昇腾创新实践课

MNIST手写体识别实验手册



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](#_Toc94123063)

[1.1 实验介绍 2](#_Toc94123064)

[1.1.1 关于本实验 2](#_Toc94123065)

[1.1.2 实验环境介绍 2](#_Toc94123066)

[2 MNIST手写体识别实验 3](#_Toc94123067)

[2.1 实验介绍 3](#_Toc94123068)

[2.1.1 关于本实验 3](#_Toc94123069)

[2.1.2 实验目的 3](#_Toc94123070)

[2.1.3 背景知识 4](#_Toc94123071)

[2.1.4 实验设计 4](#_Toc94123072)

[2.2 实验过程 4](#_Toc94123073)

[2.2.1 环境准备 4](#_Toc94123074)

[2.2.2 数据处理 5](#_Toc94123075)

[2.2.3 网络定义 6](#_Toc94123076)

[2.2.4 模型训练 8](#_Toc94123077)

[2.2.5 模型评估 8](#_Toc94123078)

[2.2.6 效果展示 9](#_Toc94123079)

[2.2.7 模型保存 9](#_Toc94123080)

[2.2.8 Web预测体验 9](#_Toc94123081)

[2.3 实验总结 12](#_Toc94123082)

[2.4 思考题 12](#_Toc94123083)

# 实验环境介绍

## 实验介绍

### 关于本实验

本实验使用MindSpore深度学习框架，进行网络搭建、数据处理、网络训练和测试，完成MNIST手写体识别任务。

### 实验环境介绍

实验、介绍、难度、软件环境、硬件环境：

实验环境介绍

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实验 | 实验介绍 | 难度 | 软件环境 | 开发环境 |
| MNIST手写体识别实验 | 基于MindSpore的进阶操作，使用MNIST数据集搭建卷积神经网络LeNet5，进行模型训练及评估； | 简单 | Python3.7 MindSpore1.5  Numpy 1.17.5 | PC机 |

请参考官网及《MindSpore环境搭建实验手册》，在PC本地安装MindSpore1.5。

# MNIST手写体识别实验

## 实验介绍

### 关于本实验

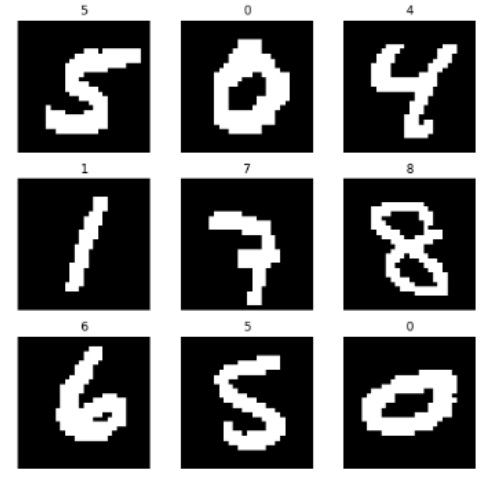
本实验使用MindSpore深度学习框架，进行网络搭建、数据处理、网络训练和测试，完成MNIST手写体识别任务。

数据集介绍：

MNIST数据集来自美国国家标准与技术研究所，National Institute of Standards and Technology(NIST),数据集由来自250个不同人手写的数字构成，其中50%是高中学生，50%来自人口普查局（the Census Bureau）的工作人员。

