华为昇腾-昇思MindSpore《机器学习原理》精品课程

机器学习原理

精品课程

实验指导手册

版本:1.0



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

[21基于MindSpore构建MaxPooling层 4](#_Toc130083037)

[21.1实验介绍 4](#_Toc130083038)

[21.1.1数据集的介绍 4](#_Toc130083039)

[21.1.2模型知识点的介绍 4](#_Toc130083040)

[21.3实验环境要求 5](#_Toc130083041)

[21.3实验详细设计与实现 6](#_Toc130083042)

[21.3.1数据准备 6](#_Toc130083043)

[21.3.2实验步骤 6](#_Toc130083044)

[21.4实验总结 7](#_Toc130083045)

[21.5实验任务与参考解答任务 8](#_Toc130083046)

[21.5.1实验任务 8](#_Toc130083047)

[21.5.2参考答案 8](#_Toc130083048)

# 21基于MindSpore构建MaxPooling层

## 21.1实验介绍

本实验主要专注于Max Pooling层的构建原理，利用MindSpore中的nn模块构建Max Pooling层。

### 21.1.1数据集的介绍

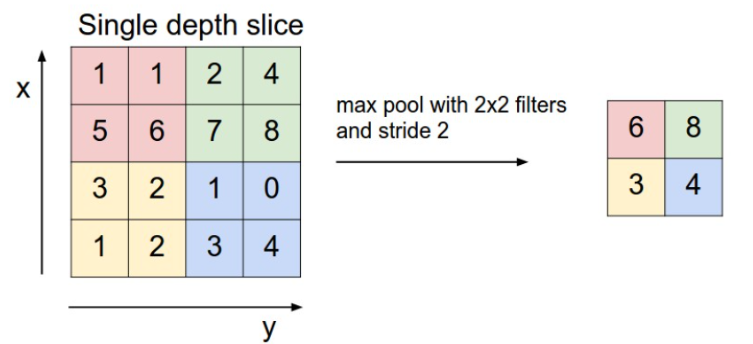
这里使用随机给出的测试数据：

（1）数据维度[1,2,4]，并设置kernel\_size=3, 步长stride=1，其中数值范围在0到10之间，在后续需要使用MindSpore的tensor进行转换，用于构建MaxPool1d层。

（2）数据维度[1, 2, 4, 4]，并设置池化层kernel\_size=3，步长stride=1，其中数值范围在0到10之间，在后续需要使用MindSpore的tensor进行转换，用于构建MaxPool2d层。

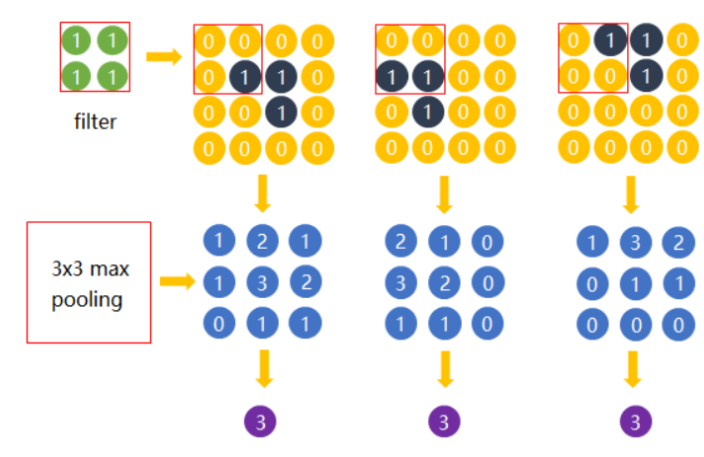
### 21.1.2模型知识点的介绍

进行卷积操作后，需要Max Pooling操作进行最大池化，我们用下图进行演示：将整个图片不重叠的分割成多个大小一致的小块（pooling size）,每个小块内仅取最大数字，舍弃其他区节点，保持原有的平面结构，输出最后结果。



Max Pooling在不同的depth上是分开执行的，且不需要参数控制。Max pooling的主要功能是缩减像素采样，但是Max pooling对识别结果有所损坏， 即卷积后的特征图像中会有冗余信息。我们需要反向思考，这些冗余信息从何而来。

为探测到某个特定形状的存在，用一个filte对整个图片逐步扫描。然而只得到了该特定形状的区域卷积的输出，这个输出结果对我们有用，进而用该fiter卷积其他区域，此时，得出的数值可以判定该形状是否存在，并且准确率很高。比如下图中：



考虑探测“横折”这个形状：进行卷积操作后得到3x3的特征图像，这些图像中真正有用的是数字为“3”的节点，其余数值对于此任务结果没有影响。所以用3x3的Max Pooling后，并没有对“横折”的探测产生影响。如果不使用Max Pooling，而让网络自己去学习，网络也会去学习与Max Pooling近似效果的权重。因为是效果近似，增加了更多的参数，其效果反而可能比Max Pooling差。

