零基础教你

使用 OpenVINO™ 工具套件
部署 YOLOv3 模型

撰写人：王一凡（神州数码 AI 应用工程师、英特尔物联网行业创新大使）

目 录

概 述

目前，人工智能(Artificial Intelligence)已经成为我们生活中触手可及的技术，基于卷积神经网络的深度学习(Deep Learning)，作为实现人工智能的一种重要方法，也得到了蓬勃发展。特别是在计算机视觉领域中，卷积神经网络凭借着其强大的自动提取特征能力和极高的图像分类准确率等特点，深受业界的认可，越来越多的开发者选择卷积神经网络应用在计算机视觉领域。

而目标检测(Object Detection)近年来一直是计算机视觉领域理论和应用的研究热点，目标检测即是在给定的图像中找到期望的物体，同时确定物体的标签和位置。自 2013 年将卷积神经网络引入目标检测算法后，极大提高了目标检测的精度(mAP)，目标检测应用的典型开发流程也精简成收集图片、标注图片、训练模型和部署模型四个步骤。

YOLOv3(You Only Look Once,Version 3)是一种实时目标检测算法，本文在 YOLOv3 算法的基础上，详细的介绍了如何搭建深度学习工具所需要的运行环境，训练目标检测模型等一系列步骤。由于训练目标检测模型需要极大的算力，本文的硬件选择为英特尔®NUC(Next Unit of Computer)产品线系列的幻影峡谷，英特尔®NUC 是英特尔®公司设计制造的功能强大的迷你计算机(Mini PC)，而该系列中的幻影峡谷是一款具备 AI 训练能力的迷你计算机。如图 1-1 所示。

图 1-1 幻影峡谷实物图

本文训练深度学习模型使用的幻影峡谷，其中央处理器(CPU)为第十一代英特尔®酷睿™i7，搭载的是 RTX2060 独立显卡(6GB GDDR6)。在深度学习开发环境所需的软件上，本文使用 PaddleX 作为目标检测模型训练的框架，OpenVINO™工具套件为目标检测模型优化部署的框架。PaddleX 是基于百度飞桨(PaddlePaddle)核心框架、开发套件和工具组件的深度学习全流程开发工具。具备全流程打通、融合产业实践、易用易集成三大特点；OpenVINO™工具套件是英特尔®发布专注于优化神经网络推理的开源工具包，正文将详细介绍 PaddleX 框架和 OpenVINO™工具套件的安装和使用，以及目标检测模型训练的全部流程。

1. 安装 Python 和 Anaconda

1.1 Python 和 Anaconda 简介

Python 是一种解释型高级通用编程语言，其在人工智能编码语言中发挥着至关重要的作用，人工智能领域的相关库或框架(如 scikit-learn、Tensorflow、Caffe 以及 PaddlePaddle 等)都是基于 Python 编程语言开发的。Python 虽然强大好用，但管理其数量庞大的第三方库，并解决其依赖关系是非常复杂的事情。

Anaconda 作为虚拟环境和 Python 库的管理工具，极大的方便了 Python 开发者管理 Python 所需要的虚拟环境和第三方库，而且解决了各种库之间的依赖关系。

1.2 下载并安装 Anaconda

首先下载并安装 Anaconda，具体步骤如下。

第一步，通过网址 https://www.anaconda.com/products/individual 进入 Anaconda 官网，点击 Download 进入下载界面，根据需求选择合适的下载文件，如图 1-2 所示。

图 1-2 下载 Anaconda

第二步，找到 Anaconda 下载文件 Anaconda3-2020.11-Windowns-x86_64.exe 并双击安装，进入用户选项界面默认选择 Just Me，再点击 Next> 按钮，如图 1-3 所示。

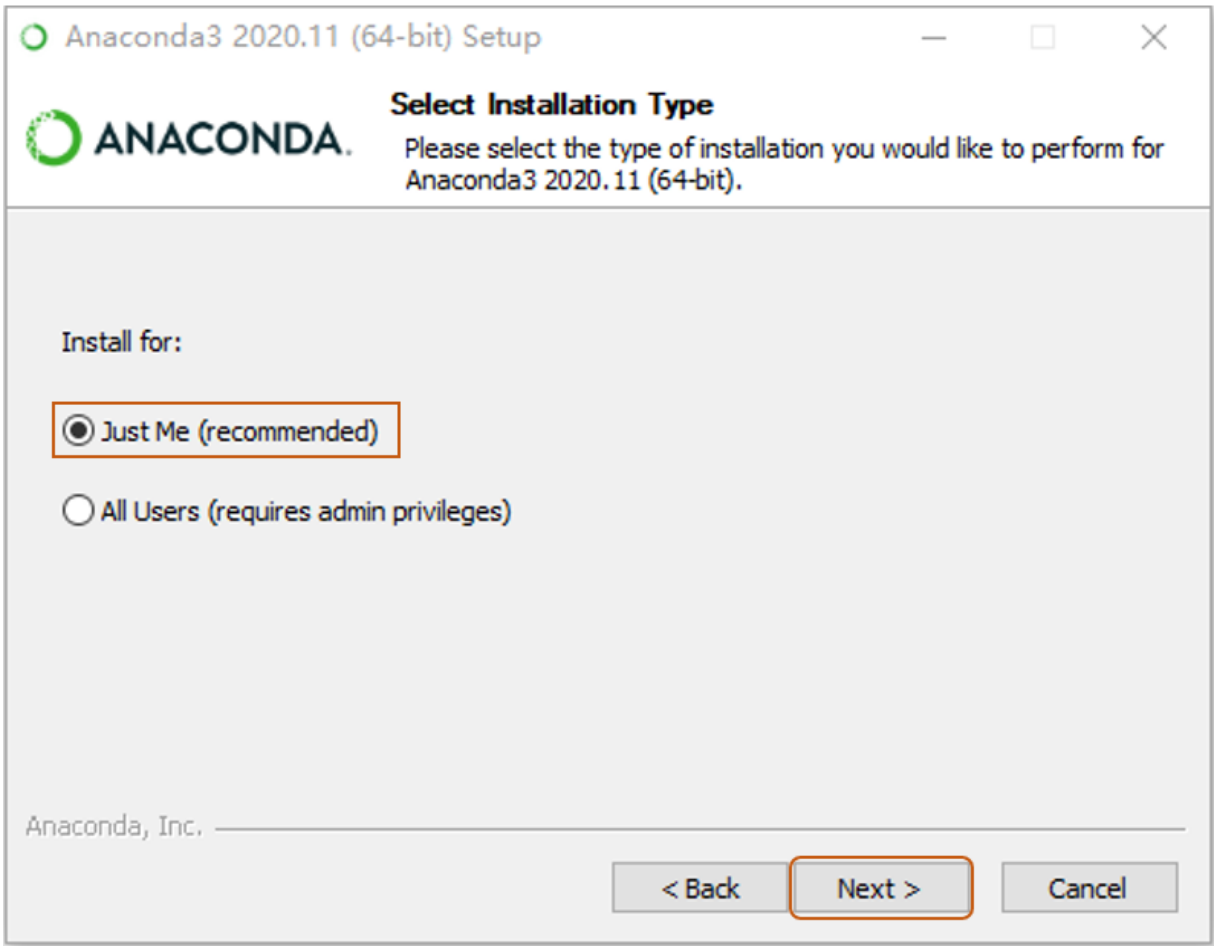


图 1-3 选择 Just Me

第三步，设置安装路径，尽量保持默认路径，然后点击 Next>按钮安装，如图 1-4 所示。

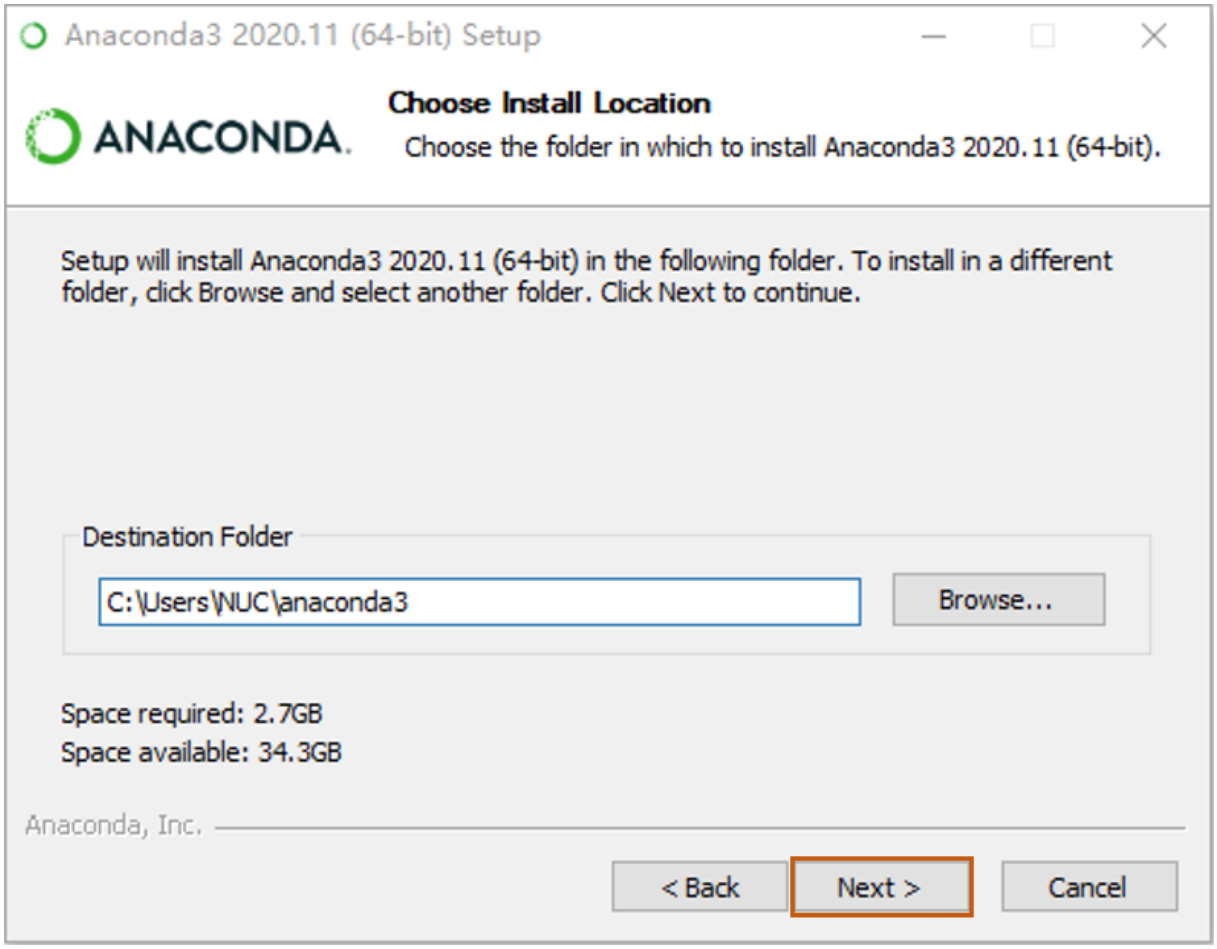


图 1-4 保持默认路径

第四步，进入高级安装选项设置，一定要勾选 Add Anaconda3 to my PATH environment variable，将 Anaconda3 的路径添加到环境变量中，然后点击 Install 按钮，Anaconda 安装完成，如图 1-5 所示。

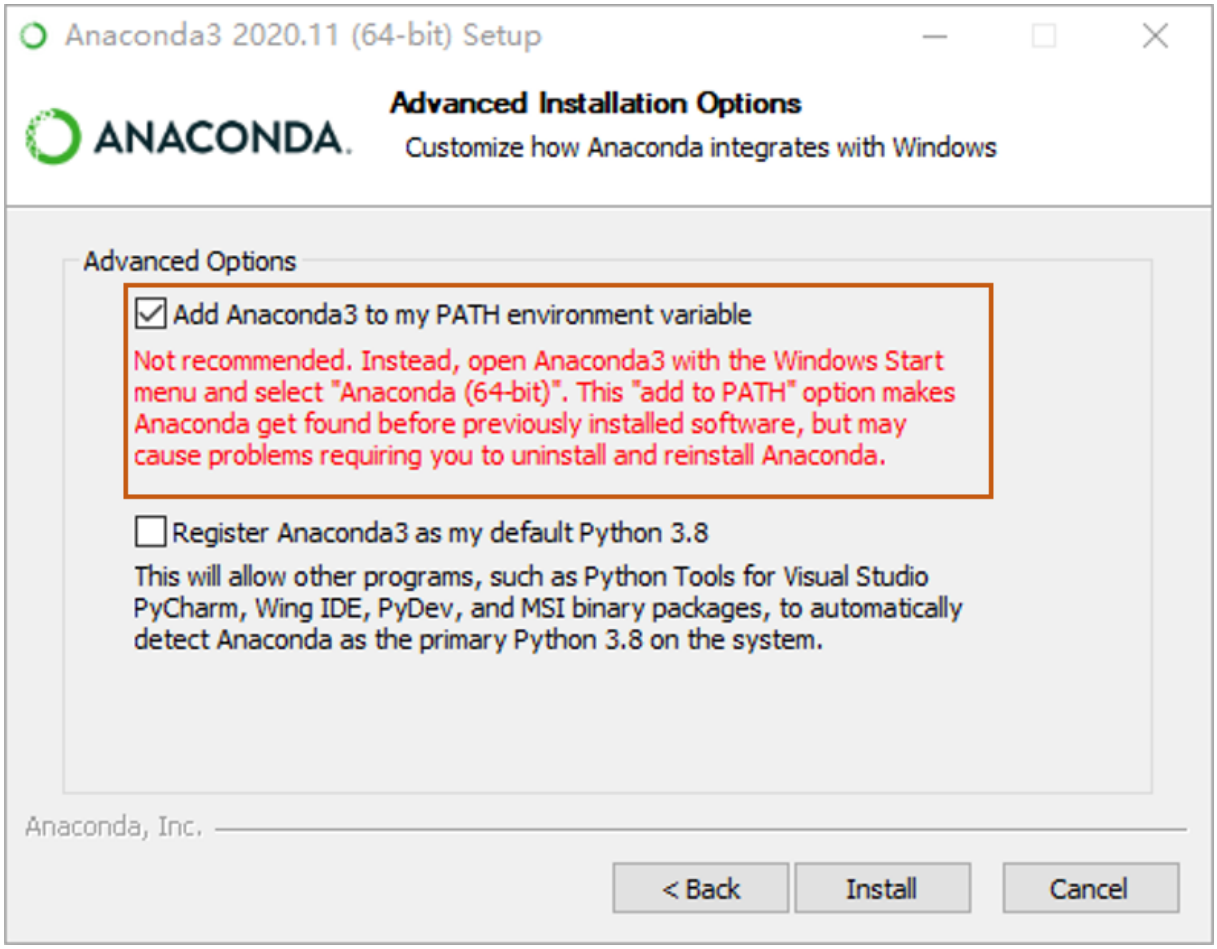


图 1-5 添加 Anaconda 路径到 PATH 环境变量

1.3 测试 Anaconda 安装

全部安装完毕后，在 Windows“开始”菜单中选择 Anaconda Navigator ，进入主界后点击 Environments 选项卡，如下图 1-6 所示可以看到当前的 Anaconda 默认虚拟环境是 base(root)，单击 base(root)右侧的绿色箭头，在弹出的菜单中选择 Open with Python。

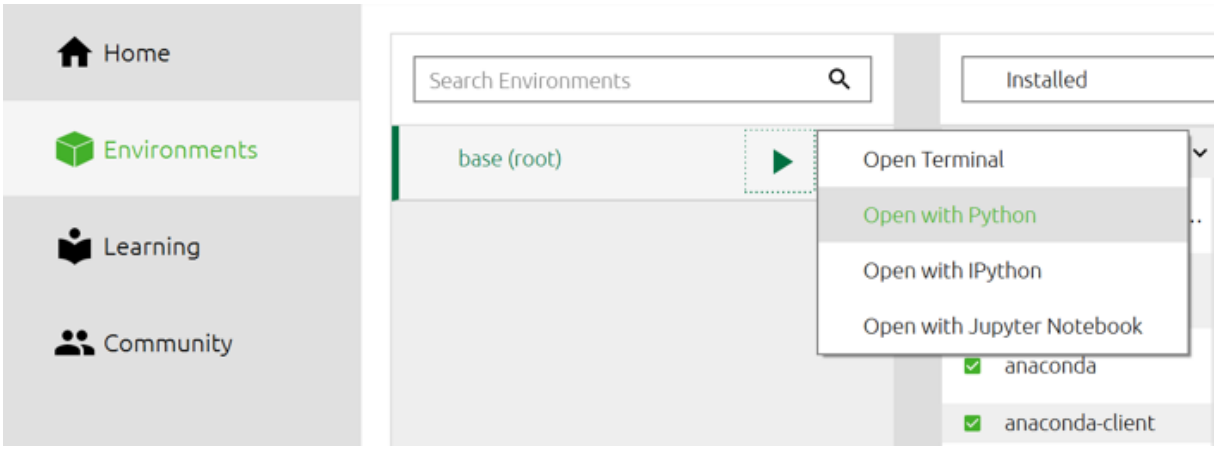


图 1-6 Open with Pythpn

在弹出 Windows 命令行窗口中，输入代码<print(“hello world!”)>，得到如下图 1-7 的结果证明 Anaconda 和 Python 全部安装成功。

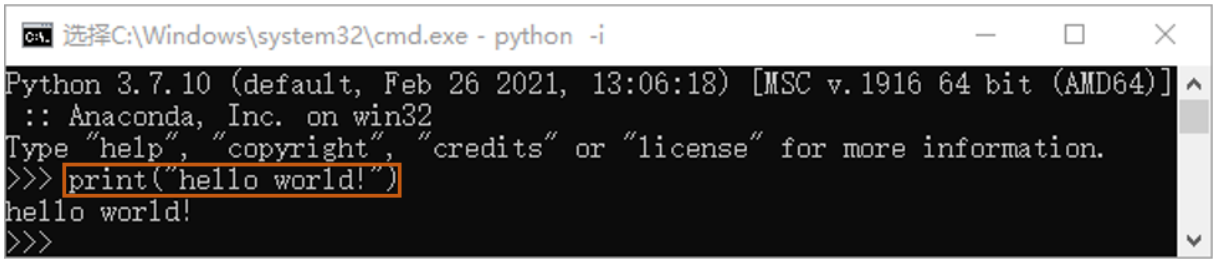


图 1-7 hello world!

