**实验（实习）报告**

1. **实验意义及目的**

卷积神经网络是深度学习的基本工具。本实验要求学生掌握常用神经网络算法原理，能够利用Matlab创建和训练简单的卷积神经网络来进行深度学习分类。

1. **实验内容**

创建简单的深度学习网络用于图像分类。

1. **实验过程**

1.加载和浏览图像数据。

2.定义网络架构。

3.指定训练选项。

4.训练网络。

5.预测新数据的标签并计算分类准确度。

1. **实验说明与代码**
2. 加载和浏览图像数据

% 加载和浏览图像数据

digitDatasetPath = fullfile(matlabroot,'toolbox','nnet','nndemos', ...

    'nndatasets','DigitDataset');

imds = imageDatastore(digitDatasetPath, ...

    'IncludeSubfolders',true,'LabelSource','foldernames');

1. 显示数据存储中的部分图像。

figure;

perm = randperm(10000,20);

for i = 1:20

    subplot(4,5,i);

    imshow(imds.Files{perm(i)});

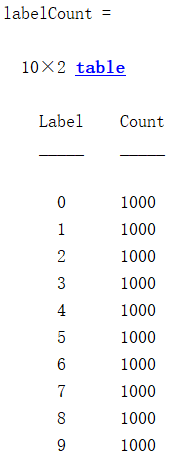
End



1. 计算每个类别中的图像数量。

labelCount 是一个表，其中列出了标签，以及每个标签对应的图像数量。数据存储包含数字 0-9 的总共 10000 个图像，每个数字对应 1000 个图像。您可以在网络的最后一个全连接层中指定类数作为 OutputSize 参数。

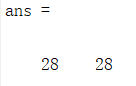
labelCount = countEachLabel(imds);



1. 检查 digitData 中第一个图像的大小。每个图像的大小均为 28×28×1 像素。

img = readimage(imds,1);

size(img)



1. 指定训练集和验证集

将数据划分为训练数据集和验证数据集，以使训练集中的每个类别包含 750 个图像，并且验证集包含对应每个标签的其余图像。splitEachLabel 将数据存储 digitData 拆分为两个新的数据存储 trainDigitData 和 valDigitData。

numTrainFiles = 750;

[imdsTrain,imdsValidation] = splitEachLabel(imds,numTrainFiles,'randomize');

1. 定义网络架构

定义卷积神经网络架构。

图像输入层 imageInputLayer 用于指定图像大小，在本例中为 28×28×1。这些数字对应于高度、宽度和通道大小。数字数据由灰度图像组成，因此通道大小（颜色通道）为 1。对于彩色图像，通道大小为 3，对应于 RGB 值。您不需要打乱数据，因为 trainNetwork 默认会在训练开始时打乱数据。trainNetwork 还可以在训练过程中的每轮训练开始时自动打乱数据。

卷积层在卷积层中，第一个参数是 filterSize，它是训练函数在沿图像扫描时使用的滤波器的高度和宽度。在此示例中，数字 3 表示滤波器大小为 3×3。您可以为滤波器的高度和宽度指定不同大小。第二个参数是滤波器数量 numFilters，它是连接到同一输入区域的神经元数量。此参数决定了特征图的数量。使用 'Padding' 名称-值对组对输入特征图进行填充。对于默认步幅为 1 的卷积层，'same' 填充可确保空间输出大小与输入大小相同。您也可以使用 convolution2dLayer 名称-值对组参数定义该层的步幅和学习率。

批量归一化层批量归一化层对网络中的激活值和梯度传播进行归一化，使网络训练成为更简单的优化问题。在卷积层和非线性部分（例如 ReLU 层）之间使用批量归一化层，来加速网络训练并降低对网络初始化的敏感度。使用 batchNormalizationLayer 创建批量归一化层。