## 21.3实验环境要求

在动手进行实践之前，需要注意以下几点：

1. 确保实验环境正确安装，包括安装MindSpore。安装过程：[登录MindSpore官网安装页面](https://www.mindspore.cn/install)，根据安装指南下载安装包及查询相关文档。同时，官网环境安装也可以按下表说明找到对应环境搭建文档链接，根据环境搭建手册配置对应的实验环境。
2. 推荐使用交互式的计算环境Jupyter Notebook，其交互性强，易于可视化，适合频繁修改的数据分析实验环境。
3. 实验也可以在华为云一站式的AI开发平台ModelArts上完成。
4. 推荐实验环境：MindSpore版本=MindSpore 2.4；Python环境=3.11。

|  |  |  |  |
| --- | --- | --- | --- |
| **硬件平台** | **操作系统** | **软件环境** | **环境搭建链接** |
| CPU | Windows-x64 | MindSpore2.4 Python3.11 | [MindSpore环境搭建实验手册第二章2.1节和第三章3.1节](MindSpore环境搭建实验手册.docx) |
| GPU CUDA 10.1 | Linux-x86\_64 | MindSpore2.4 Python3.11 | [MindSpore环境搭建实验手册第二章2.2节和第三章3.1节](MindSpore环境搭建实验手册.docx) |
| Ascend 910 | Linux-x86\_64 | MindSpore2.4 Python3.11 | [MindSpore环境搭建实验手册第四章](MindSpore环境搭建实验手册.docx) |

## 21.3实验详细设计与实现

### 21.3.1数据准备

数据集的获取：

这里使用随机给出的测试数据：

（1）数据维度x=[1,2,4]，并设置kernel\_size=3, 步长stride=1，其中数值范围在0到10之间，在后续需要使用MindSpore的tensor进行转换，用于构建MaxPool1d层。

（2）数据维度y=[1, 2, 4, 4]，并设置池化层kernel\_size=3，步长stride=1，其中数值范围在0到10之间，在后续需要使用MindSpore的tensor进行转换，用于构建MaxPool2d层。

数据加载：

模型构建部分加载数据与模型测试部分加载数据：

模型构建部分加载数据：

#导入mindspore

import mindspore

#导入numpy库

import numpy as np

#导入mindspore中的nn模块

import mindspore.nn as nn

#从MindSpore中导入Tensor库

from mindspore import Tensor

#模型构建部分加载数据

x= Tensor(np.random.randint(0, 10, [1, 2, 4]), mindspore.float32)

#模型测试部分加载数据

y= Tensor(np.random.randint(0, 10, [1, 2, 4, 4]), mindspore.float32)print("x:")

print("x:")

print(x)

print("y:")

print(y)

输出：

x:

[[[7. 0. 8. 7.]

[1. 3. 7. 6.]]]

y:

[[[[8. 1. 9. 4.]

[9. 5. 8. 0.]

[2. 6. 8. 2.]

[5. 2. 3. 3.]]

[[5. 0. 9. 4.]

[6. 2. 2. 3.]

[8. 1. 3. 0.]

[3. 5. 2. 3.]]]]

### 21.3.2实验步骤

导入Python库&模块并配置运行信息

需要使用numpy库以及MindSpore中的nn模块进行最大池化层的构建，并利用MindSpore中的Tensor进行数据转换。导入python库的代码与数据加载部分相同。

定义参数变量

使用nn模块构建MaxPool1d池化层，设置池化层kernel\_size=3, 步长stride=1，根据给定的参数建立MaxPool1d池化层。

#定义池化层大小与步长

max\_pool = nn.MaxPool1d(kernel\_size=3, stride=1)

数据的读取与处理

输入数据维度x=[1, 2, 4]，利用Tensor进行数据转换，其中数值范围在0到10之间，并给出池化后的输出结果。

#输入数据维度，并用Tensor将其转化为32浮点数

x= Tensor(np.random.randint(0, 10, [1, 2, 4]), mindspore.float32)

#经过池化后，输出数据维度

output = max\_pool(x)

result = output.shape

print(result)

输出结果：

(1,2,2)

模型测试

根据步骤1到步骤3，使用nn模块构建MaxPool2d池化层，输入数据维度y=[1, 2, 4, 4]，数据需要利用Tensor进行数据转换，并设置池化层kernel\_size=3, 步长stride=1，其中数值范围在0到10之间, 并给出池化后的输出结果。

提示：

步骤一：设置池化层kernel\_size=3, 步长stride=1；

步骤二：输入数据维度，并用Tensor将其转化为32浮点数；

步骤三：经过池化后，输出数据维度。

请根据提示，补充完整代码：

#依赖包与步骤1相同

#设置池化层kernel\_size=3, 步长stride=1

#输入数据维度，并用Tensor将其转化为32浮点数

#经过池化后，输出数据维度

## 21.4实验总结

本实验介绍了如何基于MindSpore构建Max Pooling层。实验在两个不同的数据维度与参数的基础上，利用MindSpore建立了两个不同的池化层，分别是MaxPool1d层、MaxPool2d层，并对池化后的结果进行输出。

## 21.5实验任务与参考解答任务

### 21.5.1实验任务

根据步骤提示，补充完整代码。

### 21.5.2参考答案

步骤4参考答案：

#参考答案

#导入mindspore

import mindspore

#导入numpy库

import numpy as np

#导入mindspore中的nn模块

import mindspore.nn as nn

#从MindSpore中导入Tensor库

from mindspore import Tensor

#设置池化层kernel\_size=3, 步长stride=1

pool = nn.MaxPool2d(kernel\_size=3, stride=1)

#输入数据维度，并用Tensor将其转化为32浮点数

y = Tensor(np.random.randint(0, 10, [1, 2, 4, 4]), mindspore.float32)

#经过池化后，输出数据维度

output = pool(y)

print(output.shape)

输出：

(1,2,2,2)