

指导书中有关数据集、算法卷、模型卷可以不创建（安排的实验作业用不到），在训练管理中选择开发环境

The screenshot shows the 'Create Development Environment' page. On the left, there's a sidebar with various management options like AI Workflow, Data Annotation, and Model Management. The main area is titled 'Create Development Environment'. It has fields for 'Name' (devenviron-1126-111551), 'Priority' (Normal), 'Other User Access Permissions' (Not Readable), and 'Resource Type' (Public Resource Pool). Under 'Single Node Specification', it lists CPU, MLU, and GPU options, with MLU selected. Below this is a table of resource offerings:

名称	板卡	CPU	内存	根目录存储	价格 (每小时)	RDMA
<input checked="" type="checkbox"/> 1*370-D5.10.26-SHARE	1张MLU370.SHARE	1核 ~ 12 核	2 GiB ~ 64 GiB	15 GiB	0.50	禁用
<input type="checkbox"/> 1*370-D5.10.12-SHARE	1张MLU370.SHARE	1核 ~ 12 核	2 GiB ~ 64 GiB	15 GiB	0.50	禁用

A red box highlights the first row. A note above the table says '两种规格都可以，看哪个有资源就用哪个' (Both specifications can be used, whichever has resources). At the bottom are 'Create', 'Cancel', and '预计需要排队' (Expected to queue) buttons.

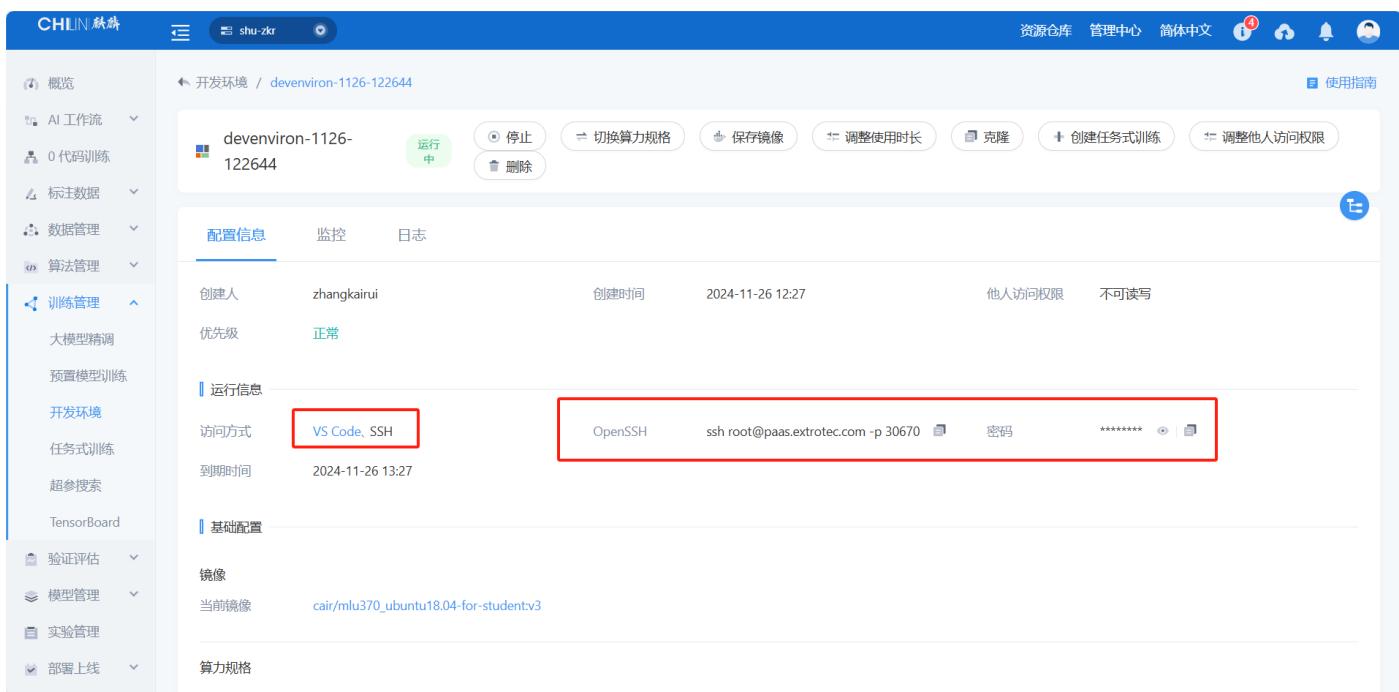
This screenshot shows the same 'Create Development Environment' page with different configuration options. The 'Access Method' section has 'VS Code' selected, with a note: '访问方式可以选VS Code或者SSH' (Access method can be selected as VS Code or SSH). The 'SSH Key' field is empty, with a note: '密钥可以不填' (SSH key can be left empty). The 'Image' section has 'Image Collection' selected, with a note: '选择ubuntu18.04-for-student 做实验x就选vx' (Select ubuntu18.04-for-student for experiments, select vx for others). The 'Node Count' is set to 1. The 'Usage Duration' section has '1 小时' selected, with a note: '选择合适的使用时长' (Select an appropriate usage duration). The 'Data' section includes 'Annotation Dataset' and 'Algorithm' sections. The 'Password' section has a password input field. At the bottom are 'Create', 'Cancel', and '剩下的可以不用管，直接创建' (The rest can be ignored, just create) buttons.