ReLU 层批量归一化层后接一个非线性激活函数。最常见的激活函数是修正线性单元 (ReLU)。使用 reluLayer 创建 ReLU 层。

最大池化层卷积层（带激活函数）有时会后跟下采样操作，以减小特征图的空间大小并删除冗余空间信息。通过下采样可以增加更深卷积层中的滤波器数量，而不会增加每层所需的计算量。下采样的一种方法是使用最大池化，可使用 maxPooling2dLayer 创建。最大池化层返回由第一个参数 poolSize 指定的矩形输入区域的最大值。在此示例中，该矩形区域的大小是 [2,2]。'Stride' 名称-值对组参数指定训练函数在沿输入扫描时所采用的步长。

全连接层卷积层和下采样层后跟一个或多个全连接层。顾名思义，全连接层中的神经元将连接到前一层中的所有神经元。该层将先前层在图像中学习的所有特征组合在一起，以识别较大的模式。最后一个全连接层将特征组合在一起来对图像进行分类。因此，最后一个全连接层中的 OutputSize 参数等于目标数据中的类数。在此示例中，输出大小为 10，对应于 10 个类。使用 fullyConnectedLayer 创建全连接层。

softmax 层 softmax 激活函数对全连接层的输出进行归一化。softmax 层的输出由总和为 1 的多个正数组成，这些数字随后可被分类层用作分类概率。使用 softmaxLayer 函数在最后一个全连接层后创建一个 softmax 层。

分类层最终层是分类层。该层使用 softmax 激活函数针对每个输入返回的概率，将输入分配到其中一个互斥类并计算损失。要创建分类层，请使用 classificationLayer。

layers = [

    imageInputLayer([28 28 1])

    convolution2dLayer(3,8,'Padding','same')

    batchNormalizationLayer

    reluLayer

    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(3,16,'Padding','same')

    batchNormalizationLayer

    reluLayer

    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(3,32,'Padding','same')

    batchNormalizationLayer

    reluLayer

    fullyConnectedLayer(10)

    softmaxLayer

    classificationLayer

];

1. 指定训练选项

定义网络结构体后，指定训练选项。使用具有动量的随机梯度下降 (SGDM) 训练网络，初始学习率为 0.01。将最大训练轮数设置为 4。一轮训练是对整个训练数据集的一个完整训练周期。通过指定验证数据和验证频率，监控训练过程中的网络准确度。每轮训练都会打乱数据。软件基于训练数据训练网络，并在训练过程中按固定时间间隔计算基于验证数据的准确度。验证数据不用于更新网络权重。打开训练进度图，关闭命令行窗口输出。

options = trainingOptions('sgdm', ...

    'InitialLearnRate',0.01, ...

    'MaxEpochs',4, ...

    'Shuffle','every-epoch', ...

    'ValidationData',imdsValidation, ...

    'ValidationFrequency',30, ...

    'Verbose',false, ...