2. 安装 Visual Studio Code

2.1 Visual Studio Code 简介

Visual Studio Code 是微软公司的一款开源免费跨平台代码编辑器，其具有丰富的其他语言(例如 C++，C#，Java，Python，PHP，Go)和运行时(例如.NET 和 Unity)扩展的生态系统。Visual Studio Code 具有语法高亮、代码补全以及多插件支持等功能，本文将以 Visual Studio Code 作为 Python 代码的集成开发环境。

2.2 Visual Studio Code 安装

下载并安装 Visual Studio Code 的具体步骤如下。

第一步，通过网址 https://code.visualstudio.com 进入 Visual Studio Code 官网，点击 Download for Windows 进入下载界面，根据需求选择合适的下载文件，本文章使用 VSCodeUserSetup-x64-1.55.2 版本,如图 2-1 所示。

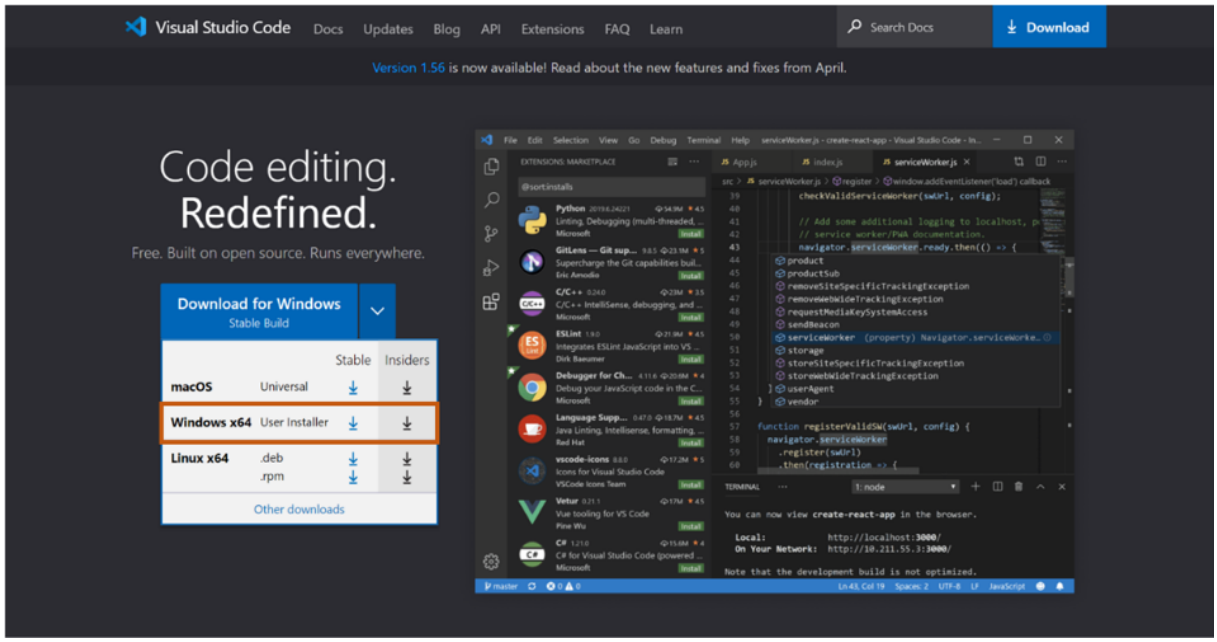


图 2-1 Visual Studio Code 下载界面

第二步，找到 VSCodeUserSetup-x64-1.55.2 下载文件双击安装，在许可协议界面勾选我同意此协议(A)，点击“下一步(N)>”按钮到安装路径界面，在安装路径设置界面，尽量保持默认设置，然后再点击“下一步(N)>”按钮，如图 2-2 所示。

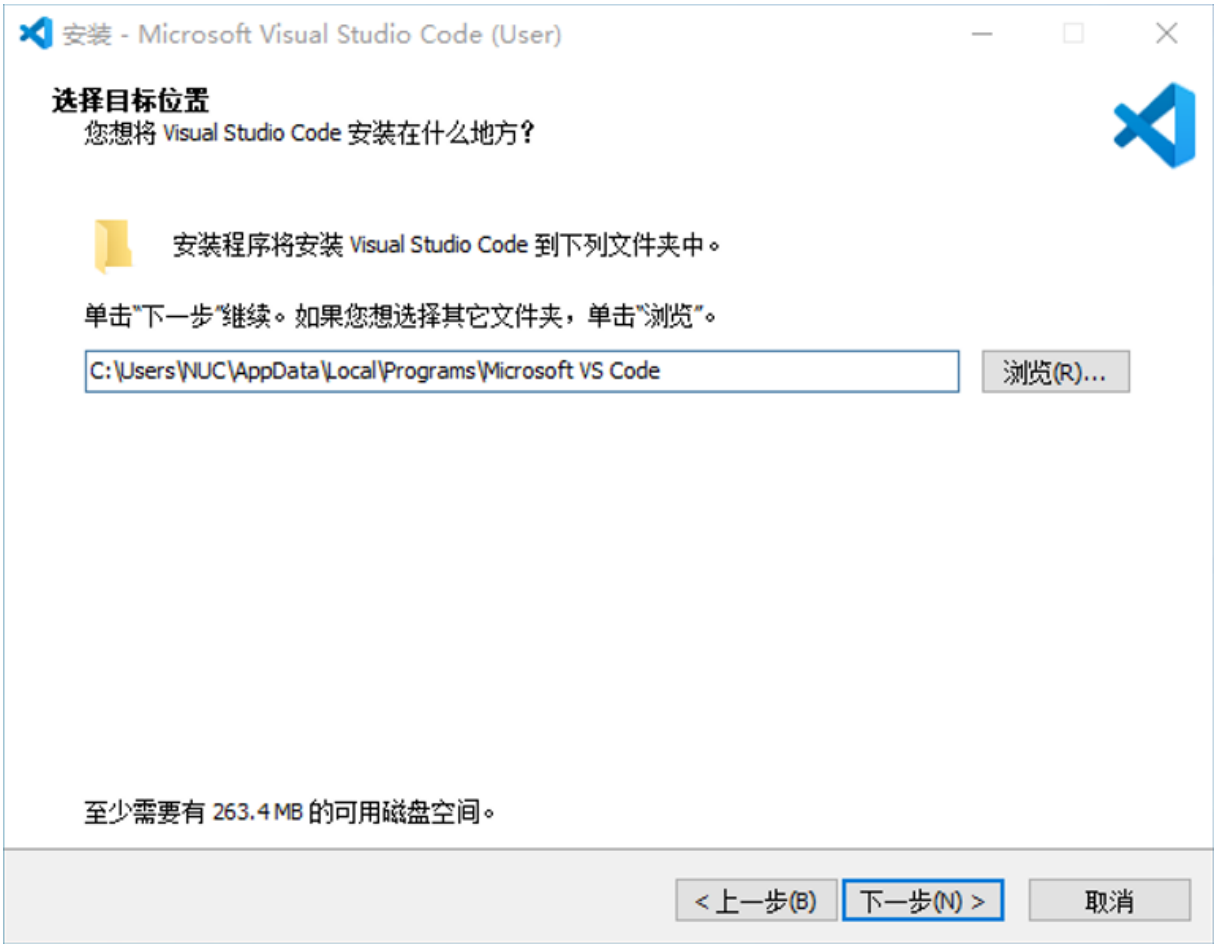


图 2-2 保持默认路径

第三步，默认选择开始文件夹菜单后，点击“下一步(N)>”按钮进入安装高级选项界面，勾选“添加到 PATH(重启后生效)”复选框，添加 Visual Studio Code 路径到 Windows PATH 环境变量，之后点击“下一步(N)>”按钮，再点击“安装”按钮完成 Visual Studio Code 安装，如图 2-3 所示。

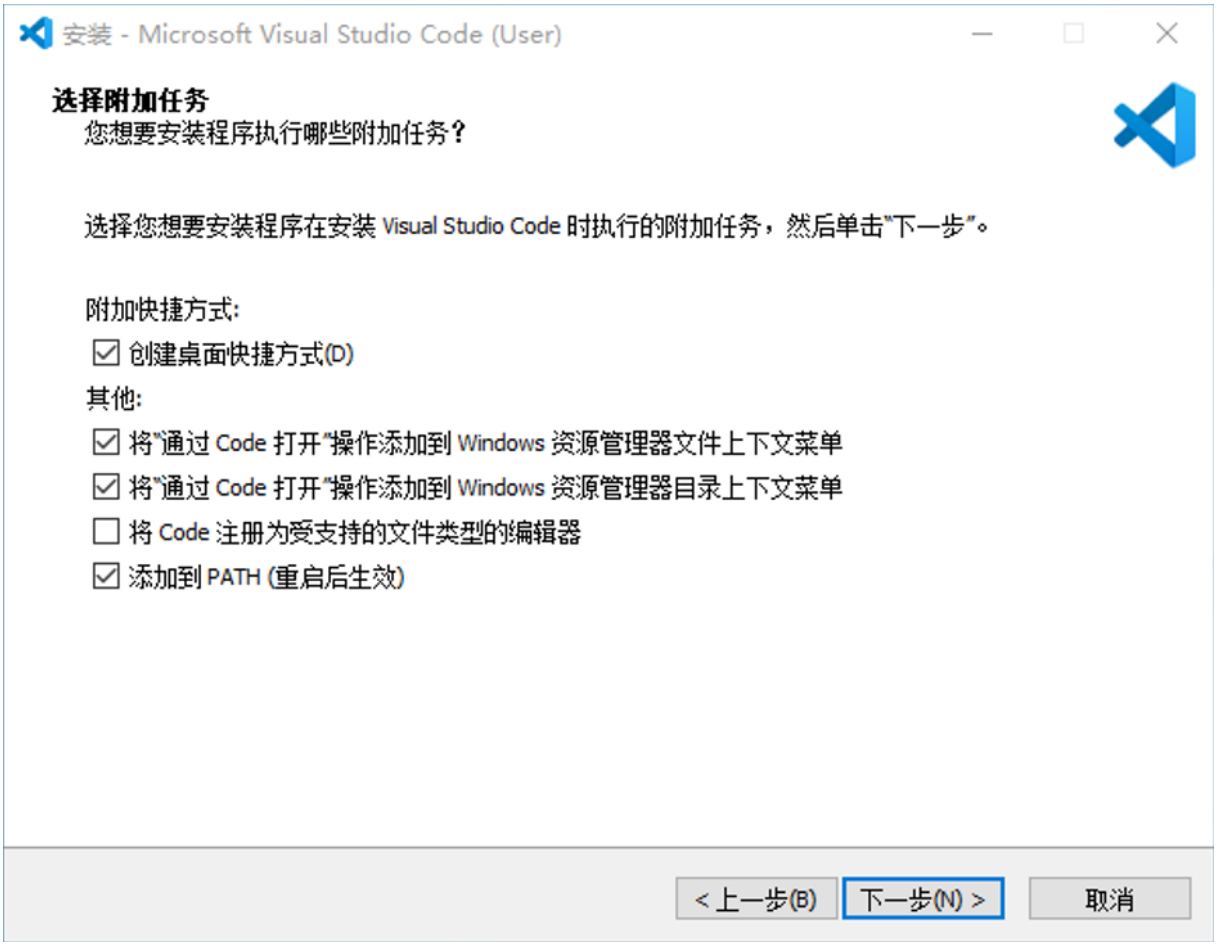


图 2-3 添加到 PATH(重启后生效)

2.3 Visual Studio Code 配置

2.3.1 安装插件

启动 Visual Studio Code，点击左侧网格图标，在输入栏中输入 Python，在弹出的菜单中选择 Python，再选择 Install 按钮，如图 2-4 安装 Python 插件所示，完成 Python 插件的安装，同样步骤可完成 Python Path 插件、Chinese 中文插件的安装。

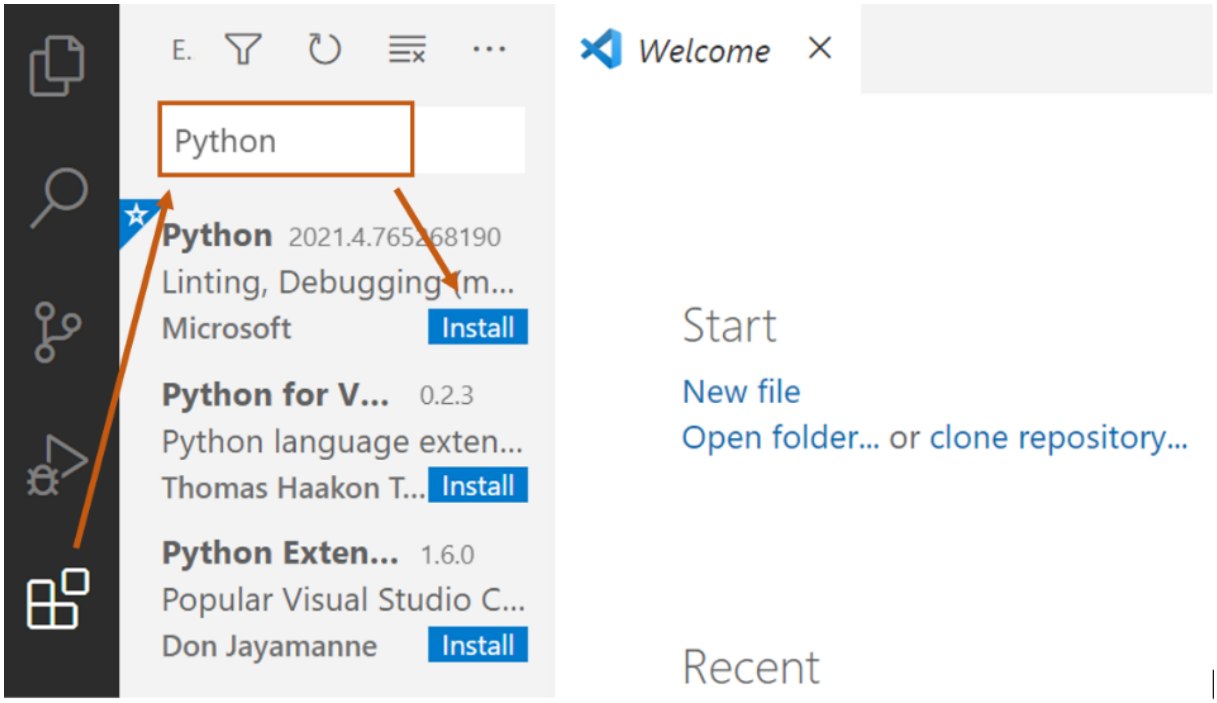


图 2-4 安装 Python 插件

2.3.2 关联 Anaconda

启动 Visual Studio Code，在 File 菜单中选择 Preferences，在弹出的菜单中选择 Settings，在搜索栏 search settings 中输入 Python.PythonPath,在下方的方框中输入安装的 Anaconda 路径，如图 2-5 所示。