创建好后需要等待一段时间，看到开发容器进入运行中状态即可使用（如果一直是排队中的话可以换另一个规格试试）

The screenshot shows the 'Development Environment List' page. It displays a table of environments, with one row highlighted by a red box:

名称	数据	算法卷	算力规格	节点数	他人访问权限	优先级	创建人	创建时间	到期时间	状态	操作
devenviron-1126-122644			1*370-D5.10.26-SHARE	1	不可读写	正常	zhangkairui	2024-11-26 12:27	2024-11-26 13:27	运行中	停止 ...

点击容器名称查看详情，访问方式里有两种选择：VS Code 和 SSH



CHILIN 极鼎

shu-zkr

资源仓库 管理中心 简体中文

使用指南

概览 AI 工作流 0 代码训练 标注数据 数据管理 算法管理 训练管理 大模型精调 预置模型训练 开发环境 任务式训练 超参搜索 TensorBoard 验证评估 模型管理 实验管理 部署上线

devenviron-1126-122644

运行中 停止 切换算力规格 保存镜像 调整使用时长 克隆 创建任务式训练 调整他人访问权限

配置信息 监控 日志

创建人: zhangkairui | 创建时间: 2024-11-26 12:27 | 他人访问权限: 不可读写

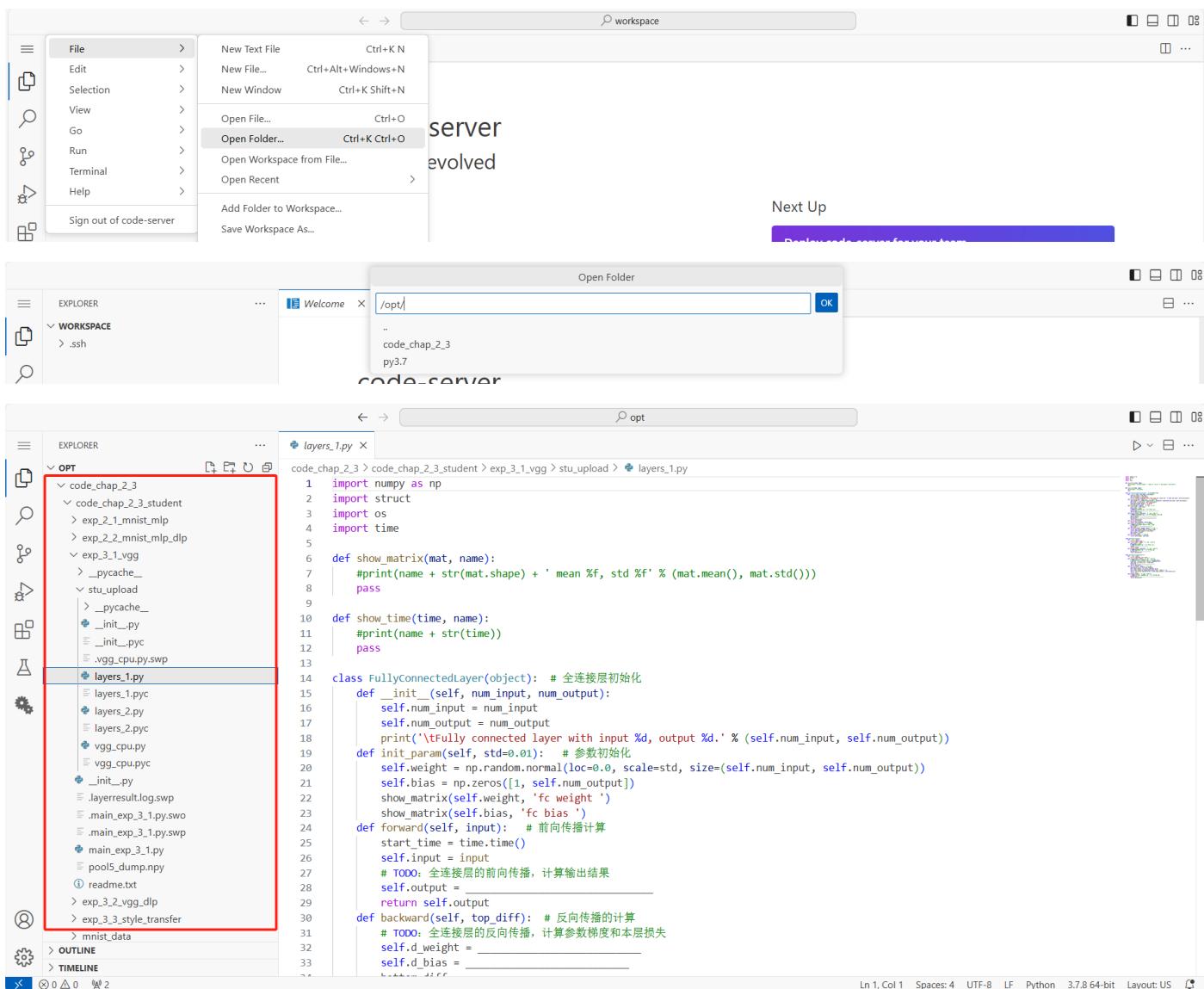
优先级: 正常

运行信息: 访问方式: VS Code, SSH | 到期时间: 2024-11-26 13:27

基础配置: 镜像: 当前镜像 cair/mlu370_ubuntu18.04-for-studentv3

算力规格

可以点击 VS code 进入 code-server， 默认路径是 /workspace，需要切换到 /opt



File >

- New Text File Ctrl+K N
- New File... Ctrl+Alt+Windows+N
- New Window Ctrl+K Shift+N
- Open File... Ctrl+O
- Open Folder... Ctrl+K Ctrl+O
- Open Workspace from File...
- Open Recent
- Add Folder to Workspace...
- Save Workspace As...

server evolved

workspace

EXPLORER WORKSPACE .ssh

WORKSPACE

code_chap_2_3

code_chap_2_3_student

exp_2_1_mnist_mlp

exp_2_2_mnist_mlp_dlp

exp_3_1_vgg

pycache

stu_upload

pycache

init.py

init.pyc

.vgg.cpu.py.swp

layers_1.py

layers_1.pyc

layers_2.py