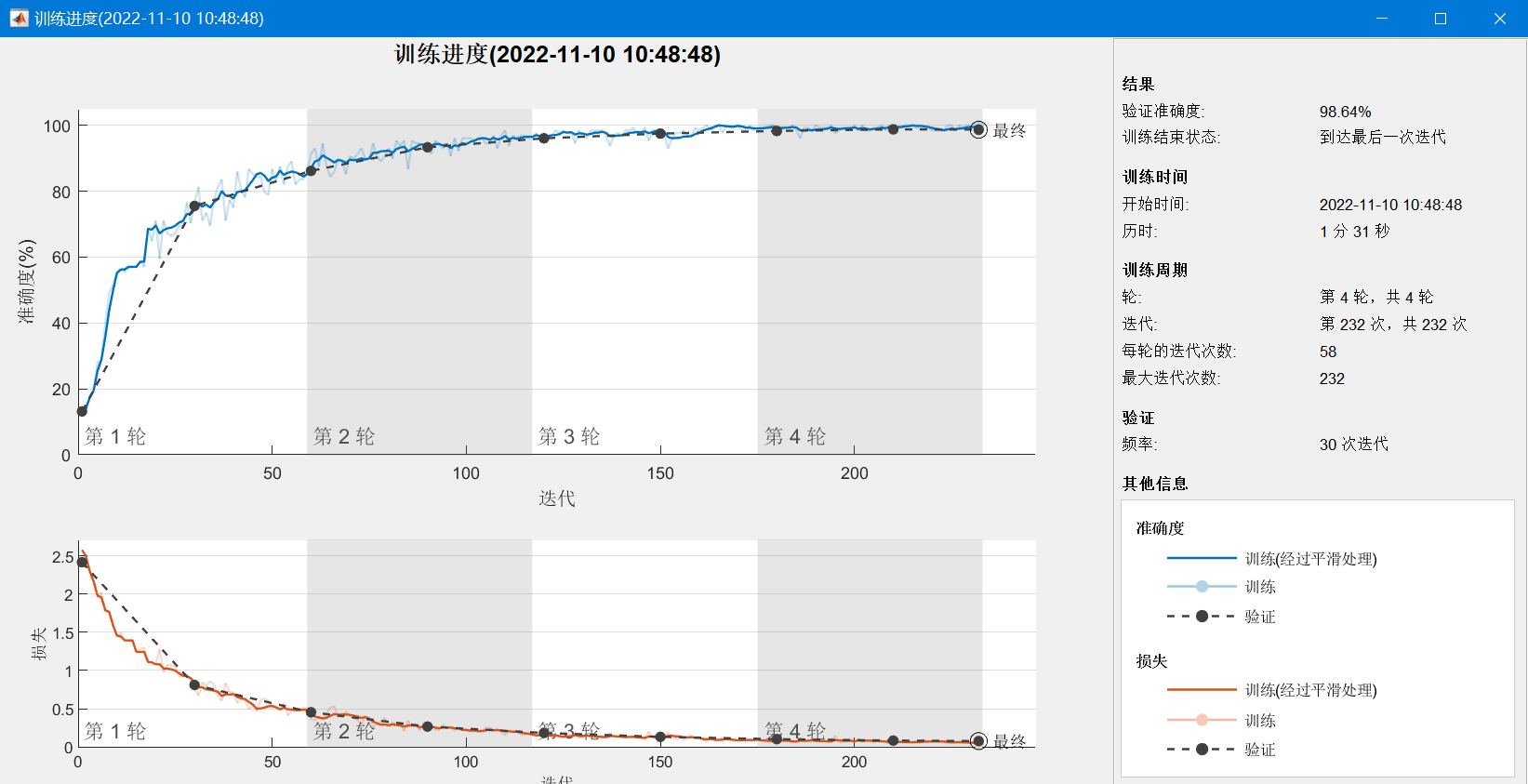
    'Plots','training-progress');

1. 使用训练数据训练网络

使用 layers 定义的架构、训练数据和训练选项训练网络。默认情况下，trainNetwork 使用 GPU（如果有），否则使用 CPU。在 GPU 上训练需要 Parallel Computing Toolbox™ 和支持的 GPU 设备。有关受支持设备的信息，请参阅GPU Computing Requirements (Parallel Computing Toolbox)。您还可以使用 trainingOptions 的 'ExecutionEnvironment' 名称-值对组参数指定执行环境。

训练进度图显示了小批量损失和准确度以及验证损失和准确度。有关训练进度图的详细信息，请参阅监控深度学习训练进度。损失是交叉熵损失。准确度是网络分类正确的图像的百分比。

net = trainNetwork(imdsTrain,layers,options);



1. 对验证图像进行分类并计算准确度

使用经过训练的网络预测验证数据的标签，并计算最终验证准确度。准确度是网络预测正确的标签的比例。在本例中，超过 98% 的预测标签与验证集的真实标签相匹配。

YPred = classify(net,imdsValidation);

YValidation = imdsValidation.Labels;

accuracy = sum(YPred == YValidation)/numel(YValidation)

