昇腾创新实践课

鸢尾花分类实验手册



华为技术有限公司

|  |
| --- |
| 版权所有 © 华为技术有限公司 2021。 保留一切权利。  非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。  商标声明  C:\Users\jwx341670\Desktop\华为标志 Huawei Logo 2018\竖版标志Vertical Version\PNG\HW_POS_RBG_Vertical-150ppi.png和其他华为商标均为华为技术有限公司的商标。  本文档提及的其他所有商标或注册商标，由各自的所有人拥有。  注意  您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。  由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。 |

|  |  |
| --- | --- |
| 华为技术有限公司 | |
| 地址： | 深圳市龙岗区坂田华为总部办公楼 邮编：518129 |
| 网址： | http://[e](http://e.huawei.com/).huawei.com |

目录

[1 实验环境介绍 2](#_Toc67586345)

[1.1 实验介绍 2](#_Toc67586346)

[1.1.1 关于本实验 2](#_Toc67586347)

[1.1.2 实验环境介绍 2](#_Toc67586348)

[2 鸢尾花分类实验 3](#_Toc67586349)

[2.1 实验介绍 3](#_Toc67586350)

[2.1.1 关于本实验 3](#_Toc67586351)

[2.1.2 实验目的 3](#_Toc67586352)

[2.1.3 背景知识 3](#_Toc67586353)

[2.1.4 实验设计 4](#_Toc67586354)

[2.2 实验过程 4](#_Toc67586355)

[2.2.1 环境导入 4](#_Toc67586356)

[2.2.2 加载数据集 4](#_Toc67586357)

[2.2.3 定义网络 5](#_Toc67586358)

[2.2.4 模型训练 6](#_Toc67586359)

[2.2.5 模型评估与预测 7](#_Toc67586360)

[2.3 实验总结 7](#_Toc67586361)

[2.4 思考题 7](#_Toc67586362)

# 实验环境介绍

## 实验介绍

### 关于本实验

本实验将使用MindSpore深度学习框架，使用鸢尾花数据集，搭建简单的全连接神经网络，完成鸢尾花种类分类任务。

### 实验环境介绍

实验环境介绍

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实验 | 实验介绍 | 难度 | 软件环境 | 开发环境 |
| 鸢尾花分类实验 | 介绍MindSpore在真实数据集上的应用，使用鸢尾花数据集搭建简单的神经网络； | 简单 | Python3.9 MindSpore2.2  Numpy  Scikit-learn | PC机或者ModelArts |

# 鸢尾花分类实验

## 实验介绍

### 关于本实验

本实验将使用MindSpore深度学习框架，使用鸢尾花数据集，搭建简单的全连接神经网络，完成鸢尾花种类分类任务。

鸢尾属约有300个品种，本实验将对下列3个品种进行分类：

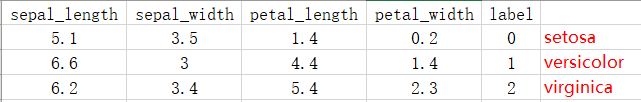
setosa

versicolor

virginica

数据集包含4个特征：sepal\_length、sepal\_width、petal\_length、petal\_width

标签中0代表setosa、1代表versicolor、2代表virginica



### 实验目的

理解MindSpore开发基本流程。

理解MindSpore基础模块的功能。

掌握MindSpore的简单操作。

### 背景知识

神经网络知识：数据集载入，数据集划分，数据集预处理，全连接神经网络，激活函数，损失函数，优化器，评估方法。

### 实验设计

1.环境导入

2.数据集加载

3.全连接网络搭建

4.模型训练

5.模型评估

## 实验过程

### 环境导入

MindSpore安装

请参考下面手册完成本地或者华为云MindSpore环境搭建



导入相关库

导入MindSpore及其他库，代码：

import mindspore.dataset as ds # 载入数据集

import mindspore.nn as nn # 网络相关

from mindspore.nn.loss import SoftmaxCrossEntropyWithLogits # 损失函数

from mindspore.nn.metrics import Accuracy # 评估矩阵

from mindspore import Model

import os # 文件路径处理

import numpy as np # numpy

from sklearn.model\_selection import train\_test\_split # 数据集划分

### 加载数据集

读取数据集

首先，我们用numpy读取csv格式的数据。

代码：

from numpy import genfromtxt

iris\_data = genfromtxt('iris.csv', delimiter=',')

print(iris\_data[:10]) # 查看前10笔数据

输出：

[[nan nan nan nan nan]

[5.1 3.5 1.4 0.2 0. ]

[4.9 3. 1.4 0.2 0. ]

[4.7 3.2 1.3 0.2 0. ]

[4.6 3.1 1.5 0.2 0. ]

[5. 3.6 1.4 0.2 0. ]

[5.4 3.9 1.7 0.4 0. ]

[4.6 3.4 1.4 0.3 0. ]

[5. 3.4 1.5 0.2 0. ]

[4.9 3.1 1.5 0.2 0. ]]

[5.9 3. 5.1 1.8 2. ]]

数据预处理

我们需要对iris\_data做简单处理，并划分数据集。

iris\_data中的第1行不需要，前4列是特征，最后1列是标签。

代码：

iris\_data = iris\_data[1:] # 移除第一行

X = iris\_data[:,:4].astype(np.float32) # 特征

y = iris\_data[:,-1].astype(np.int32) # 标签

# 数据归一化

X /= np.max(np.abs(X),axis=0)

# 划分数据集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

转换数据格式

最后，我们将数据转换成MindSpore的Dataset格式。

代码：

# 训练集

train\_data = (X\_train, y\_train)

train\_data = ds.NumpySlicesDataset(train\_data)

# 测试集

test\_data = (X\_test, y\_test)

test\_data = ds.NumpySlicesDataset(test\_data)

# 批处理

train\_data = train\_data.batch(32)

test\_data = test\_data.batch(32)

### 定义网络

完成数据处理后，我们开始定义网络。

这里的网络包含输入层、1个隐藏层和输出层，激活函数选择ReLu。



代码：

class my\_net(nn.Cell):

# 定义算子

def \_\_init\_\_(self):

super(my\_net, self).\_\_init\_\_()

self.fc1 = nn.Dense(4, 10) # 全连接层

self.fc2 = nn.Dense(10, 3) # 全连接层

self.relu = nn.ReLU() # 激活函数

# 建构网络

def construct(self, x):

x = self.fc1(x)

x = self.relu(x)

x = self.fc2(x)

return x

### 模型训练

构建模型

定义好网络后，我们接着构建模型。

除了网络，我们的模型还需要损失函数和优化器。

代码：

# 网络

net = my\_net()

# 损失函数

net\_loss = SoftmaxCrossEntropyWithLogits(sparse=True) # sparse，输出不是one hot编码时设为Ture

# 优化器

lr = 0.01 # 学习率

momentum = 0.9 # 动量

net\_opt = nn.Momentum(net.trainable\_params(), lr, momentum)

# 模型

model = Model(net, net\_loss, net\_opt, metrics={"accuracy": Accuracy()})

训练模型

模型建立好后，就能开始训练。

这里设置10次迭代。

如果看到dataset sink mode的警告可以忽略，dataset sink mode将在后续介绍。

代码：

model.train(10, train\_data)

### 模型评估与预测

最后，我们用测试集评估模型的准确率。

代码：

model.eval(test\_data)

输出：

{'accuracy': 0.956}

## 实验总结

本实验介绍了MindSpore在真实数据集上的应用，使用鸢尾花数据集搭建简单的神经网络，让学员熟悉MindSpore的基础用法，掌握MindSpore开发的简单流程。