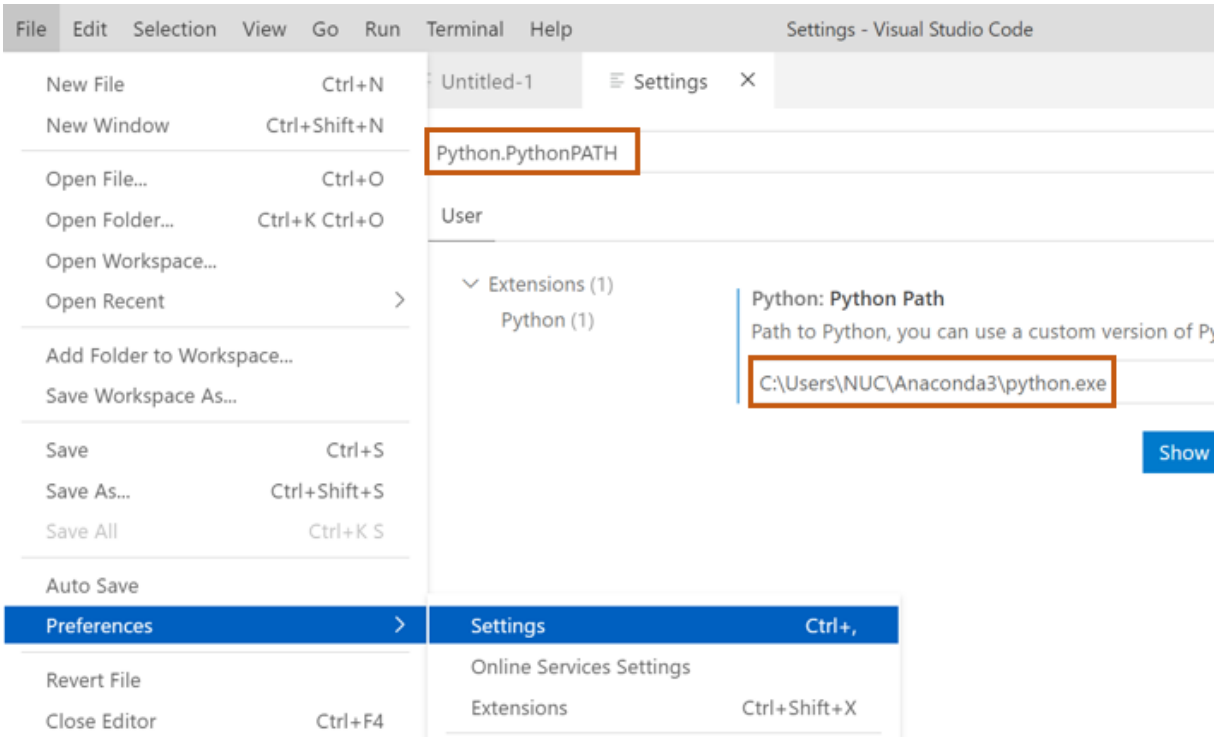


图 2-5 关联 Anaconda 路径

2.3.3 在 Visual Studio Code 中运行 Python 代码

启动 Visual Studio Code，在 File 菜单中选择 New File，新建代码文件。由于 Visual Studio Code 当前不清楚新建文件的类别，所以此时文件名默认为 Unititled-1。输入代码<print(“hello world!”)>，在 File 菜单中选择 Save 选项，在弹出的弹框中修改文件名为 test 后，在“保存类型（T）”中选择 Python，本文中的文件既保存成文件名为 test.py 的 Python 源代码文件，点击主界面右上角的绿色箭头，或者按下快捷键<Ctro+F5>即可运行 Python 代码。

3. 安装 PaddleX

3.1 PaddleX 简介

PaddleX 是百度飞桨全流程开发工具，集飞桨核心框架、模型库、工具及组件等深度学习开发所需全部能力于一身，打通深度学习开发全流程。PaddleX 提供两种使用模式，一种是简明易懂的 Python API，另一种是一键下载安装的图形化开发客户端。用户可根据实际生产需求选择相应的开发方式，获得飞桨全流程开发的最佳体验。

3.2 PaddleX Python API 安装

3.2.1 在 Anaconda 创建 PaddleX 虚拟环境

为了方便 PaddleX 对深度学习模型的训练和管理，在 Anaconda 默认虚拟环境 base(root)的基础上，创建一个新的虚拟环境 PaddleX。创建并配置 Anaconda 虚拟环境具体的步骤如下，在 Windows“开始”菜单中选择 Anaconda Navigator ，进入主界面后点击 Environments 选项卡，单击下方 Create 按钮，在弹框中将新的虚拟环境命名为 PaddleX，Python 版本选择为 3.8，因为 PaddleX 对 Python 版本的要求是 3.5~3.8，可以根据需要自行选择，本文选择 Python3.8 版本。选择好之后，单击 Create 按钮，完成 PaddleX 虚拟环境的创建和配置工作，如图 3-1 所示。

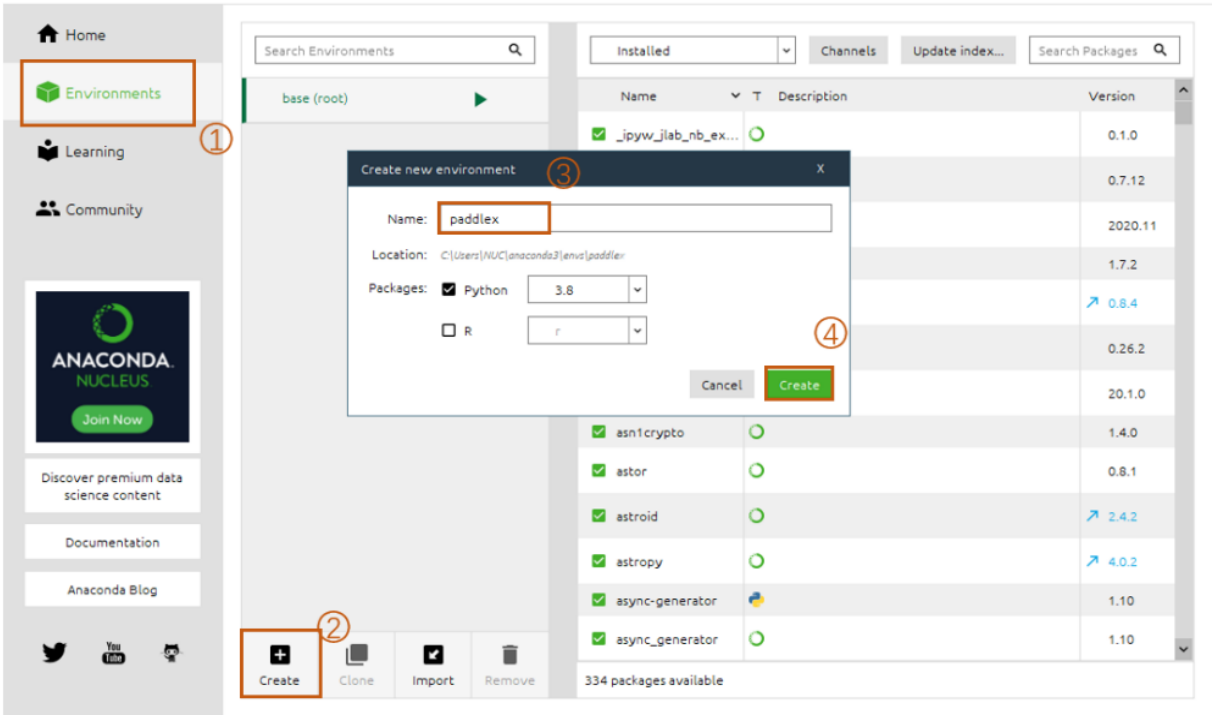


图 3-1 创建 PaddleX 虚拟环境

3.2.2 安装 PaddlePaddle

在 PaddleX 安装之前，需要安装 PaddlePaddle-GPU 或者 PaddlePaddle(版本大于或等于 1.8.1)，安装 PaddlePaddle 的具体步骤如下。

第一步，本文使用 PaddlePaddle-GPU 10.2 版本，或者点击飞桨官网 https://www.paddlepaddle.org.cn 选择合适的版本，左键单击虚拟环境 PaddleX 右侧绿色的箭头，点击 Open Terminal，在弹出的 Windows 命令行窗口中，输入命令<python -m pip install paddlepaddle-gpu -i https://mirror.baidu.com/pypi/simple>，使用 pip install 进行快速安装，如图 3-2 所示。

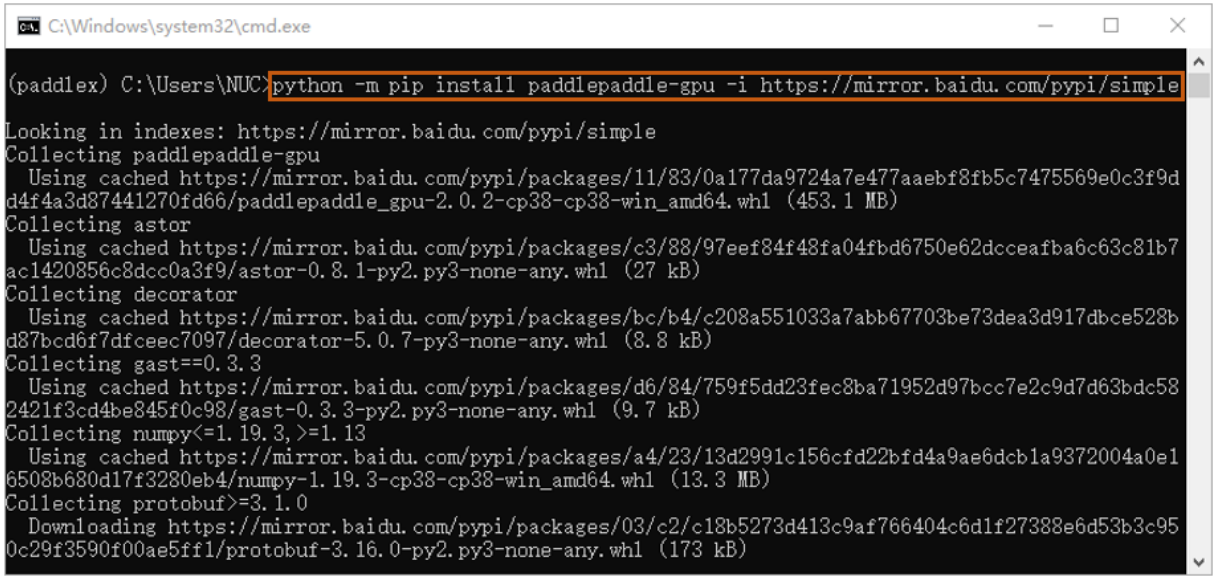


图 3-2 安装 PaddlePaddle

第二步，安装 GPU 版的 PaddlePaddle 还需要安装相应版本的 CUDA 和 cuDNN，10.2 版本的 PaddlePaddle-GPU 对应 CUDA=10.2，cuDNN=7.6.5，在打开的 Open Terminal 输入命令<conda install cudatoolkit=10.2.89 cudnn=7.6.5>，用 conda install 安装 CUDA 和 cuDNN，运行结果如图 3-3 所示。

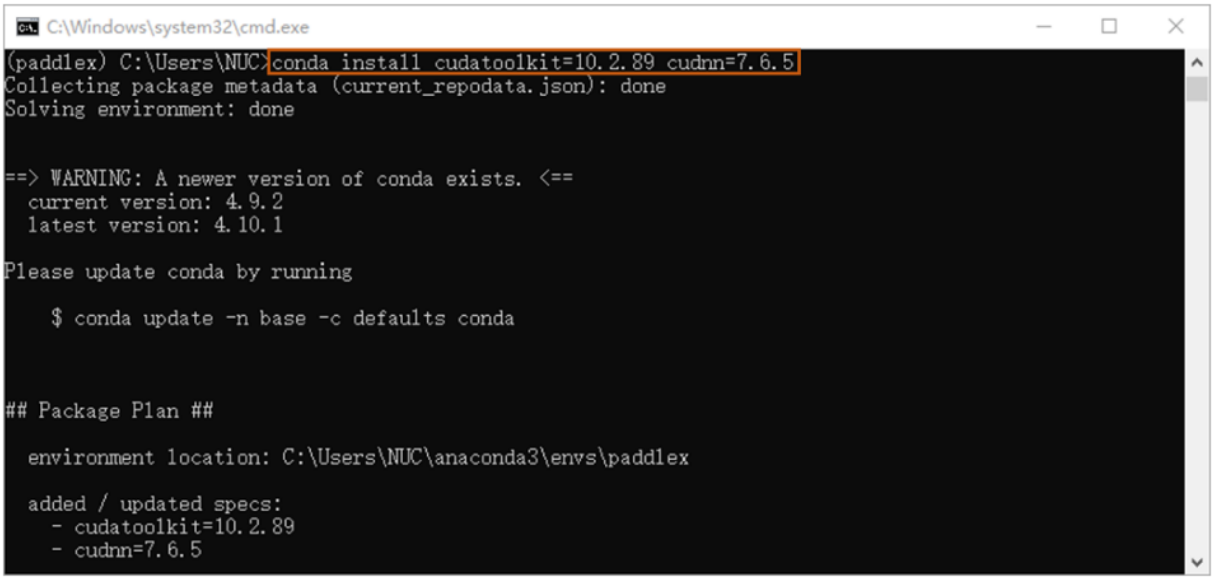


图 3-3 安装对应版本的 CUDA 和 cuDNN

第三步，验证 PaddleePaddle 安装，输入命令行<python -c “import paddle;paddle.utils.run_check()”>，如果出现 PaddlePaddle is installed successfully！则证明安装成功，如图 3-4 所示。

图 3-4 验证 PadddlePaddle 安装

3.2.3 安装 PaddleX Python API

安装 PaddleX Python API 的具体步骤如下。

第一步，在 Anaconda 的虚拟环境 PaddleX 中，打开 Open Terminal 进入 Windows 命令行窗口，输入命令< pip install PaddleX -i https://mirror.baidu.com/pypi/simple >，使用 pip install 快速安装 PaddleX，运行结果如图 3-5 所示。

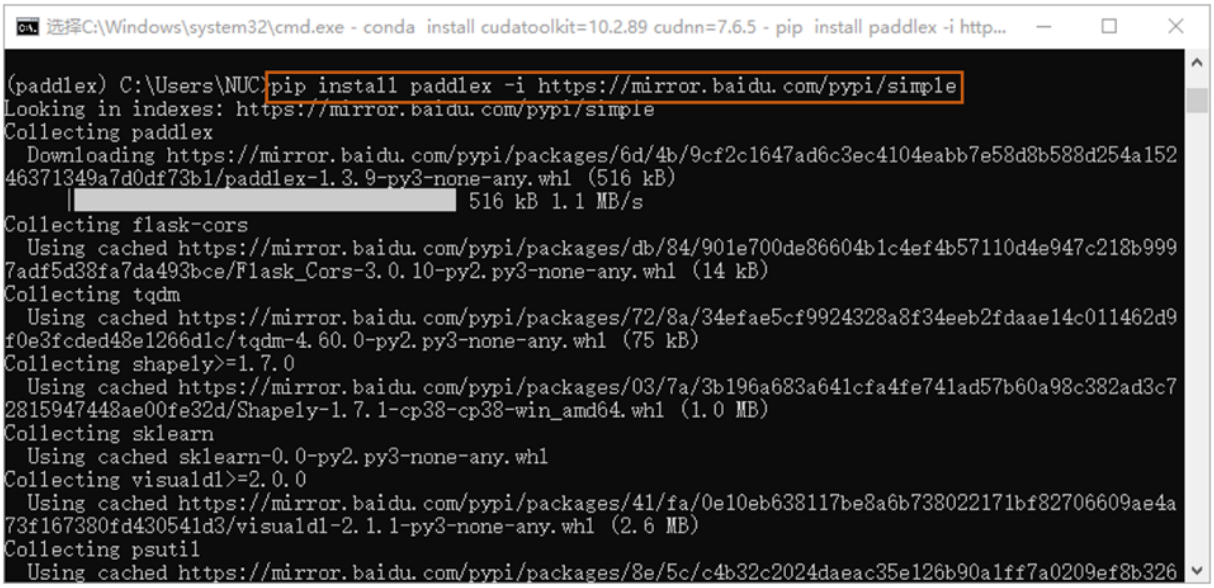


图 3-5 pip 安装 PaddleX

第二步，安装 PaddleX 依赖的 pycocotools 包，注意 pycocotools 在 Windows 安装较为特殊，可用< pip install pycocotools -i https://mirror.baidu.com/pypi/simple >命令，使用 pip install 进行快速安装，运行结果如图 3-6 所示。