训练集：60000，测试集：10000

MNIST数据集可在 http://yann.lecun.com/exdb/mnist/ 获取。



数据集展示

### 实验目的

理解MindSpore开发基本流程。

理解MindSpore常用模块的功能。

掌握MindSpore的基础操作。

掌握卷积神经网络的搭建。

### 背景知识

卷积神经网络知识，MindSpore基础知识，MindSpore进阶知识，图像数据预处理，LeNet5卷积神经网络结构。

### 实验设计



实验步骤

## 实验过程

### 环境准备

MindSpore模块主要用于本次实验卷积神经网络的构建，包括很多子模块。

mindspore.dataset：包括MNIST数据集的载入与处理，也可以自定义数据集。

mindspore.common：包中会有诸如type形态转变、权重初始化等的常规工具。

mindspore.nn：主要包括网络可能涉及到的各类网络层，诸如卷积层、池化层、全连接层，也包括损失函数，激活函数等。

Model：承载网络结构，并能够调用优化器、损失函数、评价指标。

代码：

# mindspore.dataset

import mindspore.dataset as ds # 数据集的载入

import mindspore.dataset.transforms.c\_transforms as C # 常用转化算子

import mindspore.dataset.vision.c\_transforms as CV # 图像转化算子

# mindspore.common

from mindspore.common import dtype as mstype # 数据形态转换

from mindspore.common.initializer import Normal # 参数初始化

# mindspore.nn

import mindspore.nn as nn # 各类网络层都在nn里面

from mindspore.nn.metrics import Accuracy # 测试模型用

from mindspore import Model # 承载网络结构

# os模块处理数据路径用

import os

# numpy

import numpy as np

### 数据处理

定义数据预处理函数。

函数功能包括：

1. 加载数据集

2. 打乱数据集

3. 图像特征处理（标准化、通道转换等）

4. 批量输出数据

5. 重复

代码：

def create\_dataset(data\_path, batch\_size=32):

"""

数据预处理与批量输出的函数

Args:

data\_path: 数据路径

batch\_size: 批量大小

"""

# 定义数据集

data = ds.MnistDataset(data\_path)

# 打乱数据集

data = data.shuffle(buffer\_size=10000)

# 数据标准化参数

# MNIST数据集的 mean = 33.3285，std = 78.5655

mean, std = 33.3285, 78.5655

# 定义算子

nml\_op = lambda x : np.float32((x-mean)/std) # 数据标准化，image = (image-mean)/std

hwc2chw\_op = CV.HWC2CHW() # 通道前移（为配适网络，CHW的格式可最佳发挥昇腾芯片算力）

type\_cast\_op = C.TypeCast(mstype.int32) # 原始数据的标签是unint，计算损失需要int

# 算子运算

data = data.map(operations=type\_cast\_op, input\_columns='label')

data = data.map(operations=nml\_op, input\_columns='image')

data = data.map(operations=hwc2chw\_op, input\_columns='image')

# 批处理

data = data.batch(batch\_size)

# 重复

data = data.repeat(1)

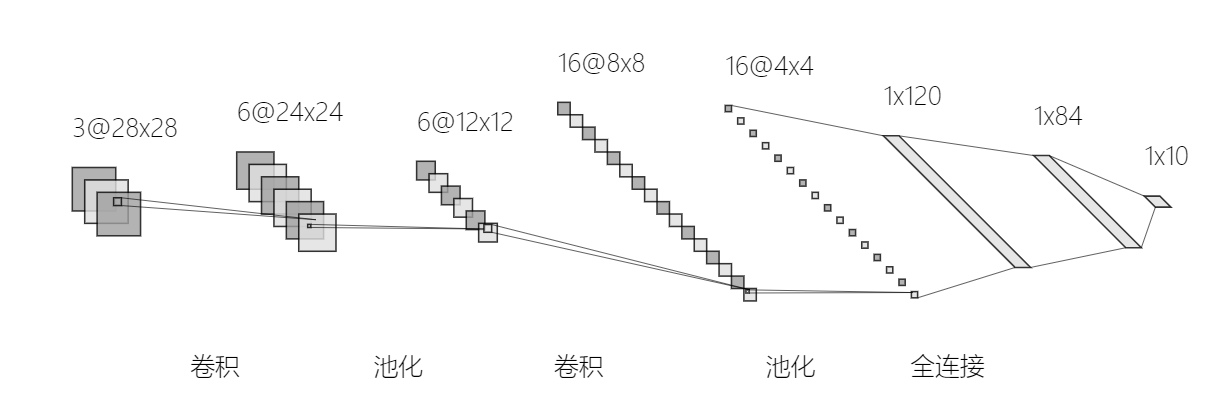
return data

### 网络定义

参考LeNet网络结构，构建网络：

LeNet-5出自论文《Gradient-Based Learning Applied to Document Recognition》，原本是一种用于手写体字符识别的非常高效的卷积神经网络，包含了深度学习的基本模块：卷积层，池化层，全连接层。

