22}) \\ \text{HL2} = \frac{1}{2} (A_{11} + A_{12} - A_{21} - A_{22}) \\ \text{HH2} = \frac{1}{2} (A_{11} - A_{12} - A_{21} + A_{22}) \end{cases} \quad (13)$$

而二重变换中的 r\_34\_100 图像具体变化如下图所示



图四 一，二重小波变换过程的前后对比

于是同理，也取低频 LL2 子带，作为最终样本，将三个色道的二重小波变换后的 LL2 子带合并得出最终的彩色图像，以此作为最终样本代入 CNN 卷积神经网络训练即可，合并过程的彩图如下所示.（其中 LL1 对应 cA1，LL2 对应 cA2）



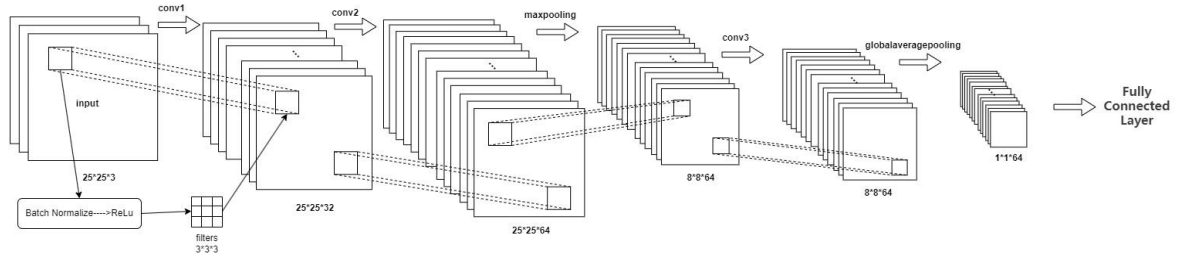
图五 三通道合并后的小波变换可视化

#### 5.1.4 基于 CNN 卷积神经网络进行图像识别与分类

在完成对图像数据的预处理之后，开始训练 CNN 卷积神经网络对处理好的图像进行识别和分析。输入层是经过二重小波变换处理过后的  $25 \times 25 \times 3$  的图像，输出层则是分别属于 45 个水果种类的概率。根据数据流类型的不同可以分为特征提取模块和全连接模块。

##### 特征提取模块

由三层卷积层和两层池化层构成，结构如图所示：



图六 本文构建 CNN 网络建构

其中滤波器的参数的初始化采用使用 He 初始化（He Initialization）方法。这是专为使用 ReLU（Rectified Linear Unit）激活函数的深度神经网络设计的一种权重初始化策略，旨在解决梯度消失或爆炸问题，提升模型训练的稳定性和收敛速度。其核心是根据输入神经元的数量（即权重矩阵的输入维度）调整初始化的方差。具体公式如下：

对于卷积层：

$$n_{in} = \text{输入通道数} \times \text{卷积核宽度} \times \text{卷积核高度} \quad (14)$$

对于全连接层：

$$n_{in} = \text{输入特征数量} \quad (15)$$

最后得到分布的标准差：

$$\sigma = \sqrt{\frac{2}{n_{in}}} \quad (16)$$

初始权重以此方差的高斯分布进行采样。并通过反向传播不断改变权重，进行学习。

每一层输出后都要经过批量归一层和 ReLu 激活层变换才能作为下一层的输入。加速训练收敛。这样不仅可以加速训练收敛，缓解梯度消失或爆炸问题，而且允许使用更大的学习率。

标准化、归一化前一层输出：

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \cdot \lambda + \beta \quad (17)$$

其中 $\lambda$ 为放缩因子， $\beta$ 为偏移量，都可作为学习参数。

激活层为 ReLu 函数：

$$Relu(x) = \max(0, x) \quad (18)$$

通过以上步骤方法，以便充分提取图像的特征，以提高模型的准确率。

### 全连接模块

该模块含有三层神经元，输入层为 64 个神经元，隐藏层为 128 个神经元，输出层为 45 个神经元，输出该图片中水果为各个品种水果的概率。通过全局平均池化层对每个通道的 8\*8 特征图取平均值，减少参数数量，然后得到 64 个参数作为全连接神经网络得输入层。

激活函数为 ReLu 函数，输入层参数为 $x$ ，对于隐藏层中的各层计算，则有模型原理知有如下输出：

隐藏层：

$$\begin{cases} z^{(1)} = W^{(1)}x + b^{(1)} \\ \alpha^{(1)} = \sigma(z^{(1)}) \end{cases} \quad (19)$$

输出层：

$$z^{(2)} = W^{(2)}\alpha^{(1)} + b^{(2)} \quad (20)$$

其中：

$W$ 为权重， $b$ 为偏置， $z$ 为线性组合， $\alpha$ 为单层输出， $\sigma$ 为激活函数

最终输出：

$$\hat{y} = \text{softmax}(z^{(2)}) \quad (21)$$

这样就可以通过多层处理转换将初始数据映射到输出空间。

计算预测值与真实标签的差异，并加入 L2 正则化防止过拟合，损失函数为：

$$L = \mathcal{L}_{CE} + \lambda \cdot \sum_{l=1}^L \|W^l\|_2^2 \quad (22)$$

其中交叉熵损失在多分类任务中可以表示为：

$$\mathcal{L}_{CE} = -\frac{1}{N} \sum_{n=1}^N \sum_{c=1}^C y_{ic} \log P_{ic} \quad (23)$$

其中， $N$  表示样本的数量， $C$  表示种类数量， $y_{ic}$  表示第  $i$  个样本的真实标签（one-

hot 编码, 如果标签为 2, 则  $y_{i2} = 1$ ,  $y_{ic} = 0 (c \neq 2)$ ,  $P_{ic}$  表示模型预测标签为  $c$  的用 SoftMax 函数输出概率。

反向传播 (Backpropagation) 是神经网络训练的核心算法, 通过链式法则 (Chain Rule) 将损失函数的梯度从输出层逐层传递到输入层, 从而计算每个参数对损失函数的梯度。在实际的训练过程中还需要考虑 Dropout 率, 因此隐藏层梯度:

$$\frac{\partial \mathcal{L}}{\partial z^{(l)}} = (W^{l+1} \cdot \frac{\partial \mathcal{L}}{\partial \alpha^{(l)}}) \odot \sigma'(z^{(l)}) \odot DropoutRate \quad (24)$$

最终损失函数对各层权重和偏置的梯度需要对每个样本的梯度求平均:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial W^{(l)}} = \frac{1}{N} \sum_{i=1}^N \frac{\partial \mathcal{L}_i}{\partial W^{(l)}} \\ \frac{\partial \mathcal{L}}{\partial b^{(l)}} = \frac{1}{N} \sum_{i=1}^N \frac{\partial \mathcal{L}_i}{\partial b^{(l)}} \end{cases} \quad (25)$$

使用优化器 (Adam) 更新参数:

$$\begin{cases} W^{(l)} \rightarrow W^{(l)} - \eta \cdot \frac{\partial \mathcal{L}}{\partial W^{(l)}} \\ b^{(l)} \rightarrow b^{(l)} - \eta \cdot \frac{\partial \mathcal{L}}{\partial b^{(l)}} \end{cases} \quad (26)$$