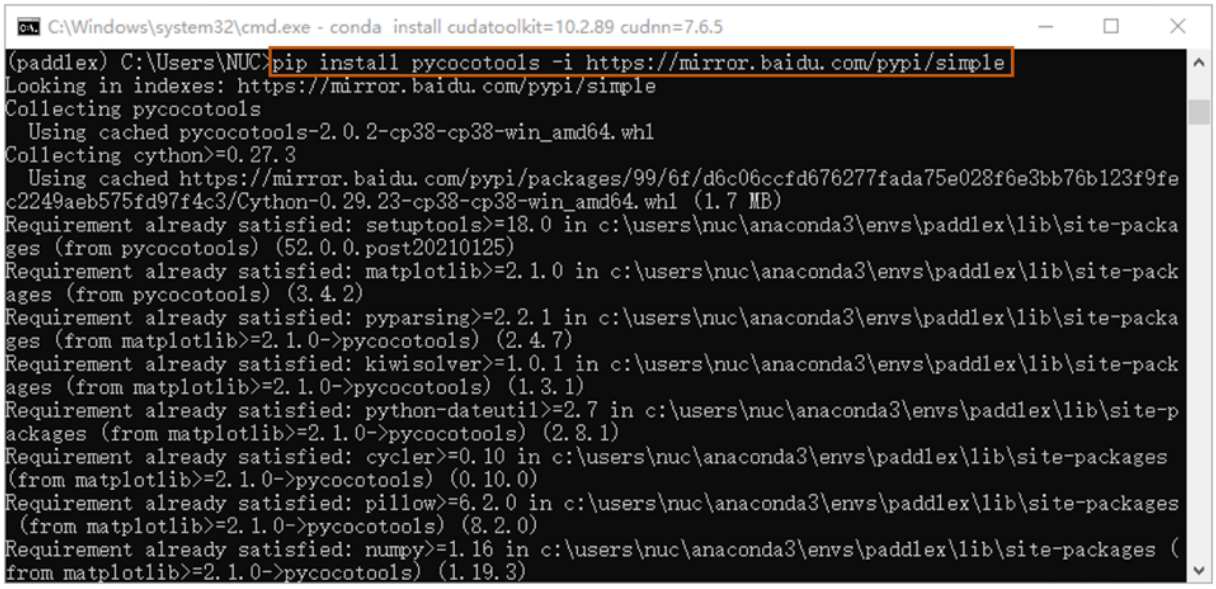


图 3-6 安装 pycocotools 包

第三步，验证 PaddleX 安装，输入命令<python -c "import PaddleX as pdx;print(pdx.__version__)">，结果如下图 3-7 所示，即为安装成功。

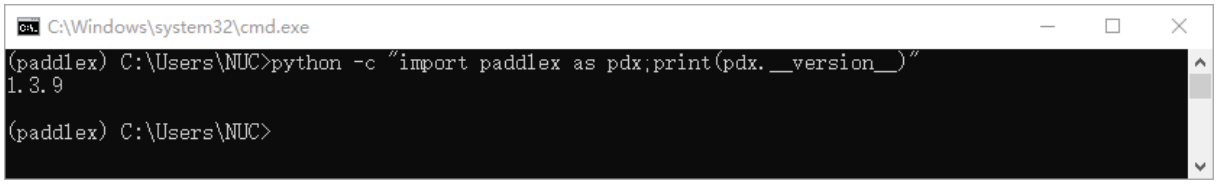


图 3-7 验证 PaddleX 安装

3.3 PaddleX 可视化客户端安装

3.3.1 PaddleX 可视化客户端简介

PaddleX 可视化客户端(GUI)基于 PaddlePaddle 开发的深度学习模型训练套件，目前支持训练计算机视觉领域的图像分类、目标检测、实例分割和语义分割四大任务，同时支持模型裁剪、模型量化两种方式压缩模型。开发者以点选、键入的方式无需代码快速体验深度学习模型开发的全流程。

3.3.2 PaddleX 可视化客户端安装

PaddleX 可视化客户端是 PaddleX API 的衍生品，它在集成 API 的基础上，额外提供了可视化分析、评估等附加功能，PaddleX 打通了深度学习模型开发必须的数据处理、超参配置、模型训练及优化以及模型发布的全部流程，无需开发一行代码，即可得到高性能深度学习推理模型。具有数据集智能分析、自动超参推荐、可视化模型评估、模型剪裁及量化、预训练模型管理、可视化模型测试和模型多端部署等多种独特的功能。

具体安装 PaddleX GUI 步骤如下。

第一步，通过网址 https://www.paddlepaddle.org.cn/paddle/paddleX 进入飞桨官网，点击“下载客户端按钮”后进入下载界面，注册完 PaddleX 后，在下载界面根据需求选择合适的版本下载文件，本文章使用 Windows 版本，如图 3-8 所示。

图 3-8 PaddleX 图形化开发界面下载

第二步，找到 PaddleX 下载文件“paddlex_gui_win10_v1.1.7”并双击安装，进入安装和路径选择界面，可以默认路径也可以自定义目标文件夹，本文使用默认路径，再点击“安装”按钮开始安装如图 3-9 所示。

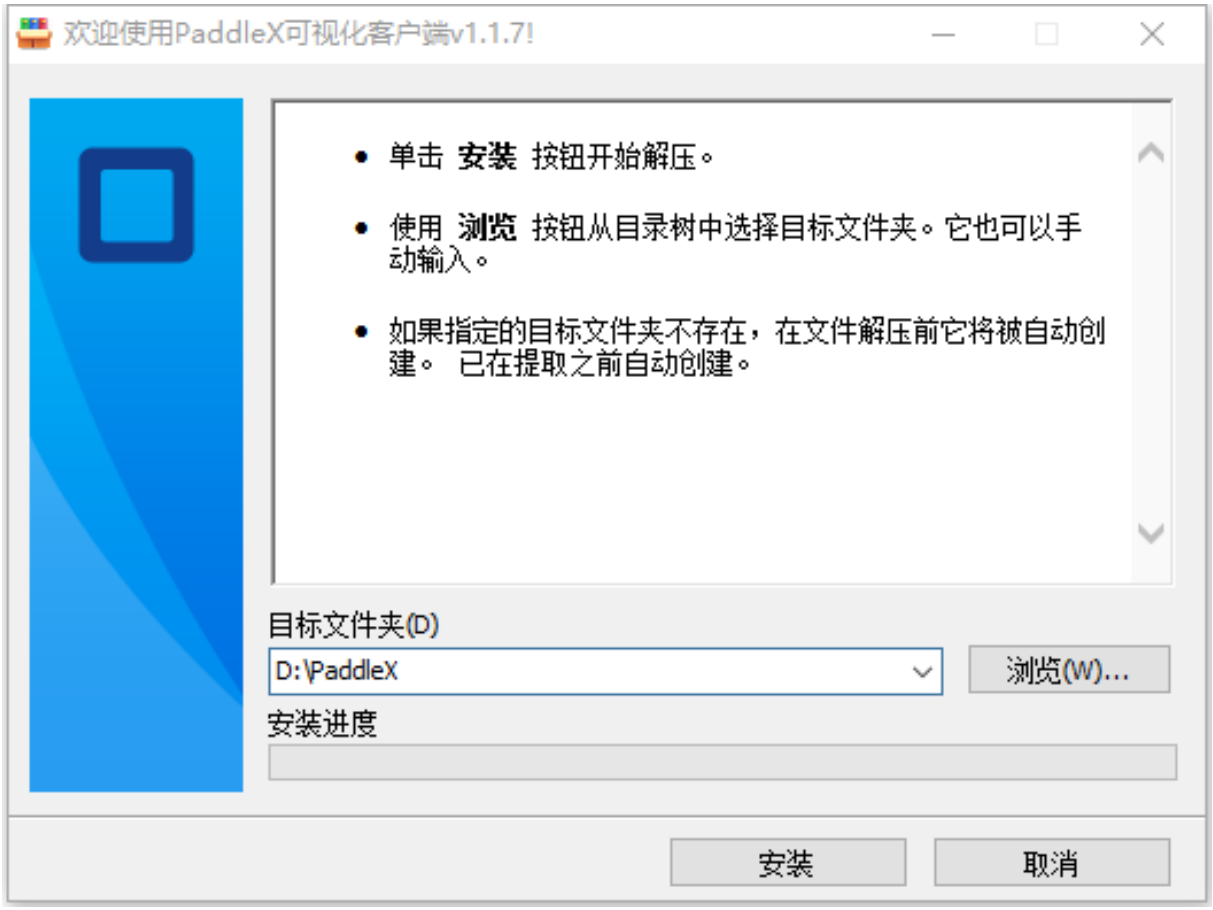


图 3-9 安装 PaddleX 可视化界面

第三步，进入初始化界面，初始化工作空间，用于存储项目所用数据集以及模型相关数据，注意，工作空间路径不能包含中文或空格字符，如图 3-10 所示，最后点击“确定”按钮。

图 3-10 进入初始化界面

第四步，下载案例工程，可以根据实际需要选择下载示例项目的类别点击“确定”按钮下载，如图 3-11 所示。也可选择“跳过”，之后在主界面点击设置下载案例工程。PaddleX 可视化客户端下载完毕，进入主界面。

图 3-11 下载案例工程

4. 准备猫狗数据集

4.1 Kaggle 猫狗数据集下载

本文从 Kaggle 直接下载猫狗数据集，下载步骤如下。

第一步，首先点击地址：

https://www.kaggle.com/account/login?phase=startSignInTab&returnUrl=%2F。

进入 Kaggle 账户注册页，根据提示注册账户。完成注册后，直接进入地址：https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data。如图 4-1 所示进入猫狗数据集下载界面，点击 Download All 下载全部数据集。

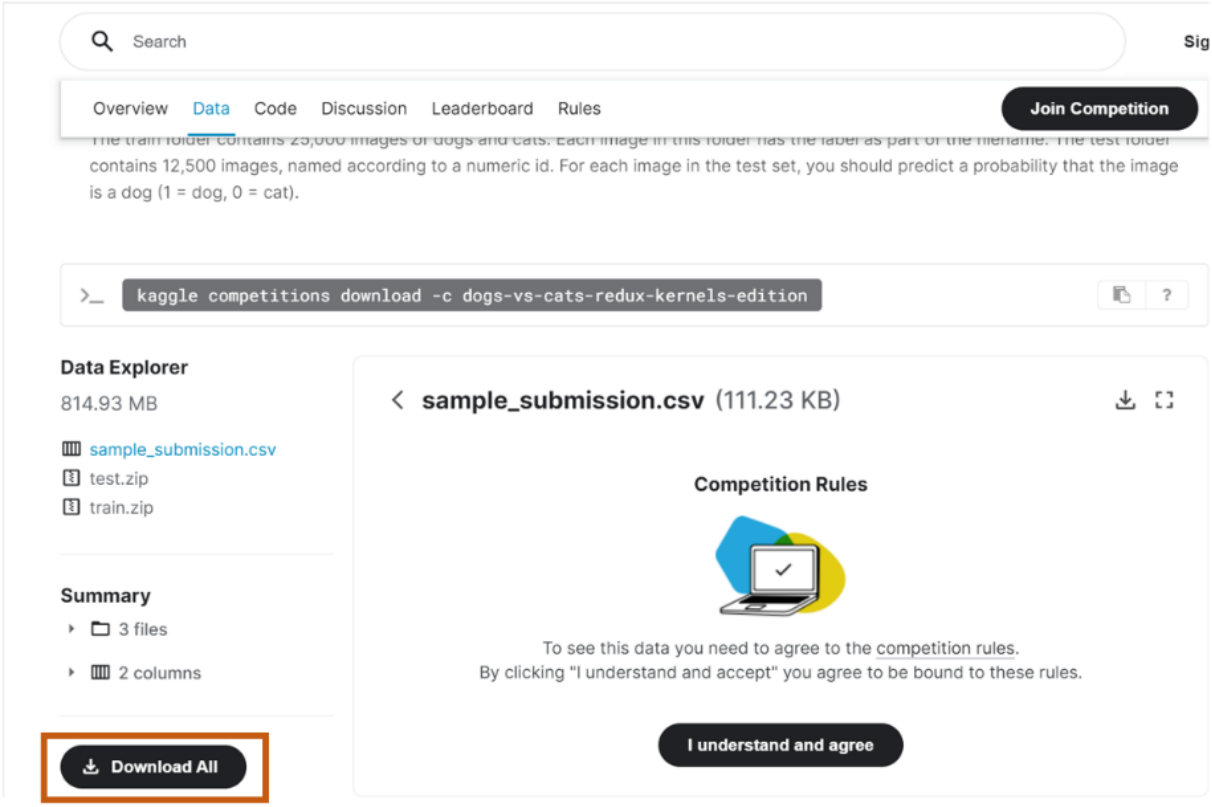


图 4-1 直接下载数据集

第二步，将猫狗数据集存放于 JPEGImages 文件夹下，本文将文件存储在 D:\MyDataset\JPEGImages 路径下，创建与图像文件夹对应的文件夹 Annotations，用来存储标注的.xml 文件，本文存储在 D:\MyDataset\Annotations 路径下。

4.2 使用 Labelimg 标注图片

第一步，打开虚拟环境 PaddleX 的 Open Terminal，进入 Windows 命令行窗口，先后分别输入命令<conda install pyqt>和命令<pip install labelimg>安装 Labelimg，命令运行结果如下图 4-2 所示。

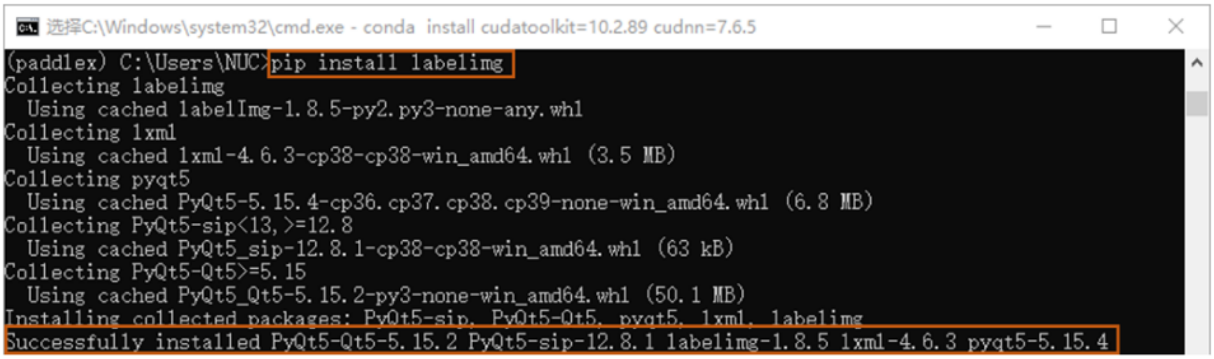


图 4-2 安装 Labelimg

第二步，启动 Labelimg，打开 Anaconda 虚拟环境 PaddleX 的 Open Terminal，进入 Windows 命令行窗口输入命令<labelimg>即可启动 Labelimg，点击左侧 Open Dir，选择需要标注的图像所在的文件夹(D:\MyaDataset\JPEGImages)打开，在右下角的 File List 对话框中会显示文件夹中所可以遍历的图像，进行遍历工作。在标记目标图片时右键单击图片，点击 Create RectBox 打开矩形框标注工具，如图 4-3 所示。

