补操作：添加了对数据进行归一化层、卷积层和池化层；定义训练次数、对加载的图片进行处理。

**流程：**

**1、定义数据加载函数：**

下载5种图片分别保存到文件中，加载数据用image\_dataset\_from\_directory从指定的目录中加载图像数据集，validation\_split定义20%用于验证集，指定加载数据集类型为training，设置随机生成器种子为123，指定图像大小，指定批次大小

**2、调用函数加载下载到文件中的数据集**

**3、使用Rescaling做归一化数据处理**

**4、定义模型：**

归一化，卷积层使用64个输出通道，卷积核3\*3，采用边界填充(padding='same')的方式， relu作为激活函数, 添加池化层用于降低特征图的空间维度，通过选取局部区域中的最大值来实现；添加Flatten层将二维的输出转化为一维，添加两个密集层，一个有128个神经元，使用relu作为激活函数，最后一层5个神经元，使用softmax作为激活函数。

**5、编译模型**

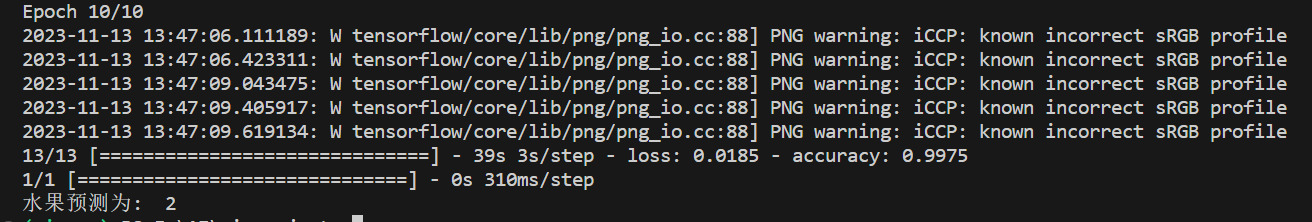
使用Adam优化器，使用稀疏分类交叉熵损失函数来计算模型的损失值，使用准确率作为评估指标来评估模型的性能。设置训练模型次数。

**6、加载待检图片**

使用load\_img函数加载图像文件，通过指定target\_size参数，将图像调整尺寸为300,200；

img\_to\_array函数将加载的图像转换为Numpy数组表示，使用expand\_dims将输入数组的维度从(300, 200, 3)扩展为(1, 300, 200, 3), 扩展张量为了满足模型对输入数据的批次要求，就需要将单个样本转换为一个大小为1的批次

**7、最后进行预测**



**代码：**

import tensorflow as tf

from tensorflow import keras

from tensorflow.python.keras import layers

import numpy as np

# 定义数据加载函数

def load\_dataset(directory):

    train\_ds = tf.keras.preprocessing.image\_dataset\_from\_directory(

        directory,

        validation\_split=0.2,

        subset='training',

        seed=123,

        image\_size=(300, 200),

        batch\_size=32

    )

    return train\_ds

# 加载数据集

train\_ds = load\_dataset('E:/AI/训练集材料/datatest/train')

# 归一化数据

normalization\_layer = tf.keras.layers.Rescaling(1. / 255)

# 定义模型

model = tf.keras.models.Sequential([

    normalization\_layer,

    tf.keras.layers.Conv2D(64,(3,3),padding='same',activation='relu'),

    tf.keras.layers.MaxPooling2D(),

    tf.keras.layers.Flatten(input\_shape=(300, 200, 3)), #Flatten 层不接受归一化，所以归一化单独处理

    tf.keras.layers.Dense(128, activation='relu'),

    tf.keras.layers.Dense(5, activation='softmax')

])

# 编译模型

model.compile(optimizer='adam',

              loss=tf.keras.losses.SparseCategoricalCrossentropy(),

              metrics=['accuracy'])

# 训练模型次数

epochs = 10

model.fit(train\_ds, epochs=epochs)

# 加载待检测图片:E:/AI/ai\_project/data/02/imag/banana\_27.jpg  ,E:/AI/训练集材料/datatest/test/beetroot/Image\_10.jpg

image = tf.keras.preprocessing.image.load\_img('E:/AI/训练集材料/datatest/test/beetroot/Image\_10.jpg', target\_size=(300, 200))

input\_array = tf.keras.preprocessing.image.img\_to\_array(image)

input\_array = tf.expand\_dims(input\_array, 0)

# 进行预测

predictions = model.predict(input\_array)

predicted\_class = np.argmax(predictions[0], axis=-1)

print("水果预测为: ", predicted\_class)