《人工智能与模式识别》

实 验 报 告 书

班级： 计科 2002

学号： 202801108

姓名： 李蔚

指导教师： 翟婷婷

2022-2023学年 第 二 学期

**实验名称： 神经网络算法的实现和应用**

**实验时间： 2023 年 5 月 19 日 第 14 周 星期 五**

一、实验目的

1. 熟悉BP神经网络的原理，会推导出BP算法对网络参数的更新公式。

2. 在Matlab中编程实现BP神经网络算法，并运用其解决手写数字识别问题。

二、实验预习（预备知识的问题及回答）

1. 一个已经训练好的BP神经网络模型对待识别的实例***x***的输出为向量，则网络应该将***x***预测为哪一类？ (0.5分)

**解答：**在一个已经训练好的BP神经网络模型中，输出向量y=(y\_1, y\_2, ⋯, y\_c)表示模型对待识别实例x的预测结果，其中c表示类别的数量。每个y\_i表示模型对应类别i的置信度或概率。通常情况下，我们会选择具有最高置信度或概率的类别作为网络对实例x的预测。

即，我们选择 y\_i 最大的索引 i 作为预测的类别。具体步骤如下：

1. 计算输出向量y中最大值的索引，记为max\_index。
2. 将max\_index作为类别的标签，即网络预测将x归属于类别max\_index。

三、实验内容（包含实验所用命令或相关程序源代码）

mnist手写数字图像数据集，其中包含了0到9的手写数字的灰度图像，每个数字都包含大量的不同风格书写的该数字的图像。手写数字图像识别的任务是：给定任意一张手写数字图像，能够精确识别出该图像所对应的数字是什么。显然，手写数字图像识别问题是个多分类问题。本实验中用到的mnist\_exp数据集是mnist数据集的一个子集。

1. 利用mnist\_exp数据集训练出一个BP神经网络模型，用来对手写数字图像进行分类，并评估训练出的模型的测试准确率。

1. 要求使用**图像的像素矩阵**作为每一张图像的特征，来构建训练数据集和测试数据集。要求每个数字文件夹中的第i张图片，如果满足i对10取余<7，则将其作为训练图片，否则将其作为测试图片。注意也要构建训练数据集和测试数据集的标记向量。

提示：使用imageDatastore函数一次性读取所有图片的路径名。

**Matlab代码：**（0.5分）

% 读取所有图片的路径名

imageDir = 'D:\mnist\_exp\mnist\_exp';

imageData = imageDatastore(imageDir, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');

% 初始化训练数据集和测试数据集

trainData = [];

trainLabels = [];

testData = [];

testLabels = [];

% 遍历所有图片

for i = 1:length(imageData.Files)

% 读取图片

img = readimage(imageData, i);

% 将图像像素矩阵转换为 double 类型

img = im2double(img);

% 将图像像素矩阵拉伸为行向量

imgVector = reshape(img, 1, []);

% 获取图像对应的标签

label = imageData.Labels(i);

% 判断图片是作为训练图片还是测试图片

if mod(i, 10) < 7

% 将图片添加到训练数据集

trainData = [trainData; imgVector];

trainLabels = [trainLabels; label];

else

% 将图片添加到测试数据集

testData = [testData; imgVector];

testLabels = [testLabels; label];

end

end

% 将标签向量转换为独热编码形式

trainLabels = dummyvar(double(trainLabels));

testLabels = dummyvar(double(testLabels));

% 保存训练和测试数据以及标签

save('Data.mat', 'trainData', 'testData', 'trainLabels', 'testLabels');

clear; % 清除工作区变量

clc; % 清空命令行

load('Data.mat');% 加载数据

1. 编写只包含单个隐层的BP神经网络的训练算法，将该训练算法写成一个函数：

function [network] = BP\_train(epoch, neuros, eta, data, label)，其中各参数含义如下：

epoch：遍历整个训练数据集的次数； neuros：隐层神经元的个数

eta：学习步长; data是训练神经网络使用的训练数据集，label是训练数据集的标签向量

Network：是一个结构体，将训练得到的所有网络参数存储在该结构体中，作为返回值

**Matlab代码：**（8分）

% Sigmoid函数

function y = sigmoid(x)

y = 1 ./ (1 + exp(-x));

end

% Sigmoid函数导数

function y = sigmoid\_derivative(x)

y = sigmoid(x) .\* (1 - sigmoid(x));

end

% BP神经网络训练函数

function [network] = BP\_train(epoch, neuros, eta, data, label)

input\_size = size(data, 2); % 输入层大小

output\_size = 10; % 输出层大小

% 初始化权重和偏置

W1 = rand(neuros, input\_size);

b1 = rand(neuros, 1);