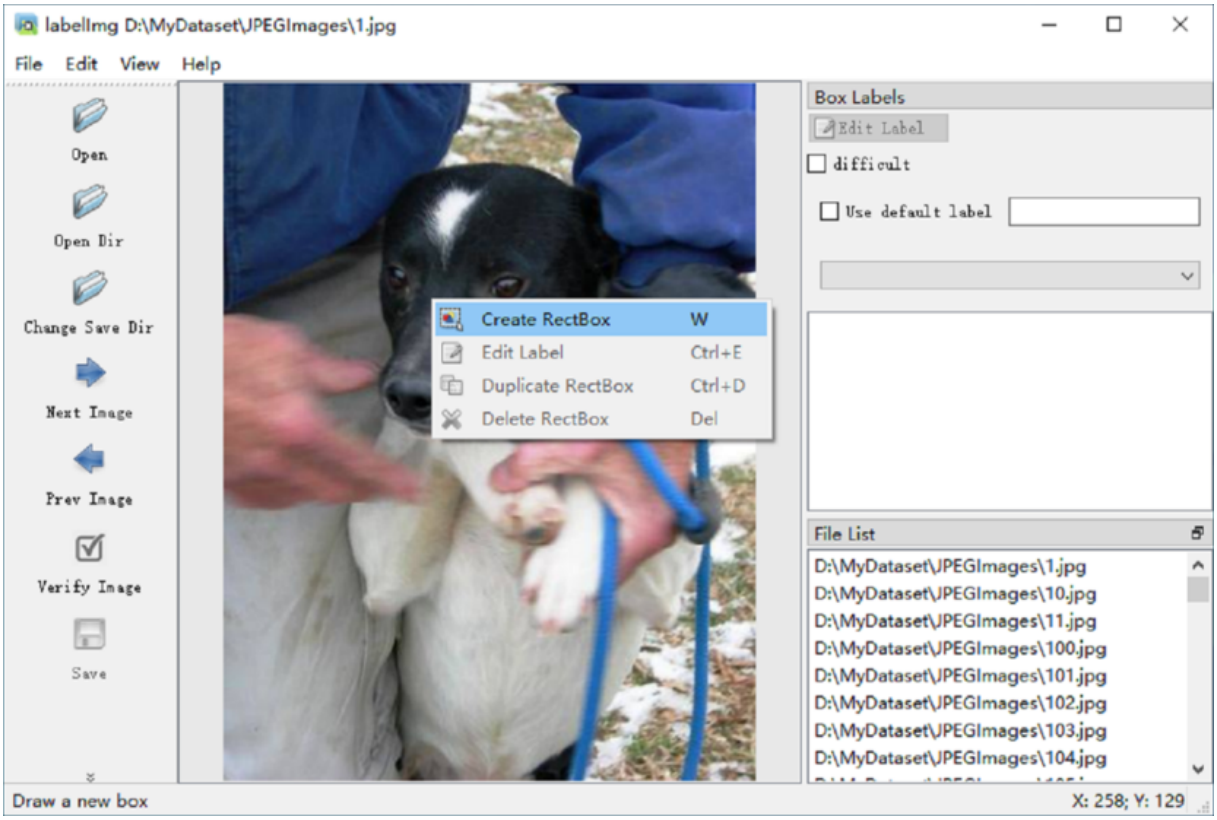


图 4-3 启动 Labelimg

第三步，使用拖拉的方式，用矩形框对目标物体进行标注，在弹出的对话框里写明对应的 label(当 label 已存在时，直接点击即可，注意 label 名字不要使用中文)，如下图 4-4 所示，标注完毕后再点击左侧的 Save，将标注后的.xml 文件保存在创建的 Annotations 文件夹中(D:\MyDataset\Annotations)。

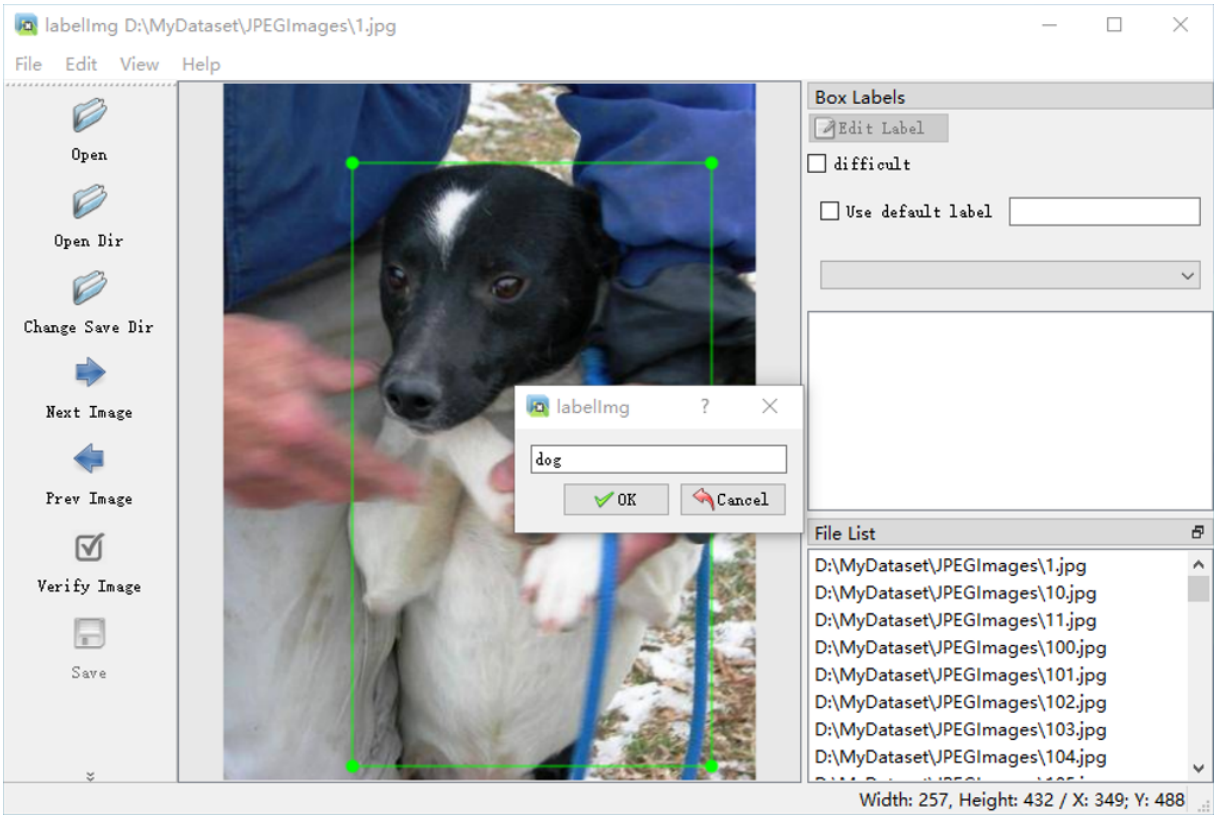

labelImg D:\MyDataset\JPEGImages\1.jpg
File
Edit
View
Help
Open
Open Dir
Change Save Dir
Next Image
Prev Image
Verify Image
Save
Box Labels
Edit Label
difficult
Use default label
labelImg
dog
OK
Cancel
File List
D:\MyDataset\JPEGImages\1.jpg
D:\MyDataset\JPEGImages\10.jpg
D:\MyDataset\JPEGImages\11.jpg
D:\MyDataset\JPEGImages\100.jpg
D:\MyDataset\JPEGImages\101.jpg
D:\MyDataset\JPEGImages\102.jpg
D:\MyDataset\JPEGImages\103.jpg
D:\MyDataset\JPEGImages\104.jpg
Width: 257, Height: 432 / X: 349; Y: 488

图 4-4 保存标注文件

5. 使用 PaddleX Python API 进行模型训练

5.1 数据集划分

标注完数据集后，在训练之前，需要将数据集划分为训练集、验证集和测试集三部分，安装 PaddleX 后，同样打开虚拟环境 PaddleX 的 Open Terminal 输入命令<PaddleX --split_dataset --format VOC --dataset_dir D:\MyDataset --val_value 0.2 --test_value 0.1>运行结果如下图 5-1 所示，同时会在 D:\ MyDataset 下生成 labels.txt,train_list.txt,val_list.txt 和 test_list.txt 分别用来存储类别信息，训练样本列表，验证样本列表和测试样本列表。

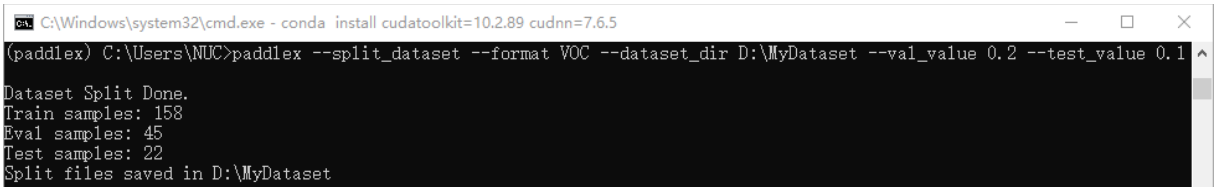


图 5-1 数据集划分

5.2 模型训练

PaddleX Python API 模式中所有模型训练都可以分为四个步骤

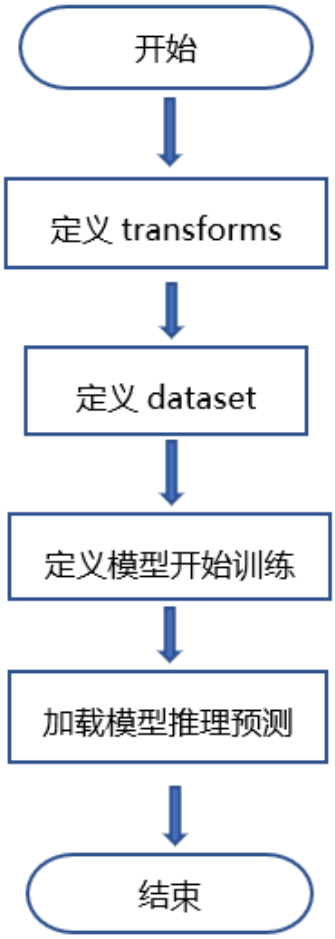
第一步，定义 transforms，用于定义模型训练、验证、预测过程中，输入图像的预处理和数据增强操作。

第二步，定义 dataset，用于定义模型要加载的训练、验证数据集。

第三步，定义模型开始训练，即选择需要的模型开始训练。

第四步，加载保存的模型推理预测。

模型训练步骤如流程图 5-1 所示。



流程图 5-1 PaddleX Python API 模型训练步骤

5.2.1 定义/验证图像处理流程 transforms

因为训练时加入了数据增强操作，因此在训练和验证过程中，模型的数据处理流程需要分别进行定义，如下代码清单 5-1 所示，代码在 train_transforms 中加入 MixupImage，RandomDistort，RandomExpand，以及 RandomCrop 等增强方式。

代码清单 5-1 定义/验证图像处理流程 transforms

```
#利用 Compose 类在训练时将图像处理增强操作进行组合
train_transforms = transforms.Compose([
    transforms.MixupImage(mixup_epoch=250), #对图像进行 mixup 操作
    transforms.RandomDistort(),                 #对图像进行随机像素内容变换
    transforms.RandomExpand(),                  #随机扩张图像
    transforms.RandomCrop(),                    #随机裁剪图像
    transforms.Resize(                          #调整图像大小
        target_size=608, interp='RANDOM'),   #RANDOM 为 resize 的插值方式。
    transforms.RandomHorizontalFlip(),          #对图像进行随机水平翻转,
    transforms.Normalize(),                     #对图像进行标准化
])
#利用 Compose 类在验证时将图像处理增强操作进行组合
eval_transforms = transforms.Compose([
    transforms.Resize(                          #调整图像大小,
        target_size=608, interp='CUBIC'),       #CUBIC 为 resize 一种插值方式
    transforms.Normalize(),                     #对图像进行标准化
])
```

5.2.2 定义 dataset 加载数据集

定义数据集，pdx.dataset.ImageNet 表示读取 ImageNet 格式的分类数据集，具体代码如下代码清单 5-2 所示。

代码清单 5-2 定义 dataset 加载数据集

```
#定义数据集 Dataset，采用 pdx.datasets.VOCDetection 来加载训练集
train_dataset = pdx.datasets.VOCDetection(
                #数据集所在的目录路径
                 data_dir='D:/MyDataset',
                #描述训练集对应标注文件的目录路径
                 file_list='D:/MyDataset/train_list.txt',
                #描述数据集包含的类别信息文件路径
                 label_list='D:/MyDataset/labels.txt',
                #训练集中每个样本的预处理
                 transforms=train_transforms,
                #是否需要对训练集样本进行打乱
                shuffle=True)
#定义数据集 Dataset，采用 pdx.datasets.VOCDetection 来加载验证集
eval_dataset = pdx.datasets.VOCDetection(
                #数据集所在的目录路径
                data_dir='D:/MyDataset',
                #描述验证集对应标注文件的目录路径
                file_list='D:/MyDataset/val_list.txt',
                #描述验证集包含的类别信息文件路径
```

```
    label_list='D:/MyDataset/labels.txt',
    #验证集中每个样本的预处理
    transforms=eval_transforms)
```

5.2.3 使用 YOLOv3 模型开始训练

PaddleX 内置了 20 多种分类模型，本文使用 YOLOv3 为预训练模型，num_epochs 设置为 300，batch_size 设置为 2，learning_rate 设置为 0.0000625 开始模型的训练，模型训练的 Python 代码清单 5-3 如下。

代码清单 5-3 使用 YOLOv3 模型开始训练

```
#初始化模型，进行训练
num_classes = len(train_dataset.labels)
#构建 YOLOv3 检测器，num_classes 为类别数，backbone 网络为 MobileNetV3_large
model = pdx.det.YOLOv3(num_classes=num_classes, backbone='MobileNetV3_large')
#YOLOv3 模型的训练接口
model.train(
    num_epochs=300,                    #训练迭代轮数
    train_dataset=train_dataset,       #训练数据读取器
    train_batch_size=2,                #训练数据 batch 大小
    eval_dataset=eval_dataset,         #验证数据读取器
    learning_rate=0.0000625,           #优化器的学习率
```

```
    lr_decay_epochs=[210, 240],             #优化器的学习率衰减轮数
    save_dir='output/MobileNetV3_large', #模型保存路径
    use_vdl=True)                           #是否使用 VisualDL 可视化
```

5.2.4 加载训练保存的模型预测

模型在训练中，会每间隔一定轮数保存一次模型，在验证集上评估效果最好的一轮会保存在 best_model 文件夹中，通过如下代码清单 5-4 可以加载模型，进行预测。

代码清单 5-4 加载训练保存的模型预测

```
#使用 paddlex 进行预测
import paddlex as pdx
#导入预测图片路径和模型
test_jpg = "D:/MyDataset/JPEGImages/1021.jpg"
model = pdx.load_model("output/MobileNetV3_large/best_model")
#predict 接口并未过滤低置信度识别结果，用户根据 score 值进行过滤
result = model.predict(test_jpg)
#可视化结果存储在./visualized_test.jpg 中
pdx.det.visualize(test_jpg,result,threshold=0.5,save_dir="./")
```

预测结果输出如

图 5-2 所示。

图 5-2 预测输出结果

5.3 使用 PaddleX 可视化客户端训练

PaddleX GUI 模式中所有模型训练都可以分为五个步骤。

第一步，准备数据和导入，将标注好的数据根据不同的任务类型导入可视化客户端。并将数据集按比例划分。

第二步，创建项目和任务，根据需求选择项目类型并创建项目。

第三步，参数配置，项目创建完成后，载入客户端加载数据集，根据实际需求对模型参数、训练参数、优化策略三方面进行参数配置，使得任务效果最佳。

第四步，模型训练，参数配置完成即可开始训练，训练过程中可以通过 VisualDL 查看模型训练过程参数变化，也可以对模型进行裁剪分析。

第五步，模型效果评估和发布，在模型评估界面可以查看训练后的模型效果，并使用模型进行推理测试，模型效果满意后，可以发布模型。

模型训练步骤如流程图 5-2 所示。

流程图 5-2 PaddleX GUI 模型训练步骤

5.3.1 加载数据集

第一步，使用 Labelimg 标注数据并保存在相应文件夹下后，在客户端新建数据集，定义数据集名称和数据集描述，再选择与数据集匹配的任务类型，选择标注数据对应的存储路径，将数据集导入，如图 5-3 所示。

图 5-3 新建并加载数据集

第二步，导入数据集后，客户端会自动校验数据集的数据及标注是否合乎规范，校验成功后，将数据集进行划分，按照实际需求的比例划分为训练集、验证集、测试集。数据集导入后，先点击主界面左侧的“我的项目”，再点击“新建项目”创建一个项目。根绝实际任务需求选择项目的任务类型，注意项目的任务类型要和数据集的任务类型一致。如图 5-4 所示