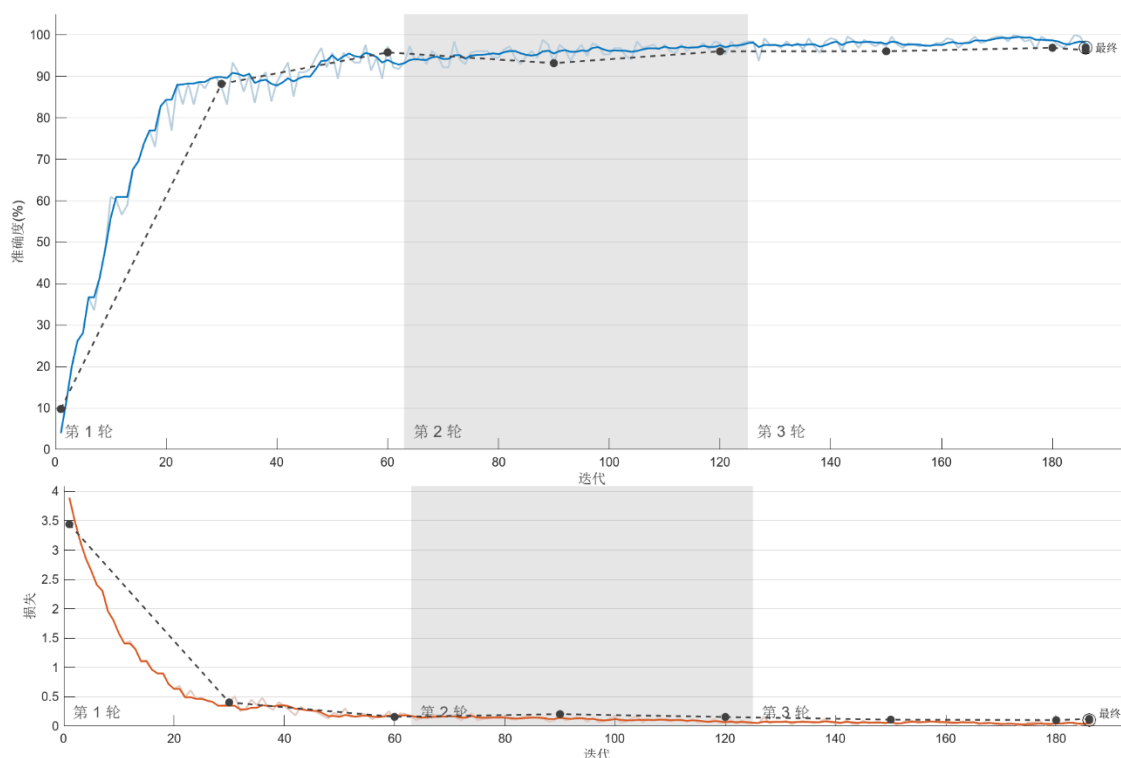
其中:

$\eta$  是学习率, 控制参数更新的进度。全连接层通过前向传播得到各品类的分类概率并根据损失函数通过反向传播不断改变权重参数不断学习。

## 训练与模型导出

基于上述两个模块构建的 CNN 卷积神经网络, 将水果数据集中的“1training”文件代入训练网络, 并利用“2validation”文件中的数据对由一所得到的权重进行验证, 获得验证曲线, 最后将训练完毕的模型导出保存为‘fruitClassifier.mat’模型, 利用‘3test’文件中的数据对其再做一次测试, 导出精确度, 检查是否存在过拟合等现象, 具体的训练过程和结果如下。





图七 本文基于水果数据集的 CNN 网络训练过程

最终在验证集与测试集上的精度分别高达 96.95%与 97.67%，总共训练时间 3min 且并没有出现明显的过拟合现象，而且，重复多次验证式训练的结果如下表所列

实验序号	训练时间	分类精度	是否过拟合	测试精度（保留整数）
1	2min40s	96.66%	否	97%
2	2min56s	97.13%	否	97%
3	3min4s	97.65%	否	98%
4	2min39s	96.88%	否	98%
5	2min53s	98.1%	是	96%
6	2min44s	96.43%	否	97%
7	3min2s	97.29%	否	97%
8	2min55s	96.44%	否	96. 5%
9	2min48s	97.31%	否	97%

表五 重复验证式实验结果列举

由上述实验结果可见，在迭代 2—3 轮的条件下，九次实验的结果相对平稳，精度均值在 97.5%上下，训练时间 3min 左右，而基于测试精度与分类精度比较得出的过拟合判断指标显示，十次实验只有一次实验 5，出现了 2%的明显过拟合，但是还算轻微，而其他的实验中几乎没有明显的此类现象，反映了本文模型的稳定性，可复现性。

## 5.2 问题一模型在实拍水果照片上的表现验证（问题二求解）

### 5.2.1 准备数据图像

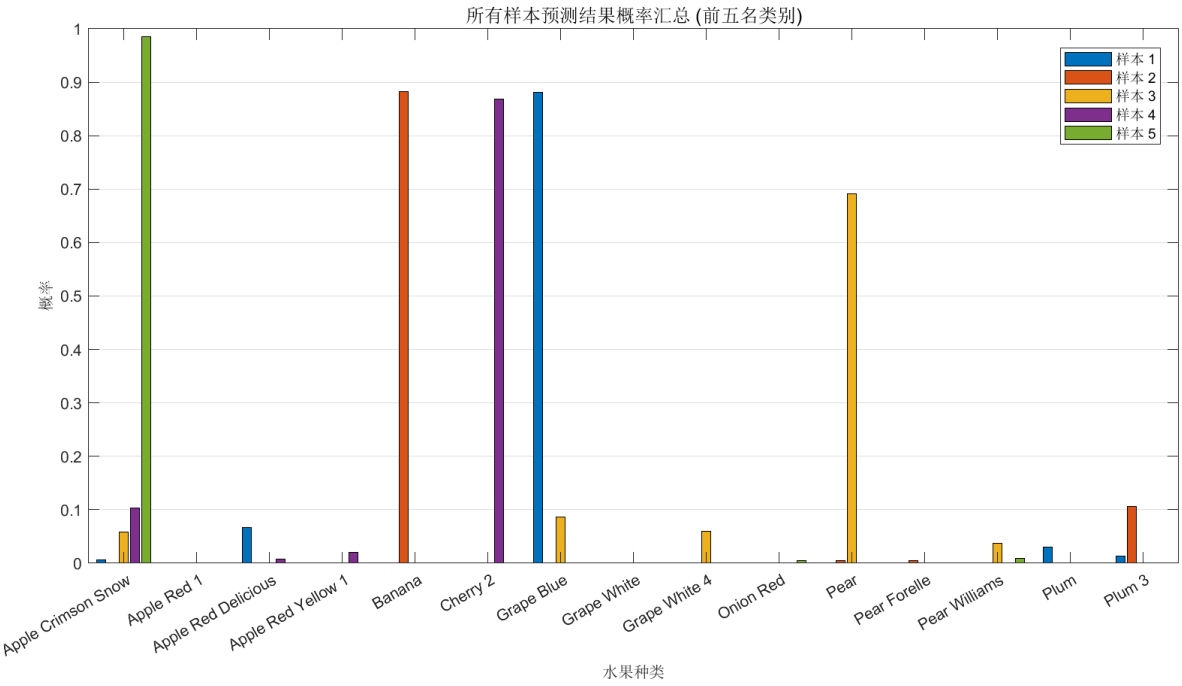
问题二的要求是利用第一题中训练好的水果识别网络来识别一些用手机拍摄的图片，从而检验模型在实际操作中的可行性，所以首先拍摄实际的图像作为实验数据备用。包括车厘子，葡萄，苹果，梨子以及香蕉的实际图像，用于实验的图像如下表所示

样本编号	图像	种类	来源	分辨率
1		葡萄	宿舍实拍	598*468
2		香蕉	宿舍实拍	862*1530
3		梨	宿舍实拍	1080*1920
4		车厘子	网上查找	298*280
5		苹果	宿舍实拍	1279*1706