本实验将参考LeNet论文，建立以下网络：



Lenet-5网络

INPUT（输入层） ：输入28∗28的图片。

C1（卷积层）：选取6个5∗5卷积核(不包含偏置)，得到6个特征图，每个特征图的一个边为28−5+1=24。

S2（池化层）：池化层是一个下采样层，输出12∗12∗6的特征图。

C3（卷积层）：选取16个大小为5∗5卷积核，得到特征图大小为8∗8∗16。

S4（池化层）：窗口大小为2∗2，输出4∗4∗16的特征图。

F5（全连接层）：120个神经元。

F6（全连接层）：84个神经元。

OUTPUT（输出层）：10个神经元，10分类问题。

代码：

class LeNet5(nn.Cell):

# 定义算子

def \_\_init\_\_(self, num\_class=10, num\_channel=1):

super(LeNet5, self).\_\_init\_\_()

# 卷积层

self.conv1 = nn.Conv2d(num\_channel, 6, 5, pad\_mode='valid')

self.conv2 = nn.Conv2d(6, 16, 5, pad\_mode='valid')

# 全连接层

self.fc1 = nn.Dense(4 \* 4 \* 16, 120, weight\_init=Normal(0.02))

self.fc2 = nn.Dense(120, 84, weight\_init=Normal(0.02))

self.fc3 = nn.Dense(84, num\_class, weight\_init=Normal(0.02))

# 激活函数

self.relu = nn.ReLU()

# 最大池化层

self.max\_pool2d = nn.MaxPool2d(kernel\_size=2, stride=2)

# 网络展开

self.flatten = nn.Flatten()

# 建构网络

def construct(self, x):

x = self.conv1(x)

x = self.relu(x)

x = self.max\_pool2d(x)

x = self.conv2(x)

x = self.relu(x)

x = self.max\_pool2d(x)

x = self.flatten(x)

x = self.fc1(x)

x = self.relu(x)

x = self.fc2(x)

x = self.relu(x)

x = self.fc3(x)

return x

### 模型训练

载入数据集

代码：

train\_path = os.path.join('data','train') # 训练集路径

train\_data = create\_dataset(train\_path) # 定义训练数据集

test\_path = os.path.join('data','test') # 测试集路径

test\_data = create\_dataset(test\_path) # 定义测试数据集

构建网络

构建网络、选择损失函数、优化器、模型。

代码：

# 网络

net = LeNet5()

# 损失函数

net\_loss = nn.SoftmaxCrossEntropyWithLogits(sparse=True, reduction='mean')

# 优化器

lr = 0.01

momentum = 0.9

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

代码：

model.train(3, train\_data) # 训练3个epoch

### 模型评估

查看模型在测试集的准确率。

代码：

model.eval(test\_data) # 测试网络

输出：

{'accuracy': 0.981}

### 效果展示

代码：

data\_path=os.path.join('data', 'test')

ds\_test\_demo = create\_dataset(test\_path, batch\_size=1)

for i, dic in enumerate(ds\_test\_demo.create\_dict\_iterator()):

input\_img = dic['image']

output = model.predict(input\_img)

predict = np.argmax(output.asnumpy(),axis=1)[0]

if i>9:

break

print('True: %s, Predicted: %s'%(dic['label'], predict))

输出：

True: [2], Predicted: 2

True: [0], Predicted: 0

True: [2], Predicted: 2

True: [9], Predicted: 9

True: [7], Predicted: 7

True: [8], Predicted: 8

True: [8], Predicted: 8

True: [2], Predicted: 2

True: [5], Predicted: 5

True: [4], Predicted: 4

### 模型保存

from mindspore import save\_checkpoint,load\_checkpoint, load\_param\_into\_net

save\_checkpoint(net,"lenet5.ckpt")