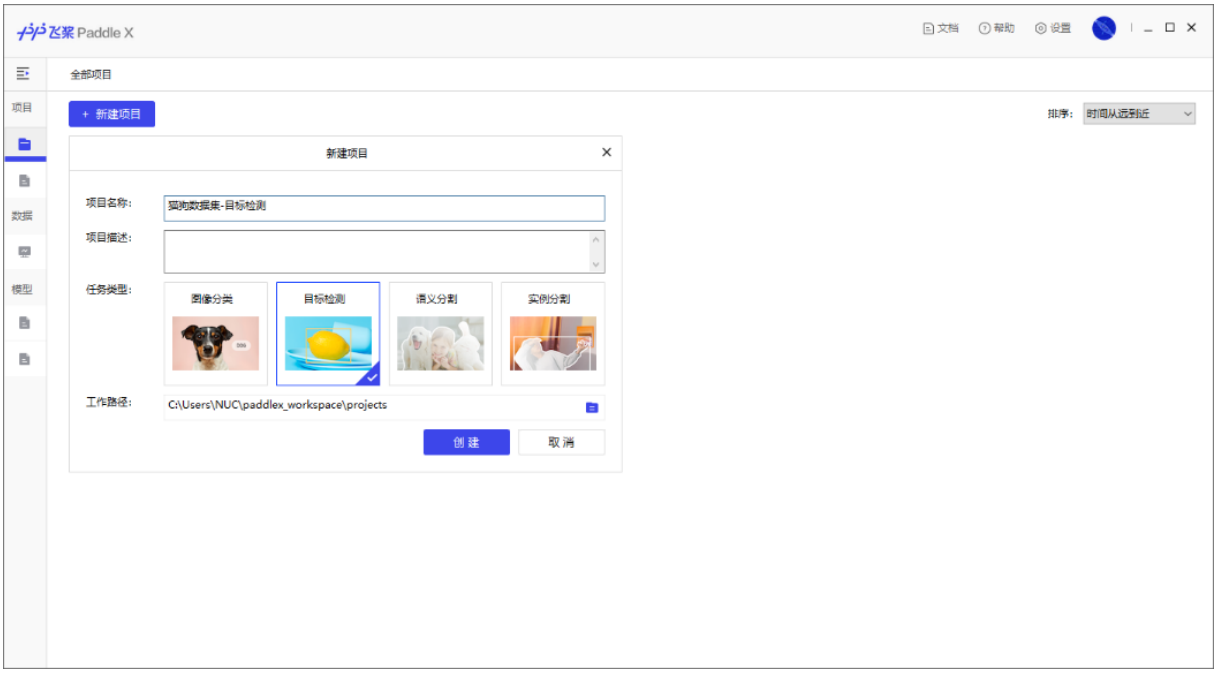


图 5-4 新建项目

5.3.2 配置参数

第一步，选择数据集，项目创建完成后，需要选择已载入客户端并校验后的数据集，如图 5-5 所示，点击下一步，进入参数配置界面。

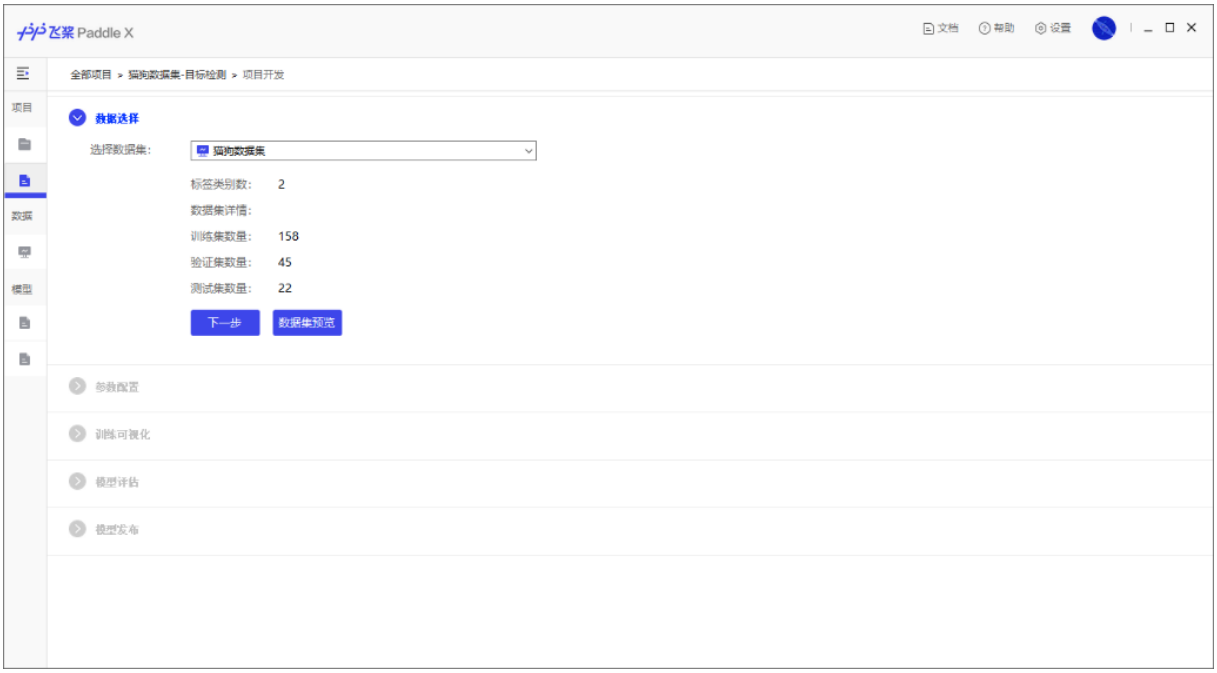


图 5-5 选择模型训练所需数据集

第二步，进行参数配置，主要是模型选择、模型参数、训练参数、优化策略四个部分，根据实际需求选择模型结构及其对应的训练参数和优化策略，以达到最佳效果。如图 5-6 所示。

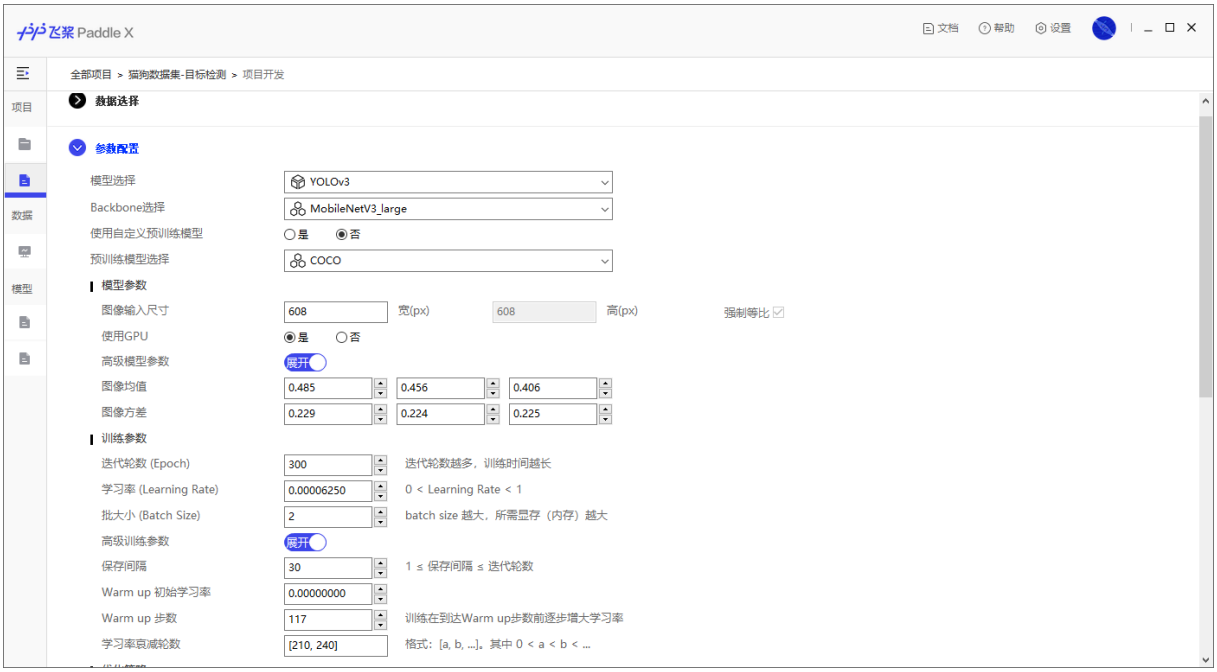


图 5-6 进行参数配置

5.3.3 启动训练

第一步，启动训练，参数配置完成后，模型开始训练并进行效果评估，如图 5-7 所示。

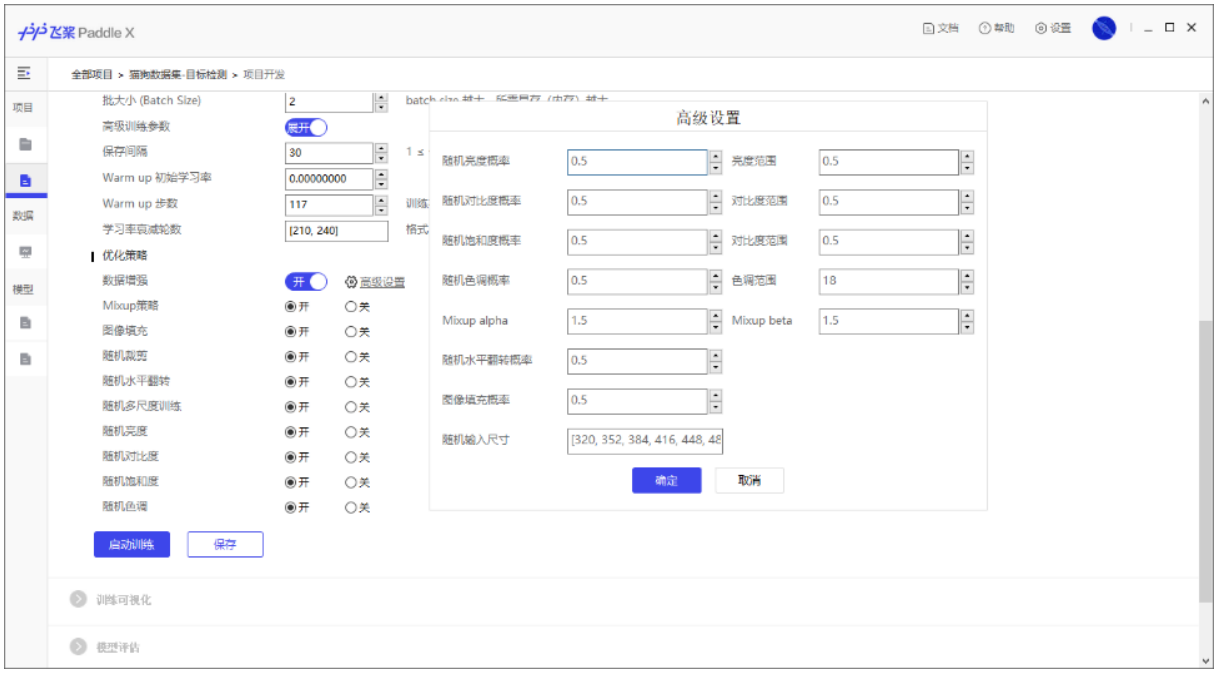


图 5-7 启动训练

第二步，在训练过程中，可以通过 VisualDL 查看模型训练过程中参数的变化、日志详情，以及当前最有的训练集和验证集训练指标。如图 5-8 所示。模型训练是最容易出错的步骤，经常遇到显存不够等问题，深度学习模型训练对显存的要求较高，可在 Cmd 命令终端执行 nvidia-smi 命令查看显存情况，请不要使用系统自带的任务管理器查看。

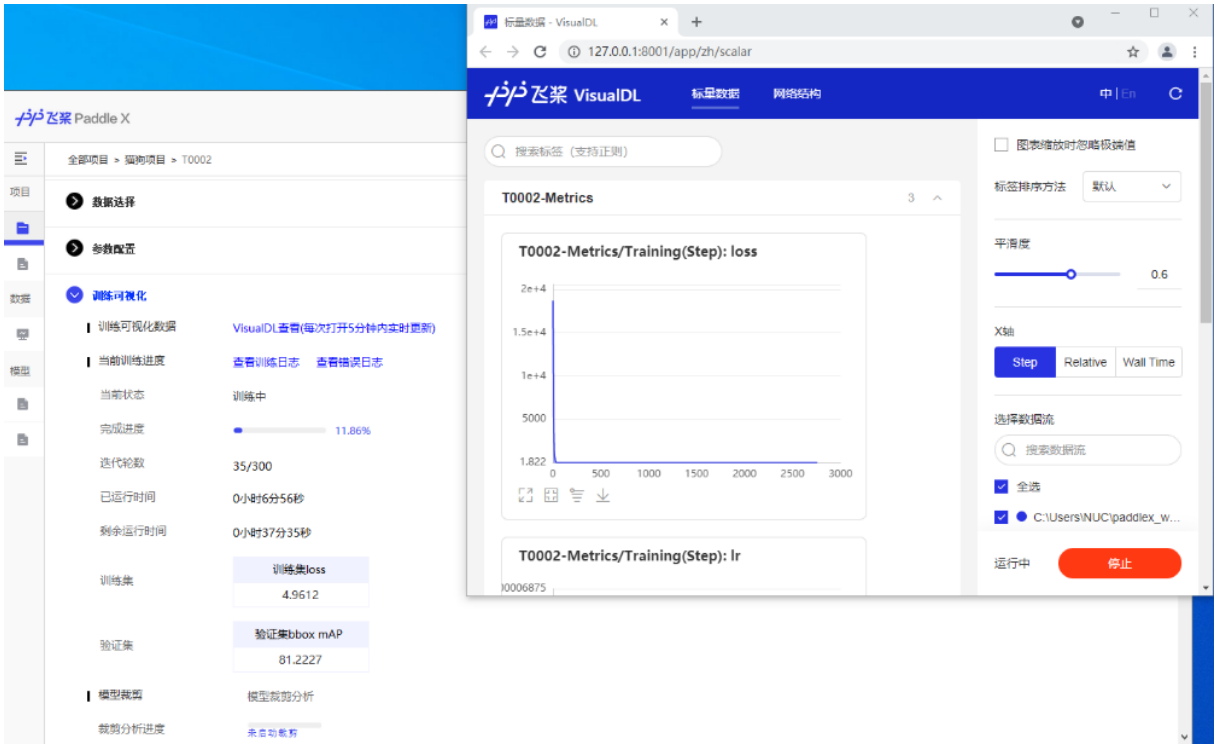


图 5-8 训练可视化

第三步，模型训练结束后，可以选择进入“模型剪裁分析”模块，剪裁过程对模型各卷积层的敏感度信息进行分析，根据各参数对模型效果的影响进行不同比例的裁剪，再进行精调训练获得最终裁剪后的模型。裁剪训练后的模型体积，计算量都会减少，并且可以提升模型在低性能设备的预测速度。或者直接进入“模型评估”模式。如