表六 实物图像准备

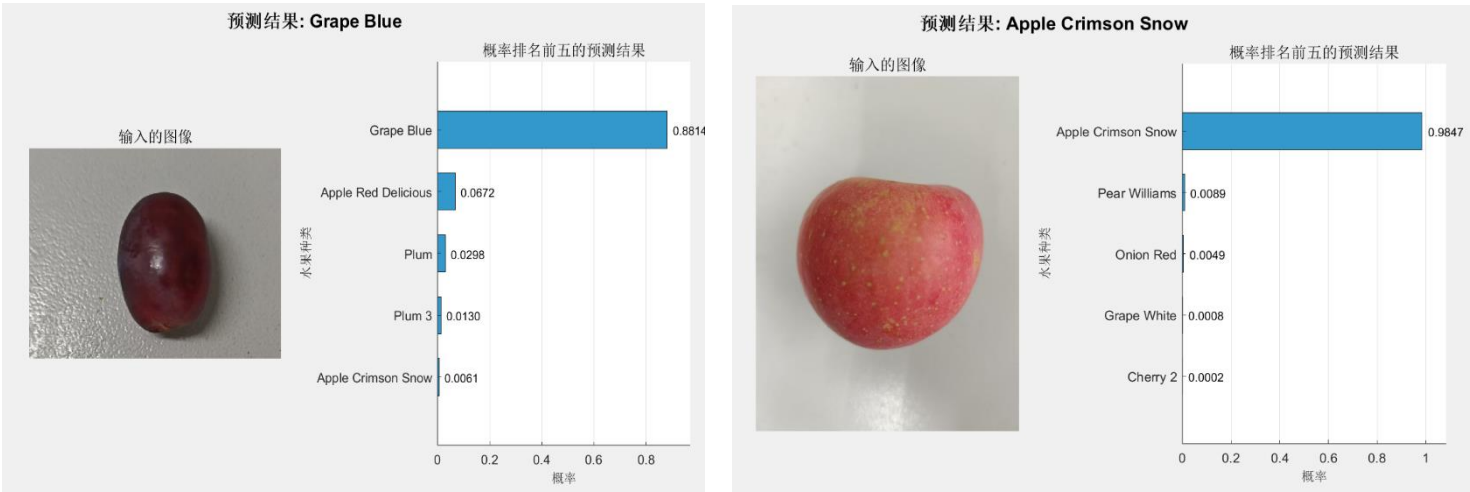
5. 2. 2 数据处理以及识别

首先，由于 CNN 网络是基于实施二重小波变换的 100\*100 图像样本训练的，所以首先将上述图像调用 MATLAB 函数 ‘imresize’ 利用多元插值方法将上述图像的分辨率放缩到 100\*100，然后均利用二重小波变换对图像进行处理，再代入 5.1.4 中训练的分类模型 “fruitClassifier.mat” 进行分类识别，可以得出下列结果



图八 所有样本预测结果概率汇总（仅显示前五名）

具体预测结果中不同种类决策的概率可由下面几张图可视化



图九 部分结果可视化

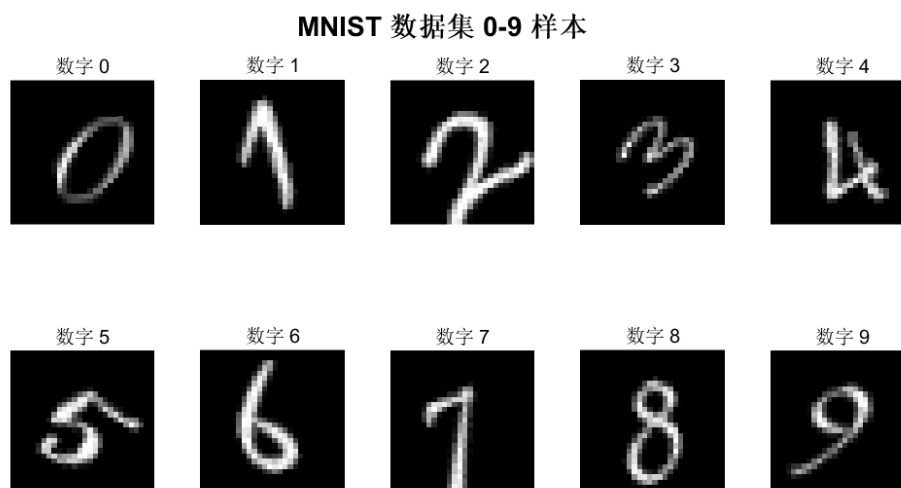
由此经过对照，易知所输入的样本验证均正确，所以反映出本模型在一定程度上能够胜任现实拍照图片中的水果识别任务，并且在本题所要求的车厘子，葡萄，苹果，

梨子以及香蕉等水果上取得了良好的实验结果。

### 5.3 问题一模型在 MNIST 手写数字集上的迁移测试（问题三求解）

#### 5.3.1 数据集处理与训练

##### 数据展示（mnist 数据集）

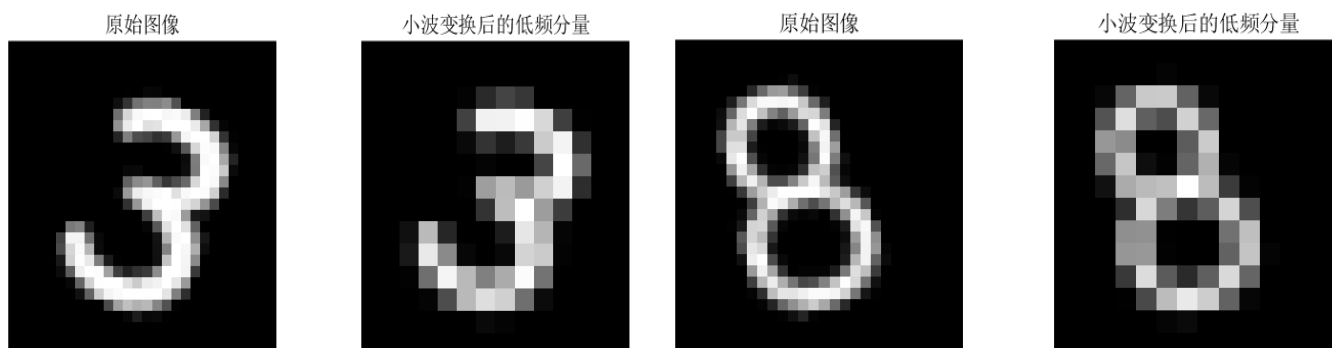


图十 MNIST 数据集部分样本可视化

##### 数据预处理（小波变换）

我们可以在数据展示中发现，MNIST 数据集中的数字图像均为黑白图，其中可以归结出与 5.1.1 中水果数据集类似的特征：不同种类数字之间形状差异明显，而且此处主要由白色区域的分布决定，在 CNN 决策中影响占比最高的因素应当是不同数字的大致形状，那么用小波变换处理也不会对数字的分辨产生显著的混淆，同时还能有效地实现图像降维降噪，理论上可以在缩短 CNN 训练时间的同时抑制过拟合。

