# 期末综合实验报告

学号：

姓名：***Steven***

实验名称：基于人脸识别的分类器设计

实验内容：使用MATLAB编程环境，采用ORL数据集（也可包括其他人脸识别数据集），设计分类器进行人脸识别实验。

实验结果：

本次实验仅使用ORL数据集，并按照预先划分的训练集(60%)和测试集(40%)进行训练和验证。

图表

描述已自动生成本次实验使用MATLAB自带的Monitor deep learning training-progress工具实时预览分类器的准确度变化。当训练结束后，该分类器在测试集上准确度达到96.25%。

问题回答：

1. 请问在实验过程中，是直接设计分类器来进行训练和测试，还是先进行特征提取(或特征选择)，再设计分类器来进行训练和测试？

答：先特征提取再设计分类器进行训练&测试

1. 如果进行了特征提取(或特征选择)，请简述所采用的特征提取(或特征选择)算法的工作原理。如果没有进行特征提取(或特征选择)，回答“无”即可。

答：特征提取的方法为卷积。卷积运算是让卷积核在矩阵上逐步滑动，将覆盖的子矩阵与卷积核进行乘积和求和，得到一个新的值。这个新的像素值就是卷积核在这个位置提取到的特征值。通过多层的卷积运算，可以逐步提取整幅矩阵中的高维特征。

1. 实验中设计的是什么分类器(给出分类器的名称)？请简述该分类器的工作原理。

答：

该分类器使用了残差神经网络(ResNet)作为主要的判别方法。

ResNet是一种基于CNN的神经网络结构，相较于常见的CNN“卷积-池化-全连接”结构，ResNet在每个卷积块后面添加一个跳过一层或多层的直连通道(shortcut)，解决了CNN层数较深的情况下退化和难以训练的问题。

ResNet本质仍然是CNN，其工作原理为：采用ResNet构建深度卷积神经网络，提取人脸的特征，并根据不同人的人脸特征对测试集和训练集进行预测，根据预测结果计算每轮的精度。

附实验代码：

本次实验仅有一个主程序，如下所示：

|  |
| --- |
| close all;  clear all;  clc;  %----------------------------------- 读取数据 -----------------------------------%  load ORL\_trainset; load ORL\_testset; load ORL\_testlabel; % 载入训练集和测试集  [~, trainnum] = size(train\_data);  [~, testnum] = size(test\_data); % trainnum 为训练集样本数，testnum 为测试集样本数  classnum = length(unique(train\_label)); % 类别数  % 根据resnetLayers的参数格式需求，将训练数据转换为四维数组，将标签转换为category数组  train\_data = reshape(train\_data, [32 32 1 trainnum]); % 4个维度分别是：高度、宽度、通道数、图像数  train\_label = categorical(train\_label);  test\_data = reshape(test\_data, [32 32 1 testnum]);  test\_label = categorical(label\_truth);  %----------------------------------- 读取数据 -----------------------------------%  %---------------------------------- 数据预处理 ----------------------------------%  imageSize = [32 32 1]; % 图像大小  pixelRange = [-4 4]; % 图像平移范围  % 数据增强，用于增加训练集样本数，防止过拟合；实现方式是对训练集进行随机平移和随机翻转  imageAugmenter = imageDataAugmenter( ...      RandXReflection = true, ...      RandXTranslation = pixelRange, ...      RandYTranslation = pixelRange);  % 将数据增强后的训练集转换为augmentedImageDatastore格式  augimdsTrain = augmentedImageDatastore(imageSize, train\_data, train\_label, ...      DataAugmentation = imageAugmenter, ...      OutputSizeMode = "randcrop");  %---------------------------------- 数据预处理 ----------------------------------%  %------------------------------- 定义网络结构和参数 -------------------------------%  % 定义网络结构：卷积核初始大小为3，初始步长为1，初始数量为16；堆栈深度为[4 3 2]；卷积核数量依次为[16 32 64]  lgraph = resnetLayers(imageSize, classnum, ...      InitialFilterSize = 3, ...      InitialNumFilters = 16, ...      InitialStride = 1, ...      InitialPoolingLayer = "none", ...      StackDepth = [4 3 2], ...      NumFilters = [16 32 64]);  plot(lgraph); % 展示网络结构  % 设置参数：batchSize为48，初始学习率为0.1，最大迭代次数为100，每次迭代后随机打乱训练集，每隔valFrequency次迭代后在测试集上验证一次  miniBatchSize = 48;  valFrequency = floor(size(train\_data, 4) / miniBatchSize);  options = trainingOptions("sgdm", ...      InitialLearnRate = 0.1, ...      MaxEpochs = 100, ...      MiniBatchSize = miniBatchSize, ...      Shuffle = "every-epoch", ...      Plots = "training-progress", ...      Verbose = false, ...      VerboseFrequency = valFrequency, ...      ValidationData = {test\_data, test\_label}, ...      ValidationFrequency = valFrequency, ...      LearnRateSchedule = "piecewise", ... % 分段学习率      LearnRateDropFactor = 0.1, ... % 学习率下降因子      LearnRateDropPeriod = 60); % 学习率下降周期  %------------------------------- 定义网络结构和参数 -------------------------------%  %----------------------------- 训练分类器并计算准确度 -----------------------------%  net = trainNetwork(augimdsTrain, lgraph, options); % 训练分类器  [test\_pred, probs] = classify(net, test\_data); % 测试集预测结果  test\_accuracy = mean(test\_pred == test\_label); % 测试集分类准确度  train\_pred = classify(net, train\_data); % 训练集预测结果  train\_accuracy = mean(train\_pred == train\_label); % 训练集分类准确度  disp("Training accuracy: " + train\_accuracy \* 100 + "%")  disp("Validation accuracy: " + test\_accuracy \* 100 + "%")  %----------------------------- 训练分类器并计算准确度 -----------------------------% |