layers_2.pyc

vgg.cpu.py

vgg.cpu.pyc

init.py

layerresult.log.swp

main_exp_3_1.py.swp

main_exp_3_1.py

pool5_dump.npy

readme.txt

exp_3_2_vgg.dlp

exp_3_3_style_transfer

mnist_data

OUTLINE

TIMELINE

layers_1.py

```
import numpy as np
import struct
import os
import time

def show_matrix(mat, name):
    print(name + str(mat.shape) + ' mean %f, std %f' % (mat.mean(), mat.std()))
    pass

def show_time(time, name):
    print(name + str(time))
    pass

class FullyConnectedLayer(object): # 全连接层初始化
    def __init__(self, num_input, num_output):
        self.num_input = num_input
        self.num_output = num_output
        print('Fully connected layer with input %d, output %d.' % (self.num_input, self.num_output))
    def init_param(self, std=0.01): # 参数初始化
        self.weight = np.random.normal(loc=0.0, scale=std, size=(self.num_input, self.num_output))
        self.bias = np.zeros([1, self.num_output])
        show_matrix(self.weight, 'weight')
        show_matrix(self.bias, 'bias')

    def forward(self, input): # 前向传播计算
        start_time = time.time()
        self.input = input
        # TODO: 全连接层的前向传播, 计算输出结果
        self.output = _____
        return self.output

    def backward(self, top_diff): # 反向传播的计算
        # TODO: 全连接层的反向传播, 计算参数梯度和本层损失
        self.d_weight = _____
        self.d_bias = _____
```

LN 1, Col 1 Spaces: 4 UTF-8 LF Python 3.7.8 64-bit Layout: US