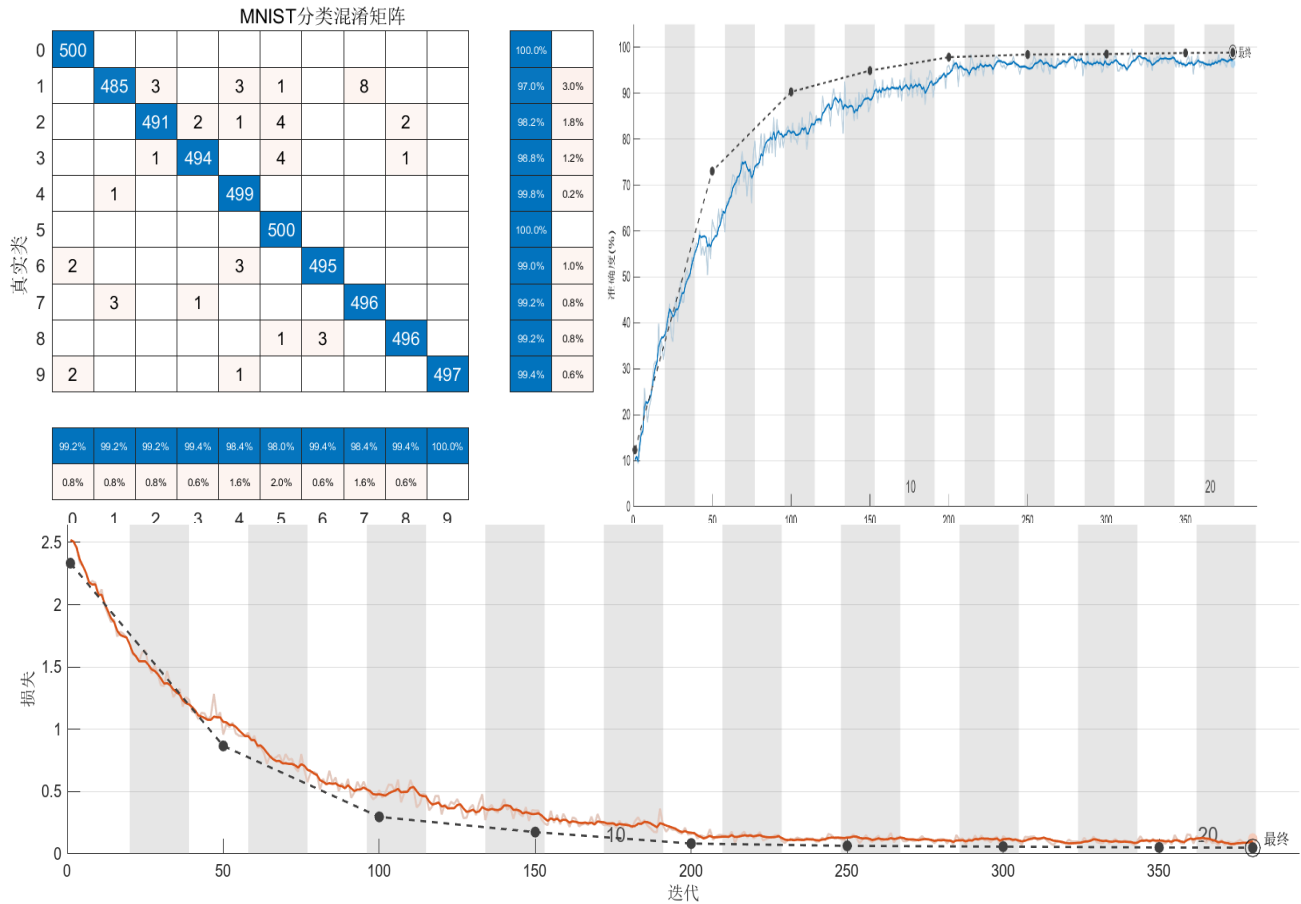
以此，我们对 MNIST 数据集的图片进行小波变换处理。



图十一 MNIST 数据集部分样本变换结果可视化

迁移测试训练

将小波变换完毕的数据集代入问题一中自定义构建的 CNN 卷积神经网络框架中训练,训练结果与分类验证结果如下图所示



图十二 迁移测试训练分类结果

所以能发现，即使在手写数据集等其他种类数据集上，本文的模型也可以发挥出极其良好的表现，甚至在图形数据分辨率由  $28 \times 28$  降维至  $7 \times 7$  后依然能够保持 98% 左右的高精度分类，而且训练速度远高于经典的预训练 CNN 架构。

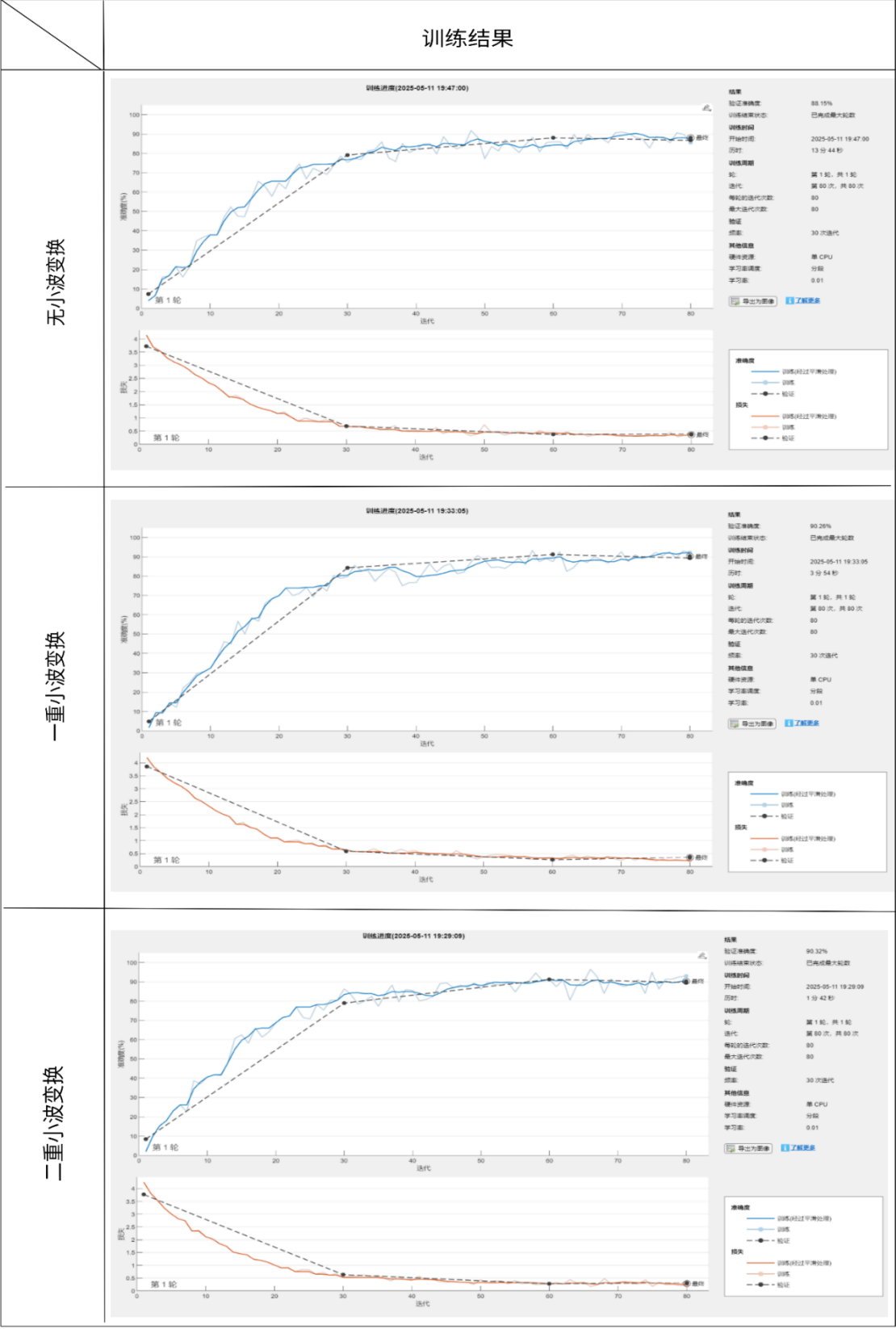
六、模型的分析与检验

基于 CNN 训练结果检验小波变换效果

根据上述部分分析可以总结出一种有效的图像分类模型范式“图像增强+小波变换+CNN 识别”，并证明了这种处理范式在水果数据集以及 MNIST 手写阿拉伯数字集上的可行性，而本部分将会在问题一的基础上拓展实验，从实际角度验证本题中（二重）小波变换处理图像的高效性。

如下图所示分别为不使用小波变换，使用一次小波变换，和使用两次小波变换条件下的“2021\_fruits\_datasets”数据集的训练过程可视化以及训练精度，时间等参数的

结果图表。（为了节省时间，只保留始终处于拟合加强状态的一轮训练作为样本，所以精度只有 90%左右）



图十三 比较实验训练过程（经训练一轮，准确率 90%上下）

由此可以总结出如下结果

小波变换次数	训练时间	分类精度	是否过拟合	样本分辨率
0	13min44s	88.15%	否	100*100
1	3min54s	90.26%	否	50*50
2	1min42s	90.32%	否	25*25