W2 = rand(output\_size, neuros);

b2 = rand(output\_size, 1);

% 开始训练

for e = 1:epoch

% 打乱训练集顺序

random\_order = randperm(size(data, 1));

data = data(random\_order, :);

label = label(random\_order, :);

for i = 1:size(data, 1)

% 前向传播

x = data(i, :)';

y = label(i,:)';

z1 = W1 \* x + b1;

a1 = sigmoid(z1);

z2 = W2 \* a1 + b2;

a2 = sigmoid(z2);

% 反向传播

delta2 = (a2 - y) .\* sigmoid\_derivative(z2);

delta1 = (W2' \* delta2) .\* sigmoid\_derivative(z1);

% 更新权重和偏置

W2 = W2 - eta \* delta2 \* a1';

b2 = b2 - eta \* delta2;

W1 = W1 - eta \* delta1 \* x';

b1 = b1 - eta \* delta1;

end

end

% 存储网络参数

network.W1 = W1;

network.b1 = b1;

network.W2 = W2;

network.b2 = b2;

end

1. 编写使用训练好的神经网络进行预测的算法，将该算法写成一个函数：

function [pred\_label, pred\_y] = BP\_predict(network, X)，其中各参数含义如下：

network: 训练好的神经网络 X：是测试数据集（不带标签）

pred\_label: 是一个向量，其中，第i个元素表示对第i个测试实例预测得到的标签

pred\_y：是一个矩阵，第i 行表示网络对第i个测试实例的输出向量

**Matlab代码：**（3分）

% BP神经网络预测函数

function [pred\_label, pred\_y] = BP\_predict(network, X)

W1 = network.W1;

b1 = network.b1;

W2 = network.W2;

b2 = network.b2;

pred\_label = zeros(size(X, 1), 1);

pred\_y = zeros(size(X, 1), size(W2, 1));

for i = 1:size(X, 1)

x = X(i, :)';

z1 = W1 \* x + b1;

a1 = sigmoid(z1);

z2 = W2 \* a1 + b2;

a2 = softmax(z2);

[~, idx] = max(a2);

pred\_label(i) = idx - 1;

pred\_y(i, :) = a2';

end

end

1. 调用上述函数，在(1)得到的训练数据集上训练一个BP神经网络，并计算该网络在测试数据集上的测试准确率。要求：

①先对训练数据运用PCA进行降维，保存转换矩阵Q，然后在降维后的数据上训练神经网络；在测试时，也要对数据进行相同的PCA变换，即利用转换矩阵Q，对测试数据进行降维，然后再进行预测。

②函数BP\_train函数的参数设置为：epoch=500 , neuros=40 , eta = 0.1。

**Matlab代码和运行截图：**（3分）

% 调用PCA进行降维

[coeff] = pca(trainData);

Q = coeff(:, 1:20);

trainData\_pca = trainData \* Q;

testData\_pca = testData \* Q;

% 参数设置

epoch = 500;

neuros = 40;

eta = 0.1;

% 训练BP神经网络

network = BP\_train(epoch, neuros, eta, trainData\_pca, trainLabels);

% 在测试集上进行预测

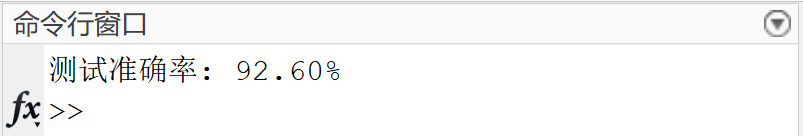
[pred\_label, pred\_y] = BP\_predict(network, testData\_pca);

% 计算测试准确率

trueLabels = vec2ind(testLabels')'-1;

accuracy = sum(pred\_label == trueLabels) / length(trueLabels);

fprintf('测试准确率: %.2f%%\n', accuracy\*100);

****