### Web预测体验

下载Web预测的实验代码：

<https://alexed.obs.cn-north-4.myhuaweicloud.com/MNIST_MINDSPORE_WITHWEB.zip>

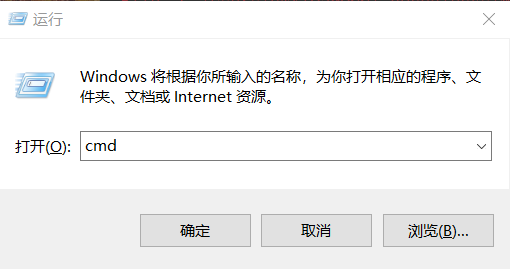
将其解压至纯英文路径下，例如：

C:\Users\\*\*\*\*\*\*\*\Desktop\1.8-9MindSpore\4：MNIST\exp1

将训练好的模型文件lenet5.ckpt文件拷贝/替换到exp1目录内

打开系统终端，以Windows为例：

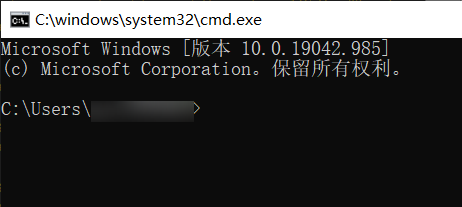
WIN+R键调出如下窗口，



输入cmd命令，点击确定；

cmd

调出Windows终端窗口，如下图，



激活MindSpore的环境：

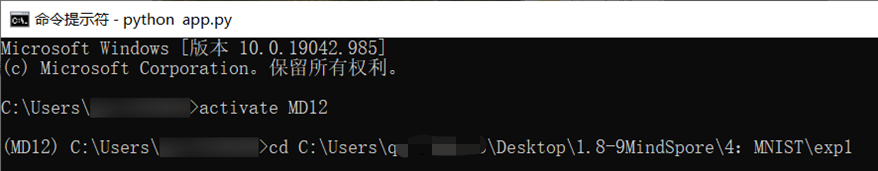
输入“activate + 环境名”，cd进入步骤2中存放exp1实验的路径下：

activate MD12

cd C:\Users\\*\*\*\*\*\*\*\Desktop\1.8-9MindSpore\4：MNIST\exp1

# 如果exp1存放于其他磁盘，需要先输入盘符切换到对应的磁盘，再cd到exp1目录下

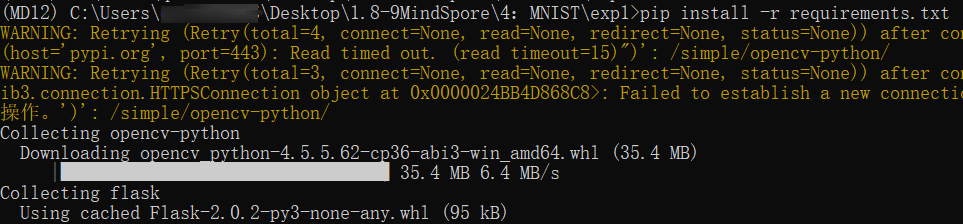
# 例如切换到D盘：d:



之后运行pip 命令：

加载web实验所需的第三方库，opencv-python和flask：

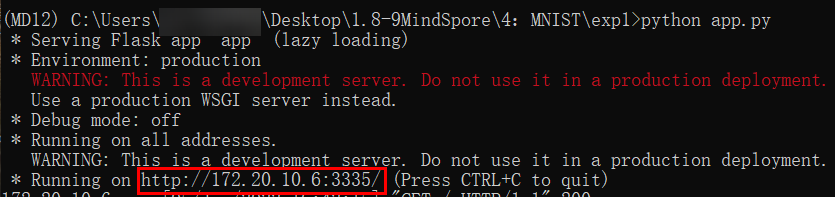
pip install -r requirements.txt



运行app.py

下载完成之后，运行app.py，调用web端。

python app.py



将网址复制到浏览器内打开

之后将出现的http://\*\*\*\*\*\*\*\*\*:3335/ （如上图红框内）的网址复制到浏览器内，打开。

可以在黑框内手写任意0-9的数字，体验Web端手写字预测实验，按“Predict”进行预测，按“Clear”进行清除。



## 实验总结

本实验介绍了MindSpore在图像数据集上的应用，使用MNIST手写体数据集搭建LeNet5卷积神经网络，让学员熟悉MindSpore的进阶用法，掌握MindSpore开发的流程。

## 思考题

1. 请描述MindSpore的基础数据处理流程。

答：数据加载 > shuffle > map > batch > repeat。

2. 定义网络时需要继承哪一个基类？

答：mindspore.nn.Cell。

3. 定义网络时有哪些必须编写哪两个函数？

答：\_\_init\_\_()，construct()。

4. 思考3中提到的两个函数有什么用途？

答：一般会在\_\_init\_\_()中定义算子，然后在construct()中定义网络结构。\_\_init\_\_()中的语句由Python解析执行；construct()中的语句由MindSpore接管，有语法限制。