表七 训练结果对比

由此可以发现,随着小波变换次数的增加,样本的分辨率不断下降,则训练的时间也出现了显著下降,由 14 分钟左右经第一次小波变换到只需要 4 分钟,经过第二次变换后甚至只需要 2 分钟即可训练完毕,而迭代次数保持 80 次不变,说明训练的效率大幅提高,而分类精度几乎保持不变,甚至有细微的上升趋势!从而有力地证明了在水果识别等图像分类问题背景下提前小波变换处理图像的策略对提高 CNN 网络训练效率的有效性,而小波变换次数为零的条件下 CNN 较高的分类精度也反映出本文所构建的 CNN 模型在该数据集上的基础表现极其良好,但是仍需要利用小波变换进行预处理以提高权重学习效率,由此,才能方便推广到更为庞大而复杂的数据集上。

## 七、模型的评价、改进与推广

### 7.1 模型的创新点

#### 小波变换与 CNN 的结合

本研究创新性地将二重离散小波变换(DWT)与 CNN 相结合,用于图像分类任务。通过小波变换对图像进行预处理,提取低频子带特征,显著降低了输入数据的维度,减少了计算冗余,同时保留了关键的图像特征,为 CNN 的高效训练提供了优化的输入,提高了模型的训练速度和分类精度。

#### 轻量化 CNN 结构设计

设计了一种轻量化的 CNN 结构,结合批量归一化(BatchNorm)与全局平均池化层,有效减少了模型参数量,降低了模型复杂度,抑制了过拟合现象的发生。这种结构在保证分类精度的同时,显著缩短了训练时间,提高了模型的训练效率,使其更适合在资源受限的环境下进行实时图像识别。

#### 数据增强策略

采用基于 Pytorch 仿射变换算法的图像旋转处理,对水果数据集进行数据增强,生成多角度的训练样本,增强了模型对空间变化的鲁棒性。这种数据增强策略有效提高

了模型在实际场景中的分类能力，使其能够更好地应对实际拍摄图像中可能出现的各种变化，如光照不均、背景干扰、拍摄角度偏移等。

## 7.2 模型的缺点

### 小波变换的局限性

虽然小波变换在降维和特征提取方面表现出色，但它可能会丢失一些高频细节信息。在某些对细节特征要求较高的图像分类任务中，这种信息丢失可能会对分类精度产生一定影响。例如，在对纹理复杂或边缘信息丰富的图像进行分类时，仅依赖小波变换提取的低频特征可能无法完全满足分类需求。

### 轻量化 CNN 的表达能力限制

为了降低模型复杂度和计算成本，采用了轻量化的 CNN 结构，这可能会在一定程度上限制模型对复杂图像特征的学习能力。与一些深度和宽度较大的 CNN 架构相比，轻量化 CNN 可能无法捕捉到图像中更深层次的语义信息和复杂的特征组合，从而在处理一些难度较高的图像分类任务时，分类精度可能略低于更复杂的模型。



## 八、参考文献

- [1]张晓莉,张喜珍,林冬梅,等.基于 CNN-BiGRU 和多头自注意力机制的自动睡眠分期方法[J].中国医学物理学杂志,2025,42(04):496-504.
- [2]吴迪,肖衍,沈学军,等.基于改进 Res2Net 与迁移学习的水果图像分类[J].电子科技大学学报,2025,54(01):62-71.
- [3]吴慧敏,王丽,王威,等.基于小波变换的图像数据融合[J].信息技术,2025,(03):42-48.DOI:10.13274/j.cnki.hdzj.2025.03.007.
- [4]赵红伟,李朋,程振飞.基于 Faster R-CNN 图像处理的光伏并网变电站运行故障检测方法[J].电工技术,2025,(01):27-29+32.DOI:10.19768/j.cnki.dgjs.2025.01.008.
- [5]贾艳平,桑妍丽,李月茹.基于改进 Faster R-CNN 模型的水果分类识别[J].食品与机械, 2023, 39(8):129-135.

## 附录

### 附录 1（基于 MATLAB 的问题一求解代码）

```
clc,clear;
%% 阶段 1：数据加载与预处理
% 设置训练集和测试集路径
trainFolder = '2021_fruits_dataset/1training'; % 训练集路径
testFolder = '2021_fruits_dataset/2validation'; % 测试集路径
validationFolder='2021_fruits_dataset/3test'; % 验证集路径

%% 步骤 1：加载图像数据
% 使用 imageDatastore 读取训练集图像，并使用文件夹名称作为标签
trainDS = imageDatastore(trainFolder, 'IncludeSubfolders', true, 'LabelSource',
'foldernames');

% 使用 imageDatastore 读取测试集图像，并使用文件夹名称作为标签
testDS = imageDatastore(testFolder, 'IncludeSubfolders', true, 'LabelSource',
'foldernames');

validationDS = imageDatastore(validationFolder, 'IncludeSubfolders', true,
'LabelSource', 'foldernames');

% 显示训练集和测试集的基本信息
disp(['训练集图像数量: ', num2str(numel(trainDS.Files))]); % 使用 numel 获取
文件数量
```

```

disp(['验证集图像数量: ', num2str(numel(testDS.Files))]);    % 使用 numel 获取文件数量
disp(['测试集图像数量: ', num2str(numel(validationDS.Files))]);    % 使用 numel 获取文件数量

%% 步骤 2: 二重小波变换特征提取（每张图像的低频分量）
% 获取图像数据
XTrain = readall(trainDS);
XTest = readall(testDS);
Xvalidation=readall(validationDS);

% 初始化处理后的数据容器（每张图像会提取低频分量）
processedXTrain = cell(size(XTrain));
processedXTest = cell(size(XTest));
processedXvalidation = cell(size(Xvalidation));

% 对训练集的每张图像进行小波变换（包括二重小波变换）
parfor i = 1:numel(XTrain) % 使用 parfor 加速处理
    % 处理每个图像的每个颜色通道（RGB）
    img = XTrain{i}; % 获取当前图像
    processedImg = zeros(25, 25, 3, 'single'); % 初始化存储小波变换结果的容器

    for c = 1:3 % 对每个颜色通道进行小波变换
        % 第一层小波变换
        [cA1, ~, ~, ~] = dwt2(img(:, :, c), 'haar'); % 执行小波变换，提取低频分量

        % 第二层小波变换
        [cA2, ~, ~, ~] = dwt2(cA1, 'haar'); % 对第一层低频分量再次进行小波变换

        % 存储第二层低频分量
        processedImg(:, :, c) = cA2; % 只保留第二层低频分量
    end
    processedXTrain{i} = processedImg; % 存储处理后的图像
end

% 对测试集的每张图像进行二重小波变换
parfor i = 1:numel(XTest)
    img = XTest{i}; % 获取当前图像
    processedImg = zeros(25, 25, 3, 'single'); % 初始化存储小波变换结果的容器

    for c = 1:3 % 对每个颜色通道进行小波变换
        % 第一层小波变换
        [cA1, ~, ~, ~] = dwt2(img(:, :, c), 'haar'); % 执行小波变换，提取低频分

```

```

量
    % 第二层小波变换
    [cA2, ~, ~, ~] = dwt2(cA1, 'haar'); % 对第一层低频分量再次进行小波
变换
    % 存储第二层低频分量
    processedImg(:, :, c) = cA2; % 只保留第二层低频分量
end
processedXTest{i} = processedImg; % 存储处理后的图像
end

% 对验证集的每张图像进行二重小波变换
parfor i = 1:numel(Xvalidation)
    img = Xvalidation{i}; % 获取当前图像
    processedImg = zeros(25, 25, 3, 'single'); % 初始化存储小波变换结果的容
器
    for c = 1:3 % 对每个颜色通道进行小波变换
        % 第一层小波变换
        [cA1, ~, ~, ~] = dwt2(img(:, :, c), 'haar'); % 执行小波变换，提取低频分
量
        % 第二层小波变换
        [cA2, ~, ~, ~] = dwt2(cA1, 'haar'); % 对第一层低频分量再次进行小波
变换
        % 存储第二层低频分量
        processedImg(:, :, c) = cA2; % 只保留第二层低频分量
    end
    processedXvalidation{i} = processedImg; % 存储处理后的图像
end

%% 步骤 3：数据管道构建（关键修复）
% 将训练数据和标签组合成数据管道
trainLabels = categorical(trainDS.Labels); % 转换标签为分类类型
testLabels = categorical(testDS.Labels); % 转换标签为分类类型
validationLabels = (categorical(validationDS.Labels))

% 将图像数据从 cell 转换为 numeric 数组
% 转换训练集
XTrainArray = cat(4, processedXTrain{:}); % 使用 cat 函数将 cell 数组合并
为 4D 数组
XTrainArray = single(XTrainArray); % 转换为 single 类型

% 转换测试集
XTestArray = cat(4, processedXTest{:}); % 使用 cat 函数将 cell 数组合并为