图 5-9 所示。

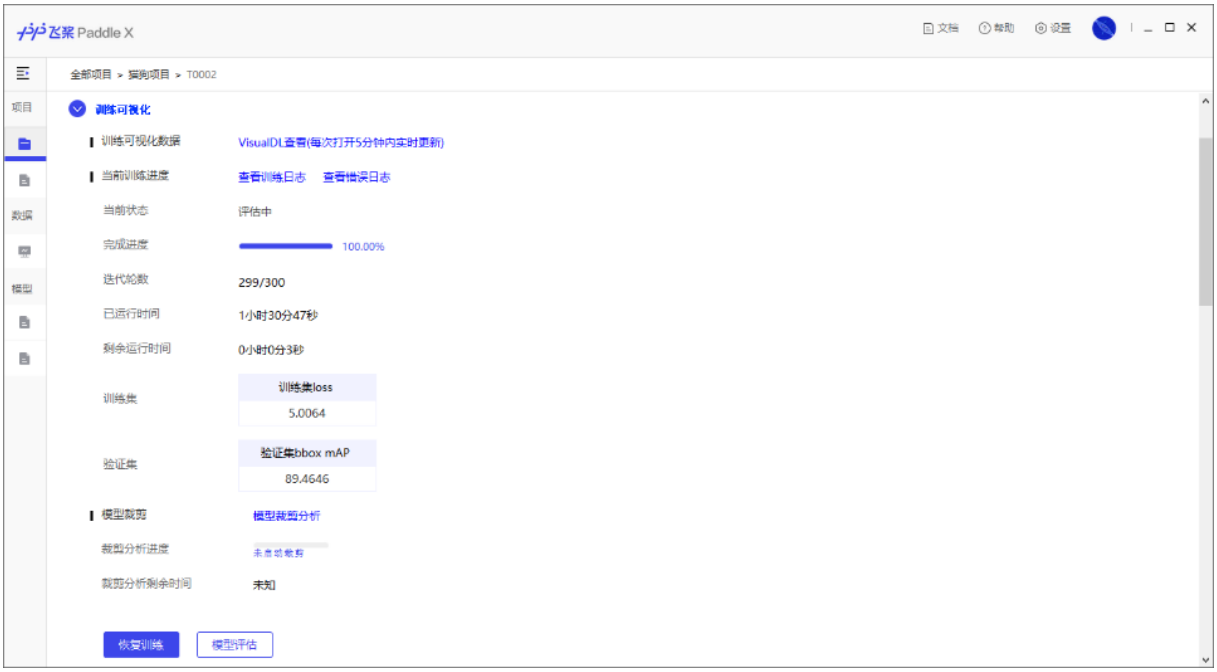


图 5-9 模型剪裁分析和模型评估

5.3.4 模型评估

第一步，在模型评估页面，可以查看训练后的模型效果。模型评估的方法包括混淆矩阵、精度、召回率等等。也可以自行选择 epoch 重新进行评估，如图 5-10 所示。

图 5-10 进入模型评估界面

第二步，预览测试图片和导出报告。如图 5-11 所示，可以选择数据集切分时的测试数据集，或者从本地文件夹中导入图片，使用训练后的模型进行测试。

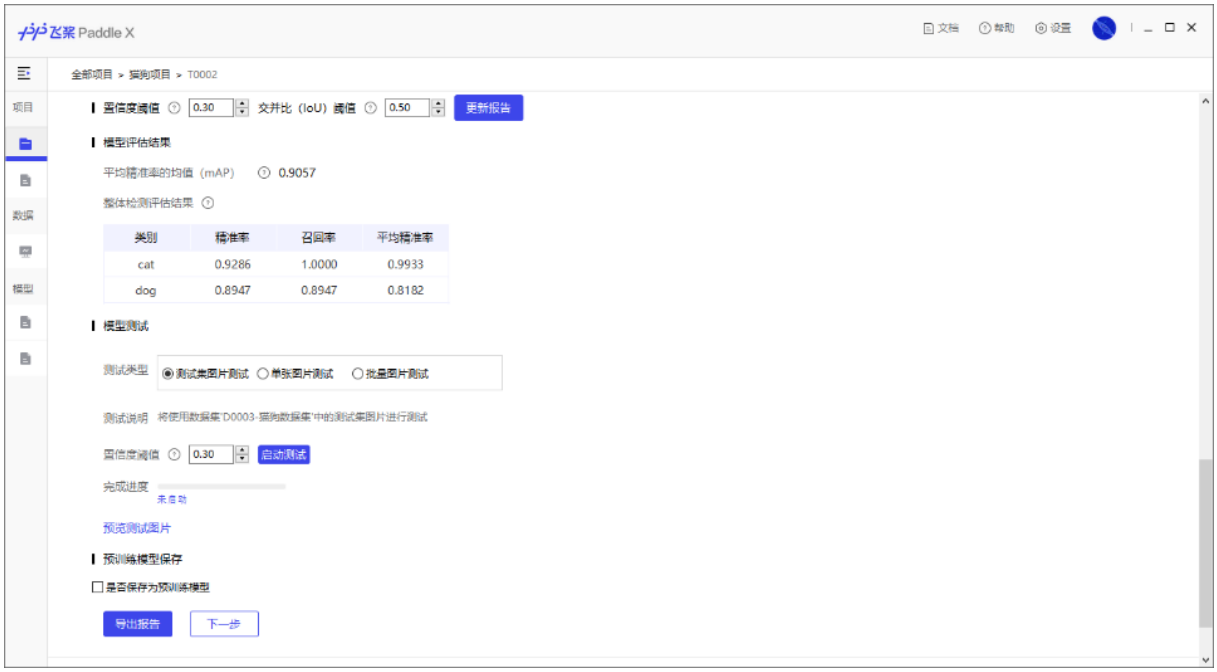


图 5-11 模型测试

第三步，查看模型预测效果如图 5-12 所示。首先点开“启动测试”按钮，等待完成进度条结束，再点击“预览测试图片”按钮对图片进行预测，单击图片查看测试精度和效果。根据测试结果可以决定是否将训练完的模型保存为预训练模型并进入模型发布界面，或者重新返回参数配置步骤重新调整参数。

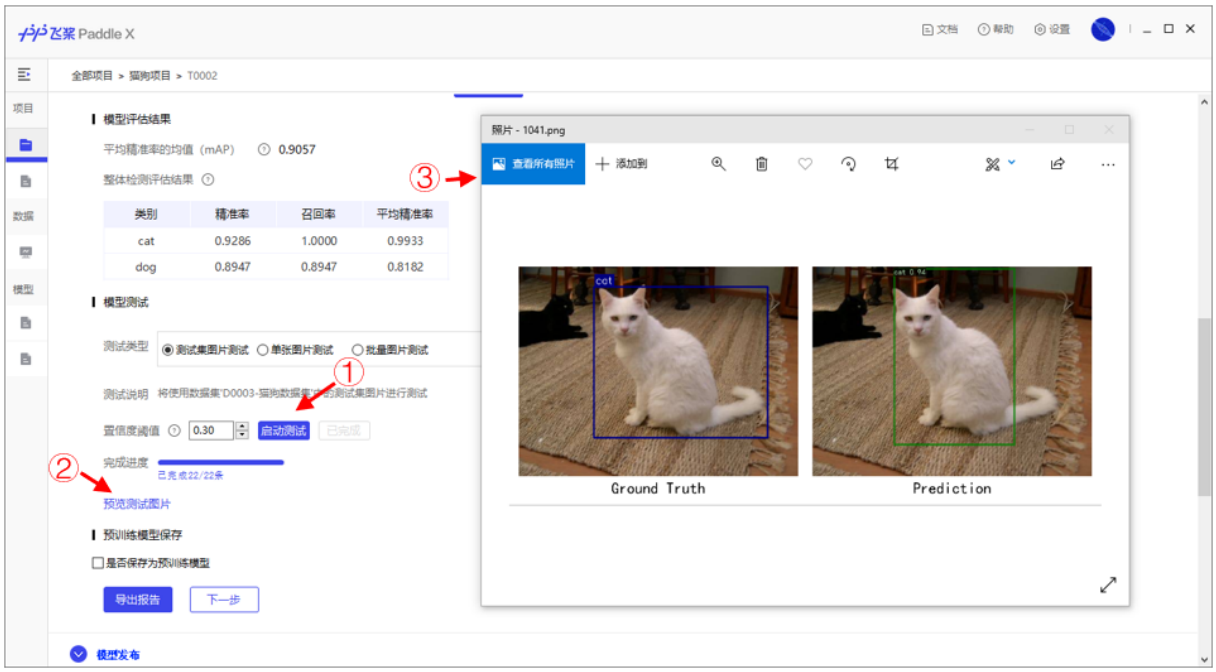


图 5-12 预览模型效果

5.3.5 模型发布

如果模型训练的效果满意，可以将模型进行发布，根据实际生产环境需求，将模型发布为需要的版本，如图 5-13 所示。

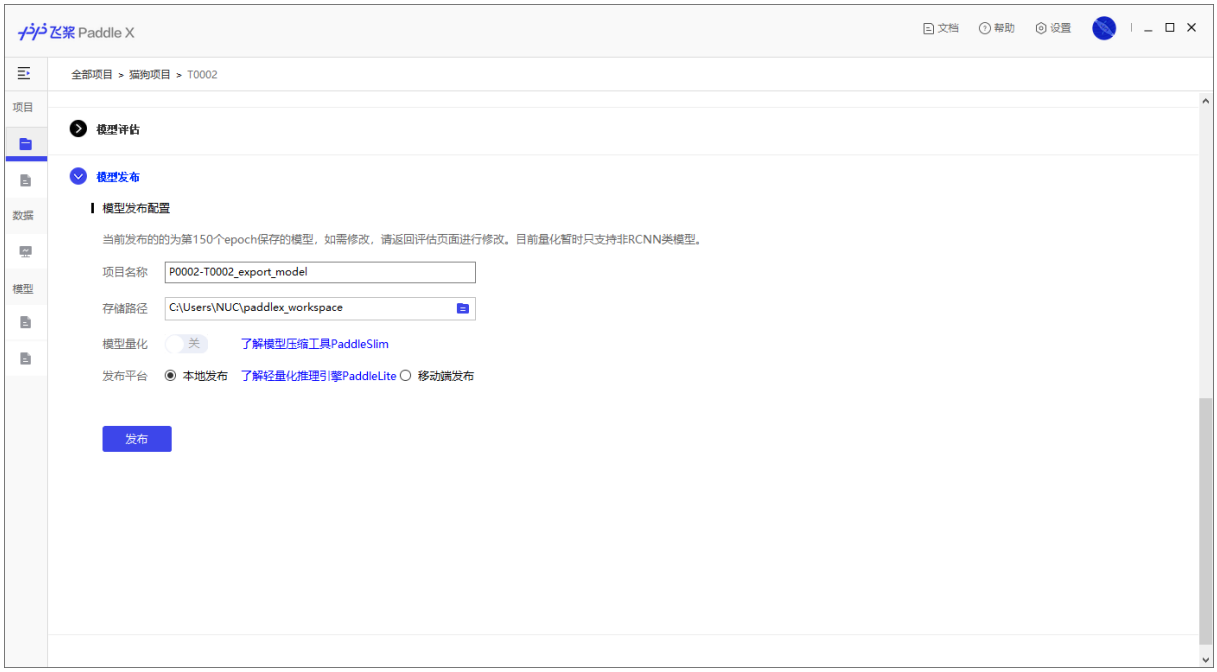


图 5-13 模型发布

5.3.6 模型预测

使用 PaddleX 客户端训练完毕模型后，导出模型，使用 python 脚本对模型进行推理预测，模型预测代码见代码清单 5-5。

代码清单 5-5 加载模型进行预测

```
#使用 PaddleX 进行预测
import paddlex as pdx
#导入预测图片路径和模型
test_jpg = "D:/MyDataset/JPEGImages/1140.jpg"
model = pdx.load_model("C:/Users/NUC/paddlex_workspace/P0002-
T0002_export_model/inference_model")
#predict 接口并未过滤低置信度识别结果，用户根据 score 值进行过滤
result = model.predict(test_jpg)
#可视化结果存储在./visualized_test.jpg 中
pdx.det.visualize(test_jpg,result,threshold=0.5,save_dir="./")
```

预测结果如图 5-12 所示。

图 5-14 使用 GUI 训练模型预测结果

6. 使用 OpenVINO™工具套件部署

6.1 OpenVINO™工具套件简介

OpenVINO™工具套件全称是 Open Visual Inference & Neural Network Optimization Toolkit，是英特尔®于 2018 年发布的开源工具包，专注于优化神经网络推理。OpenVINO™工具套件主要包括 Model Optimizer(模型优化器)和 Inference Engine(推理引擎)两个部分。Model Optimizer 是用于优化神经网络模型的工具，Inference Engine 是用于加速推理计算的软件包。如图 6-1 所示，即为 OpenVINO™工具套件的主要组成部分。

图 6-1 OpenVINO™工具套件

6.2 OpenVINO™工具套件安装

6.2.1 OpenVINO™工具套件下载和安装

下载并安装 OpenVINO™工具套件的具体步骤如下。

第一步，通过网址

https://software.intel.com/content/www/us/en/develop/tools/openvino-toolkit/download.html 进入 OpenVINO™工具套件官网下载页面，选择合适的版本，本文选择 2021.2 版本的 OpenVINO™工具套件，按照如图 6-2 所示选择，再点击 Download 按钮即可下载 OpenVINO™工具套件 2021.2 版本的安装程序。

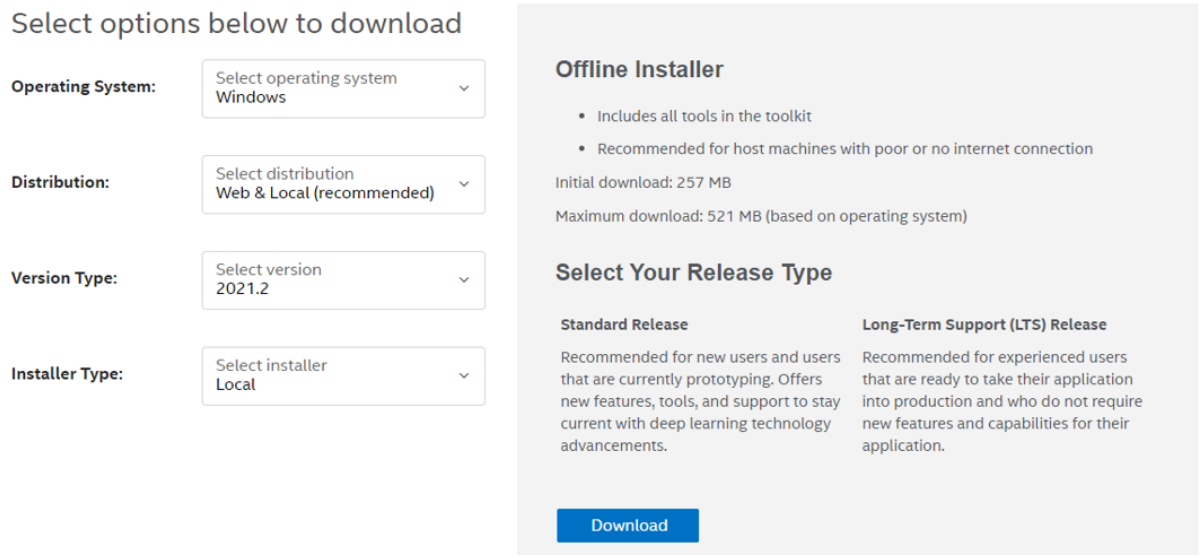


图 6-2 下载 OpenVINO™工具套件

第二步，找到 OpenVINO™工具套件的安装文件 w_openvino_toolkit_p_2021.2.185.exe，双击下载安装，安装步骤全部默认安装即可，如图 6-3 所示。

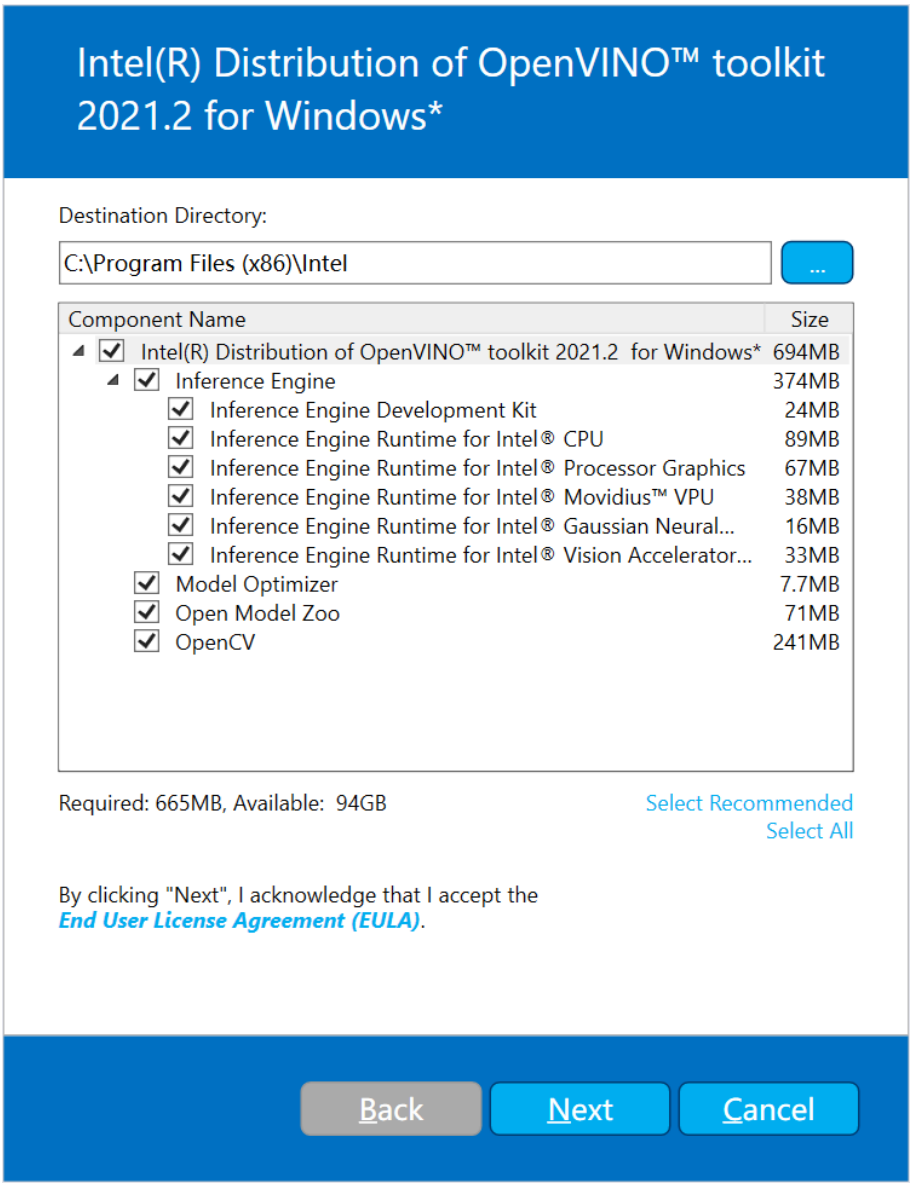


图 6-3 保持默认选项安装

第三步，安装过程中会有 CMake 和 Mircrosoft Visual Studio 依赖软件安装的提示，如图 6-4 所示，下面我们继续安装 CMake 和 Mircrosoft Visual Studio 软件。