```

#### 4D 数组

```
XTestArray = single(XTestArray); % 转换为 single 类型

% 创建新的数据管道
trainDS = arrayDatastore(XTrainArray, 'IterationDimension', 4); % 适应每张图像
labelDS = arrayDatastore(trainLabels); % 直接转换原始标签
combinedTrainDS = combine(trainDS, labelDS);

testDS = arrayDatastore(XTestArray, 'IterationDimension', 4); % 适应每张图像
testLabelDS = arrayDatastore(testLabels);
combinedTestDS = combine(testDS, testLabelDS);

%% 阶段 4: 优化 CNN 架构设计
layers = [
    imageInputLayer([25 25 3], 'Name','input')

    % 特征提取模块
    convolution2dLayer(3,32,'Padding','same','WeightsInitializer','he')
    batchNormalizationLayer
    reluLayer
    convolution2dLayer(3,64,'Padding','same','WeightsInitializer','he')
    batchNormalizationLayer
    reluLayer
    maxPooling2dLayer(3,'Stride',3)

    % 分类模块
    convolution2dLayer(3,64,'Padding','same','WeightsInitializer','he')
    batchNormalizationLayer
    reluLayer
    globalAveragePooling2dLayer
    dropoutLayer(0.1) % 防止过拟合

    fullyConnectedLayer(128,'WeightsInitializer','he')
    reluLayer
    fullyConnectedLayer(45)
    softmaxLayer
    classificationLayer];

%% 阶段 5: 高级训练配置
options = trainingOptions('adam',...
    'MaxEpochs', 3,...
    'MiniBatchSize', 256,...
    'ValidationData', combinedTestDS,...
    'ValidationFrequency', 30,...
```

```

        'InitialLearnRate', 0.01,...
        'LearnRateSchedule', 'piecewise',...
        'LearnRateDropFactor', 0.05,...
        'LearnRateDropPeriod', 10,...
        'Shuffle', 'every-epoch',...
        'Plots', 'training-progress',...
        'Verbose', true);
%% 步骤 6: 训练网络
net = trainNetwork(combinedTrainDS, layers, options); % 训练网络

%% 步骤 7: 使用测试集进行评估
% 确保图像的大小是 [50 50 3]
inputSize = [25 25 3]; % 网络输入的大小

% 手动调整测试集图像大小
XvalidationResized = cell(size(processedXvalidation)); % 初始化存储调整后的
图像容器
for i = 1:numel(processedXvalidation)
    XvalidationResized{i} = imresize(processedXvalidation{i},
inputSize(1:2)); % 调整图像大小
end

% 将调整后的图像转换为数值数组
XvalidationResizedArray = cat(4, XvalidationResized{:}); % 使用 cat 函数将
cell 数组合并为 4D 数组
XvalidationResizedArray = single(XvalidationResizedArray); % 转换为 single
类型

% 创建新的数据管道
validationDSResized = arrayDatastore(XvalidationResizedArray,
'IterationDimension', 4); % 适应每张图像
% 将标签与图像数据分开存储
validationLabelsArray = validationLabels; % 存储验证集的标签

% 使用测试集进行预测
[predLabels, scores] = classify(net, validationDSResized);

% 计算准确率
accuracy = mean(predLabels == validationLabelsArray); % 使用单独的标签数组
来计算准确率
disp(['模型准确率: ', num2str(accuracy * 100, '%.2f'), '%']);

% 混淆矩阵可视化
figure

```

```

confusionchart(validationLabelsArray, predLabels, ...
    'Title', '水果分类混淆矩阵', ...
    'RowSummary', 'row-normalized', ...
    'ColumnSummary', 'column-normalized');

```

## 附录 2（基于 MATLAB 求解第二问的代码之主程序）

介绍：问题二代码由分类函数与主程序两部分组成

```

clc;
clear;
close all; % 关闭所有打开的 figure 窗口

% 定义实验样本信息
samples = {
    struct('ID', 1, 'Image', '葡萄.jpg', 'Kind', '葡萄', 'Source', '宿舍实拍', 'Resolution',
    '598*468');
    struct('ID', 2, 'Image', '香蕉.jpg', 'Kind', '香蕉', 'Source', '宿舍实拍', 'Resolution',
    '862*1530');
    struct('ID', 3, 'Image', '梨.jpg', 'Kind', '梨', 'Source', '宿舍实拍', 'Resolution',
    '1080*19020'); % 注意分辨率可能写错了
    struct('ID', 4, 'Image', '车厘子.jpg', 'Kind', '车厘子', 'Source', '网上查找', 'Resolution',
    '298*280');
    struct('ID', 5, 'Image', '苹果.jpg', 'Kind', '苹果', 'Source', '宿舍实拍', 'Resolution',
    '1279*1706');
};

% 初始化存储预测结果的容器
numSamples = length(samples);
allTop5Scores = cell(numSamples, 1);
allTop5Labels = cell(numSamples, 1);
allPredictedLabels = cell(numSamples, 1);

% 循环处理每个样本
for i = 1:numSamples
    imagePath = samples{i}.Image;
    sampleID = samples{i}.ID;
    sampleKind = samples{i}.Kind;

    disp(['--- 处理样本 ', num2str(sampleID), ': ', imagePath, ' ---']);

```

```

% 调用 predictFruit 函数 (它会绘制单个样本的图)
[predictedLabel, top5Scores, top5Labels] = predictFruit(imagePath);

% 调试输出, 检查返回值的类型和内容
disp(['样本 ', num2str(sampleID), ' 返回的 top5Labels 类型: ',
class(top5Labels)]);
disp(['样本 ', num2str(sampleID), ' 返回的 top5Scores 类型: ',
class(top5Scores)]);
disp(['样本 ', num2str(sampleID), ' 返回的 top5Labels 内容示例:']);
disp(top5Labels);
disp(['样本 ', num2str(sampleID), ' 返回的 top5Scores 内容示例:']);
disp(top5Scores);

% 存储结果
allPredictedLabels{i} = char(predictedLabel);
allTop5Scores{i} = top5Scores;
allTop5Labels{i} = top5Labels; % 存储前五名的类别名称
end

% --- 绘制总的大柱状图 ---

% 准备绘制分组柱状图的数据
% 收集所有样本的前五名类别, 并找到所有出现过的唯一类别
allUniqueTop5Labels = unique(vertcat(allTop5Labels{:})); % 联合所有样本的前五名
类别

% 创建一个矩阵来存储每个样本在所有唯一前五名类别上的概率
% 行代表唯一类别, 列代表样本
numUniqueLabels = length(allUniqueTop5Labels);
groupedScores = zeros(numUniqueLabels, numSamples);

% 填充 groupedScores 矩阵
for i = 1:numSamples
    currentTop5Labels = allTop5Labels{i};
    currentTop5Scores = allTop5Scores{i};
    for j = 1:length(currentTop5Labels)
        % 找到当前类别在所有唯一类别列表中的索引
        labelIndex = find(strcmp(allUniqueTop5Labels, currentTop5Labels{j}));
        if ~isempty(labelIndex)
            groupedScores(labelIndex, i) = currentTop5Scores(j);
        end
    end
end
end
end

```

```

% 绘制分组柱状图
figure('Name','所有样本预测结果概率汇总'); % 创建新的 figure
h = bar(groupedScores, 'grouped'); % 绘制分组柱状图

% 设置 x 轴刻度和标签为唯一类别名称
xticks(1:numUniqueLabels);
xticklabels(allUniqueTop5Labels);

% 添加图例，显示每个样本的编号
legend(cellfun(@(x) ['样本 ', num2str(x.ID)], samples, 'UniformOutput', false),
'Location', 'best');

% 设置标题和轴标签
title('所有样本预测结果概率汇总 (前五名类别)');
xlabel('水果种类');
ylabel('概率');

% 调整字体大小
set(gca, 'FontSize', 10);
title('所有样本预测结果概率汇总 (前五名类别)', 'FontSize', 12);
xlabel('水果种类', 'FontSize', 10);
ylabel('概率', 'FontSize', 10);

% 添加网格线
grid on;
set(gca, 'YGrid', 'on', 'XGrid', 'off'); % 只显示 Y 轴网格线

% 调整 figure 大小
set(gcf, 'Position', [100, 100, 1200, 600]); % 调整 figure 大小

% --- 总大柱状图绘制结束 ---

% --- predictFruit 函数定义 ---

function [predictedLabel, top5Scores, top5Labels] = predictFruit(imagePath)
% 载入保存的训练模型
loadedNet = load('fruitClassifier.mat');
net = loadedNet.net; % 这是保存的训练好的网络

% 获取网络的类别名称
classNames = net.Layers(end).Classes; % 通常分类网络的最后一层是分类层，其
Classes 属性包含类别名称

% 读取输入图像

```



```

img = imread(imagePath);

% 调整图像大小为 100x100 以与第二张图片一致
inputSize = [100 100]; % 期望的输入尺寸
imgResized = imresize(img, inputSize); % 将图像调整为 100x100

% 对图像进行二重小波变换
processedImg = zeros(25, 25, 3, 'single'); % 初始化存储小波变换结果的容器
for c = 1:3 % 对每个颜色通道进行小波变换
    % 确保通道数据是 uint8 类型, dwt2 对 uint8 效果更好且输出类型一致
    channelImg = imgResized(:, :, c);
    if ~isa(channelImg, 'uint8')
        channelImg = im2uint8(channelImg);
    end

    % 第一层小波变换
    [cA1, ~, ~, ~] = dwt2(channelImg, 'haar'); % 执行小波变换, 提取低频分量
    % 第二层小波变换
    [cA2, ~, ~, ~] = dwt2(cA1, 'haar'); % 对第一层低频分量再次进行小波变换

    % 存储第二层低频分量
    processedImg(:, :, c) = cA2; % 只保留第二层低频分量
end

% 将处理后的图像转换为 single 类型
processedImg = single(processedImg);

% 扩展图像的维度为 4D 数组, 以便网络处理
processedImg = reshape(processedImg, [25, 25, 3, 1]); % 将图像转换为 4D 数组

% 使用训练好的模型进行预测
[predictedLabel, scores] = classify(net, processedImg);

% 输出预测结果
disp(['预测的水果类别: ', char(predictedLabel)]);

% --- 新增部分: 列出概率排名前五的水果种类 ---

% 将 scores 转换为向量
scores = scores(:);

% 对 scores 和对应的类别名称进行排序
[sortedScores, sortedIndices] = sort(scores, 'descend');

```

```

% 获取排名前五的概率和对应的类别名称
top5Scores = sortedScores(1:min(5, end)); % 取前 5 个，如果类别总数少于 5，则取全部
top5Labels = cellstr(classNames(sortedIndices(1:min(5, end)))); % 转换为字符串元胞数组

% 显示排名前五的结果
disp('概率排名前五的水果种类: ');
for i = 1:length(top5Labels)
    fprintf('%d. %s (概率: %.4f)\n', i, top5Labels{i}, top5Scores(i));
end

% --- 绘制单个样本的可视化图 (美化版柱状图) ---

figure('Name', ['预测结果: ', char(predictedLabel)]); % 创建新的 figure 窗口，标题包含预测类别

% 左边子图：显示输入的原始图像
subplot(1, 2, 1); % 1 行 2 列的第一个子图
imshow(img); % 显示原始图像 (不是调整大小后的，更直观)
title('输入的图像', 'FontSize', 12); % 设置标题，调整字体大小

% 右边子图：绘制概率排名前五的横向柱状图
subplot(1, 2, 2); % 1 行 2 列的第二个子图

% 绘制横向柱状图，设置柱子宽度和颜色
hBar = barh(top5Scores, 0.6, 'FaceColor', [0.2 0.6 0.8]); % 0.6 是柱子宽度，颜色为蓝色

% 设置 y 轴刻度和标签为水果类别名称
yticks(1:length(top5Labels)); % 设置 y 轴刻度位置
yticklabels(top5Labels); % 设置 y 轴刻度标签为类别名称

% 反转 y 轴，使概率最高的类别在顶部显示
set(gca, 'YDir', 'reverse');

% 设置 x 轴范围，确保从 0 开始，并留出一些空间给标签
xlim([0, max(top5Scores) * 1.1]); % 将 x 轴最大值设置为最高概率的 1.1 倍

% 添加网格线
grid on; % 添加网格线
set(gca, 'XGrid', 'on', 'YGrid', 'off'); % 只显示 X 轴网格线

% 调整字体大小

```

```

set(gca, 'FontSize', 10); % 调整轴标签和刻度的字体大小
title('概率排名前五的预测结果', 'FontSize', 12); % 设置标题，调整字体大小
xlabel('概率', 'FontSize', 10); % 设置 x 轴标签，调整字体大小
ylabel('水果种类', 'FontSize', 10); % 设置 y 轴标签，调整字体大小

% 添加概率数值标签到柱状图上
% 调整标签位置，稍微向右偏移
for i = 1:length(top5Scores)
    text(top5Scores(i) + 0.01 * max(top5Scores), i, sprintf(' %.4f', top5Scores(i)), ...
        'VerticalAlignment', 'middle', 'HorizontalAlignment', 'left', 'FontSize', 9);
end

% 移除不必要的边框
box off; % 移除图表边框

% 调整子图之间的间距
sgtitle(['预测结果: ', char(predictedLabel)], 'FontSize', 14, 'FontWeight', 'bold'); % 设置
整个 figure 的标题
set(gcf, 'Position', [100, 100, 1000, 500]); % 调整 figure 大小以便更好地显示

% --- 新增部分结束 ---

% 返回预测结果、前五名概率和类别
end

```

### 附录 3（基于 MATLAB 求解第二问之分类函数）

```

function [predictedLabel, top5Scores, top5Labels] = predictFruit(imagePath)
% 载入保存的训练模型
loadedNet = load('fruitClassifier.mat');
net = loadedNet.net; % 这是保存的训练好的网络

% 获取网络的类别名称
classNames = net.Layers(end).Classes; % 通常分类网络的最后一层是分类层，其
Classes 属性包含类别名称

```

```

% 读取输入图像
img = imread(imagePath);

% 调整图像大小为 100x100 以与第二张图片一致
inputSize = [100 100]; % 期望的输入尺寸
imgResized = imresize(img, inputSize); % 将图像调整为 100x100

% 对图像进行二重小波变换
processedImg = zeros(25, 25, 3, 'single'); % 初始化存储小波变换结果的容器
for c = 1:3 % 对每个颜色通道进行小波变换
    % 确保通道数据是 uint8 类型, dwt2 对 uint8 效果更好且输出类型一致
    channelImg = imgResized(:, :, c);
    if ~isa(channelImg, 'uint8')
        channelImg = im2uint8(channelImg);
    end

    % 第一层小波变换
    [cA1, ~, ~, ~] = dwt2(channelImg, 'haar'); % 执行小波变换, 提取低频分量
    % 第二层小波变换
    [cA2, ~, ~, ~] = dwt2(cA1, 'haar'); % 对第一层低频分量再次进行小波变换

    % 存储第二层低频分量
    processedImg(:, :, c) = cA2; % 只保留第二层低频分量
end

% 将处理后的图像转换为 single 类型
processedImg = single(processedImg);

% 扩展图像的维度为 4D 数组, 以便网络处理
processedImg = reshape(processedImg, [25, 25, 3, 1]); % 将图像转换为 4D 数组

% 使用训练好的模型进行预测
[predictedLabel, scores] = classify(net, processedImg);

% 输出预测结果
disp(['预测的水果类别: ', char(predictedLabel)]);

% --- 新增部分: 列出概率排名前五的水果种类 ---

% 将 scores 转换为向量
scores = scores(:);

% 对 scores 和对应的类别名称进行排序
[sortedScores, sortedIndices] = sort(scores, 'descend');

```

```

% 获取排名前五的概率和对应的类别名称
top5Scores = sortedScores(1:min(5, end)); % 取前 5 个，如果类别总数少于 5，则取全部
top5Labels = classNames(sortedIndices(1:min(5, end)));

% 显示排名前五的结果
disp('概率排名前五的水果种类: ');
for i = 1:length(top5Labels)
    fprintf('%d. %s (概率: %.4f)\n', i, char(top5Labels(i)), top5Scores(i));
end

% --- 绘制单个样本的可视化图 (美化版柱状图) ---

figure('Name', ['预测结果: ', char(predictedLabel)]); % 创建新的 figure 窗口，标题包含预测类别

% 左边子图：显示输入的原始图像
subplot(1, 2, 1); % 1 行 2 列的第一个子图
imshow(img); % 显示原始图像 (不是调整大小后的，更直观)
title('输入的图像', 'FontSize', 12); % 设置标题，调整字体大小

% 右边子图：绘制概率排名前五的横向柱状图
subplot(1, 2, 2); % 1 行 2 列的第二个子图

% 绘制横向柱状图，设置柱子宽度和颜色
hBar = barh(top5Scores, 0.6, 'FaceColor', [0.2 0.6 0.8]); % 0.6 是柱子宽度，颜色为蓝色

% 设置 y 轴刻度和标签为水果类别名称
yticks(1:length(top5Labels)); % 设置 y 轴刻度位置
yticklabels(top5Labels); % 设置 y 轴刻度标签为类别名称

% 反转 y 轴，使概率最高的类别在顶部显示
set(gca, 'YDir', 'reverse');

% 设置 x 轴范围，确保从 0 开始，并留出一些空间给标签
xlim([0, max(top5Scores) * 1.1]); % 将 x 轴最大值设置为最高概率的 1.1 倍

% 添加网格线
grid on; % 添加网格线
set(gca, 'XGrid', 'on', 'YGrid', 'off'); % 只显示 X 轴网格线

% 调整字体大小

```

```

set(gca, 'FontSize', 10); % 调整轴标签和刻度的字体大小
title('概率排名前五的预测结果', 'FontSize', 12); % 设置标题，调整字体大小
xlabel('概率', 'FontSize', 10); % 设置 x 轴标签，调整字体大小
ylabel('水果种类', 'FontSize', 10); % 设置 y 轴标签，调整字体大小

% 添加概率数值标签到柱状图上
% 调整标签位置，稍微向右偏移
for i = 1:length(top5Scores)
    text(top5Scores(i) + 0.01 * max(top5Scores), i, sprintf(' %.4f', top5Scores(i)), ...
        'VerticalAlignment', 'middle', 'HorizontalAlignment', 'left', 'FontSize', 9);
end

% 移除不必要的边框
box off; % 移除图表边框

% 调整子图之间的间距
sgtitle(['预测结果: ', char(predictedLabel)], 'FontSize', 14, 'FontWeight', 'bold'); % 设置
整个 figure 的标题
set(gcf, 'Position', [100, 100, 1000, 500]); % 调整 figure 大小以便更好地显示

% --- 新增部分结束 ---

% 返回预测结果、前五名概率和类别
end

```

#### 附录 4（基于 MATLAB 求解第三问代码）

```

%% 阶段 1：数据加载与预处理
% 加载官方 MNIST 数据集（自动处理标签格式）
[XTrain, YTrain] = digitTrain4DArrayData(); % 60000 个训练样本
[XTest, YTest] = digitTest4DArrayData(); % 10000 个测试样本

% 像素归一化（[0,255]→[0,1]）
XTrain = single(XTrain)/255;
XTest = single(XTest)/255;

%% 阶段 2：小波特征提取（14x14 低频分量）
% 初始化处理后的数据容器

```

```

processedXTrain = zeros(14,14,1,size(XTrain,4), 'single');
processedXTest = zeros(14,14,1,size(XTest,4), 'single');

% 训练集小波变换
parfor i = 1:size(XTrain,4) % 并行加速
[cA, ~, ~, ~] = dwt2(XTrain(:, :, 1, i), 'haar');
processedXTrain(:, :, 1, i) = cA;
end

% 测试集小波变换
parfor i = 1:size(XTest,4)
[cA, ~, ~, ~] = dwt2(XTest(:, :, 1, i), 'haar');
processedXTest(:, :, 1, i) = cA;
end

%% 阶段 3: 数据管道构建（关键修复）
% 创建 ArrayDatastore（适配 MATLAB 最新 API）
trainDS = arrayDatastore(processedXTrain, 'IterationDimension',4);
labelDS = arrayDatastore(categorical(YTrain)); % 直接转换原始标签
combinedTrainDS = combine(trainDS, labelDS);

% 验证数据集
testDS = arrayDatastore(processedXTest, 'IterationDimension',4);
testLabelDS = arrayDatastore(categorical(YTest));
combinedTestDS = combine(testDS, testLabelDS);

%% 阶段 4: 优化 CNN 架构设计
layers = [
    imageInputLayer([14 14 1], 'Name','input')

    % 特征提取模块
    convolution2dLayer(3,16,'Padding','same','WeightsInitializer','he')
    batchNormalizationLayer
    reluLayer
    convolution2dLayer(3,32,'Padding','same','WeightsInitializer','he')
    batchNormalizationLayer
    reluLayer
    maxPooling2dLayer(2,'Stride',2)

    % 分类模块
    convolution2dLayer(3,64,'Padding','same','WeightsInitializer','he')
    batchNormalizationLayer
    reluLayer
    globalAveragePooling2dLayer

```

```

dropoutLayer(0.5) % 防止过拟合

fullyConnectedLayer(128,'WeightsInitializer','he')
reluLayer
fullyConnectedLayer(10)
softmaxLayer
classificationLayer];

%% 阶段 5: 高级训练配置
options = trainingOptions('adam',...
    'MaxEpochs', 20,...
    'MiniBatchSize', 256,...
    'ValidationData', combinedTestDS,...
    'ValidationFrequency', 50,...
    'InitialLearnRate', 0.01,...
    'LearnRateSchedule', 'piecewise',...
    'LearnRateDropFactor', 0.1,...
    'LearnRateDropPeriod', 10,...
    'Shuffle', 'every-epoch',...
    'Plots', 'training-progress',...
    'Verbose', true);

%% 阶段 6: 模型训练与验证
net = trainNetwork(combinedTrainDS, layers, options);

% 最终评估
[predLabels, scores] = classify(net, processedXTest);
accuracy = mean(predLabels == categorical(YTest));
disp(['模型准确率: ', num2str(accuracy*100, '%.2f'), '%']);

% 混淆矩阵可视化
figure
confusionchart(YTest, predLabels, ...
    'Title', 'MNIST 分类混淆矩阵',...
    'RowSummary', 'row-normalized',...
    'ColumnSummary', 'column-normalized');
% 随机选择一个图像索引
randIndex = randi([1, size(XTest, 4)]);

% 获取原始图像（未做小波变换）
originalImage = XTest(:,1,randIndex);

% 进行小波变换
[cA, ~, ~, ~] = dwt2(originalImage, 'haar');

```



```
% 创建一个并排显示变换前后图像的图形
figure;

% 显示原始图像
subplot(1, 2, 1);
imshow(originalImage, []);
title('原始图像');

% 显示小波变换后的低频分量 (cA)
subplot(1, 2, 2);
imshow(cA, []);
title('小波变换后的低频分量');
```