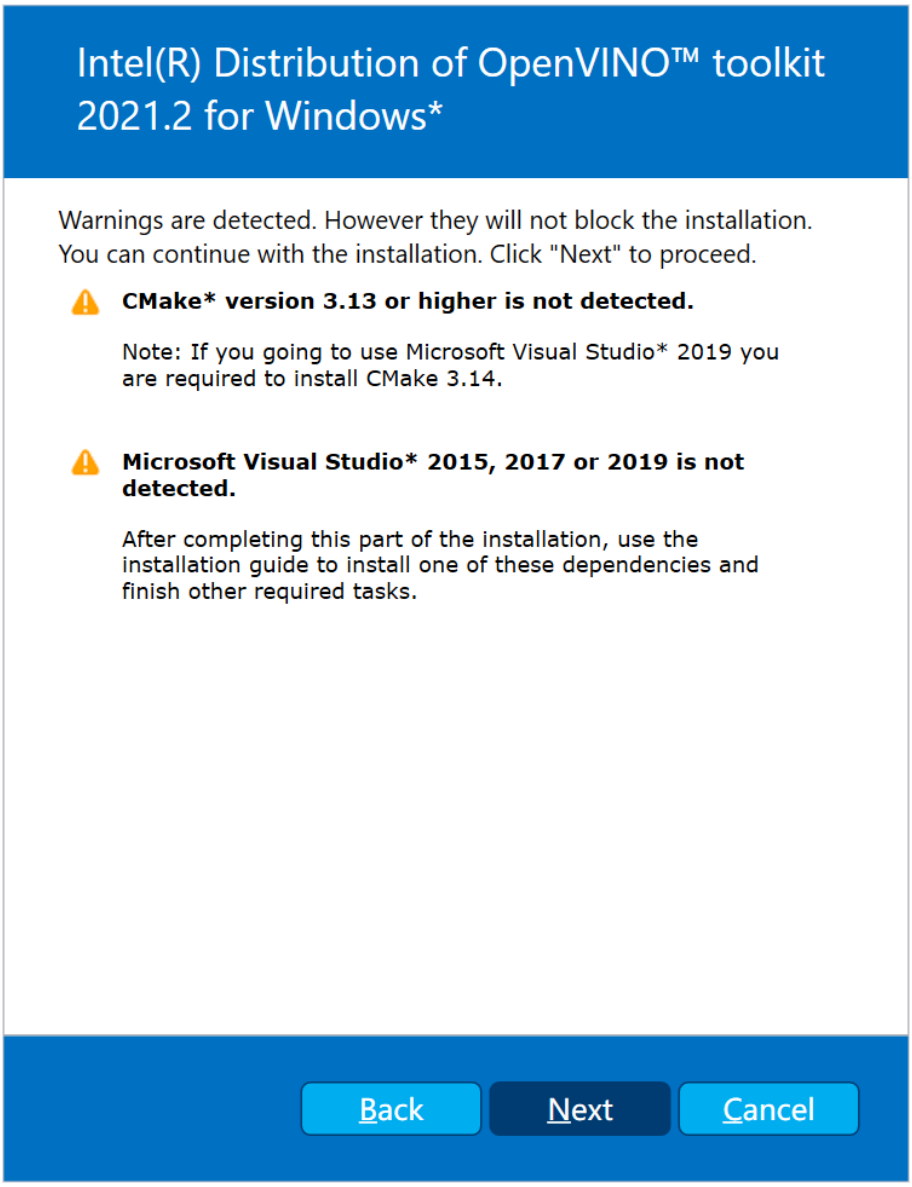


图 6-4 依赖软件提示

6.2.2 CMake 下载和安装

CMake 作为一个跨平台的 C/C++程序编译开源配置工具，在 OpenVINO™工具套件的应用中，CMake 用来管理 OpenVINO™工具套件中的演示程序(Demos)和范例程序(Samples)。

下载并安装 Cmake 的步骤如下所示。

第一步，通过网址 https://cmake.org/download/ 进入 CMake 官网下载界面，下载安装文件，选择的 CMake 版本大于等于 3.4 版本即可，本文的版本选择为 cmake-3.20.2-windows-x86_64.msi，如图 6-5 所示

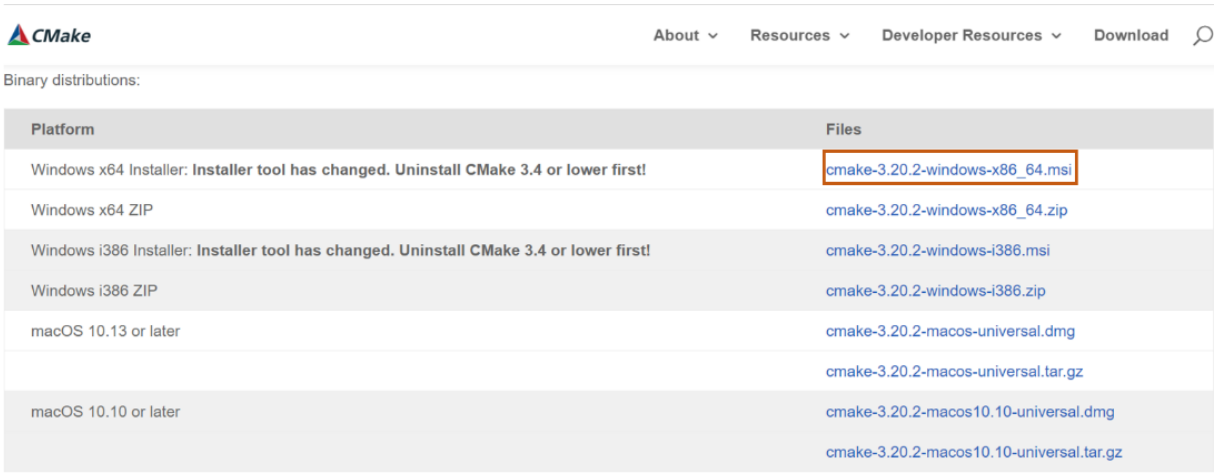


图 6-5 下载 CMake

第二步，双击安装文件，默认选项完成安装，在 Install Options 页面选择 Add Cmake to the system PATH for all users 将 CMake 添加到系统变量 PATH 中。如图 6-6 所示。

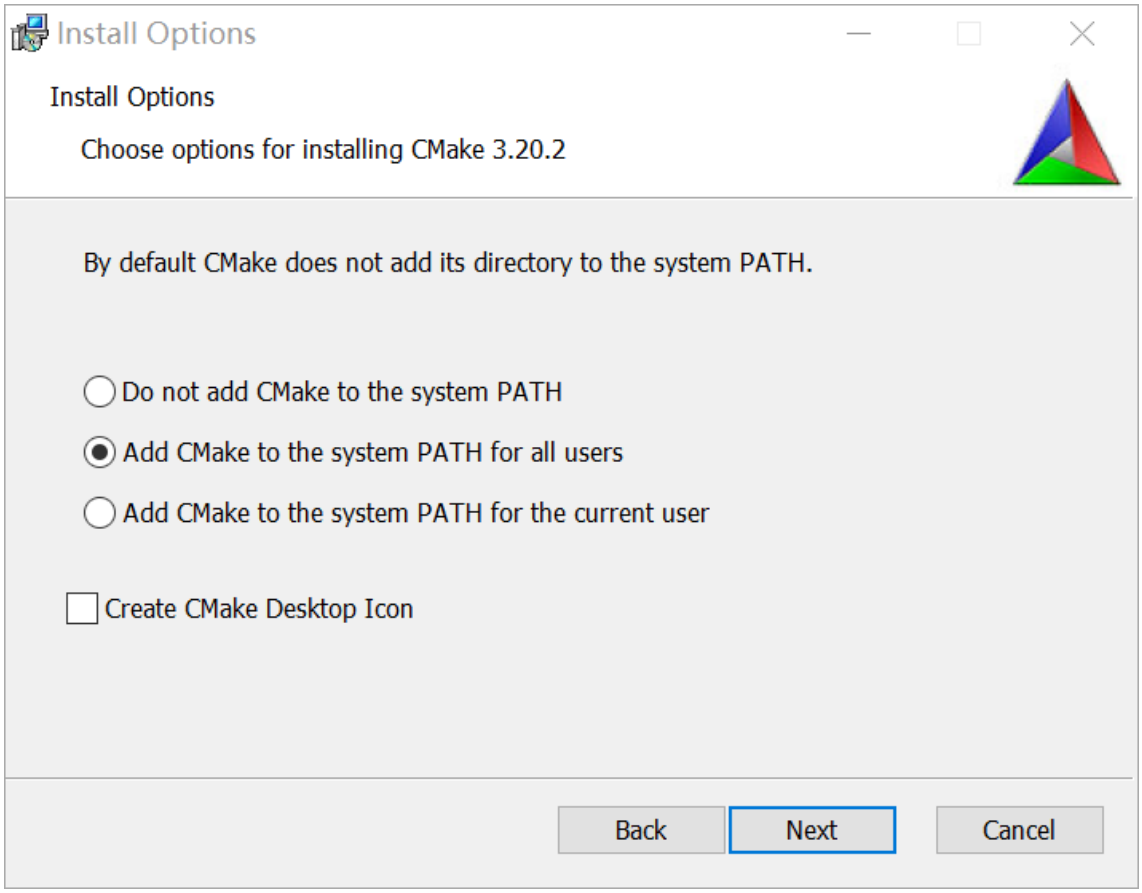


图 6-6 将 CMake 添加到环境变量中

6.2.3 Mircrosoft Visual Studio 下载和安装

OpenVINO™工具套件支持 Mircrosoft Visual Studio 2015、2017 和 2019。由于 Mircrosoft Visual Studio 2017 是目前 Windows 操作系统下应用最广泛的 C++ IDE，本文选择使用 Mircrosoft Visual Studio 2017 版本。

Mircrosoft Visual Studio 2017 安装步骤如下。

第一步，通过网址 https://visualstudio.microsoft.com/zh-hans/vs/older-downloads/ 进入 Mircrosoft Visual Studio 旧版本下载地址，单击 2017，在展开的下载选项中点击“下载”按钮进入 Mircrosoft Visual Studio 2017 下载页面，在左侧选择 Visual

Studio 2017(version 15.9),在右侧的选择页面中选择 Visual Studio Community 2017(version 15.9)，单击 Download 下载，如所示。

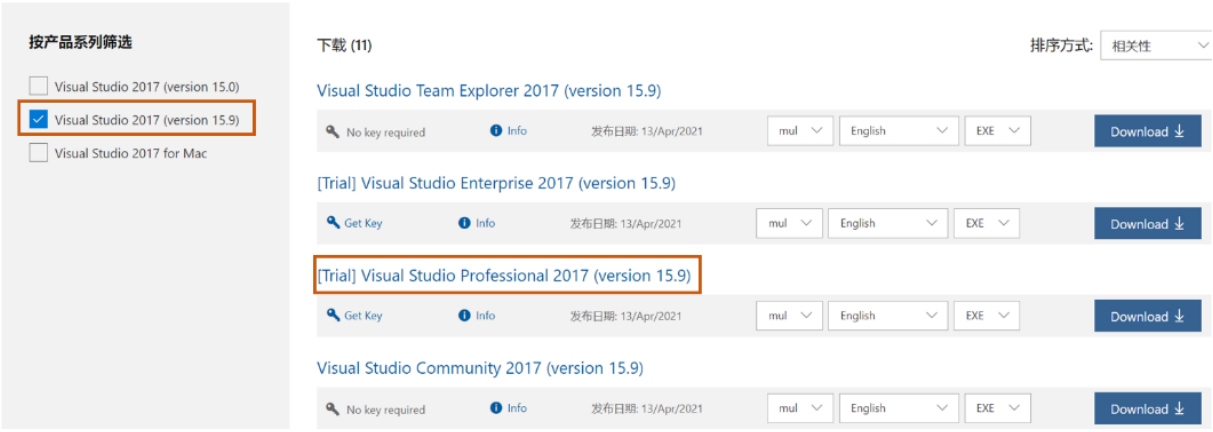


图 6-7 下载 Mircrosoft Visual Studio 2017

第二步，找到安装文件双击打开，在安装配置中选择“.NET 桌面开发”、“使用 C++的桌面开发”、“通用 Windows 平台开发”三个选项后，再选择右下角的“安装”按钮开始安装，如图 6-8 所示。

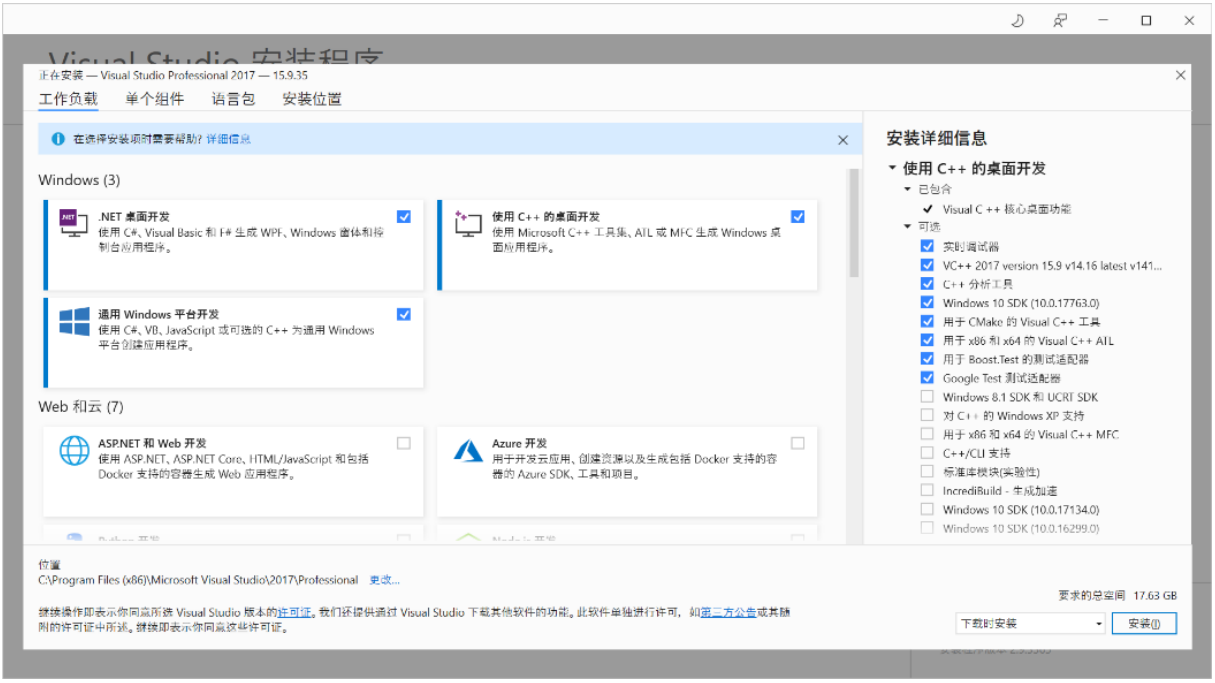


图 6-8 安装 Mircrosoft Visual Studio 2017

7. 使用 OpenVINO™工具套件部署 YOLOv3 模型

7.1 安装 Paddle2ONNX 和 ONNX

PaddleX 支持将训练好的深度学习模型通过 OpenVINO™工具套件对模型优化部署，在初始化 OpenVINO™工具套件使用环境，安装完毕 OpenVINO™工具套件相关依赖后即可进行加速部署。

ONNX(Open Neural Network Exchange)是针对机器学习所设计的开源格式，不同的深度学习框架可以通过 ONNX 格式存储模型并进行转换和交互。同样的，Paddle 模型可以通过 ONNX 格式使用 OpenVINO™工具套件进行推理。

安装 Paddle2ONNX 和 ONNX 首先打开 Anaconda 的虚拟环境 PaddleX，打开 Open Terminal 进入 Windows 命令行窗口，输入命令<pip install paddle2onnx==0.4 onnx onnxruntime -i https://mirror.baidu.com/pypi/simple>安装 Paddle2ONNX 0.4 和 ONNX 1.9.0 版本。如图 7-1 所示。

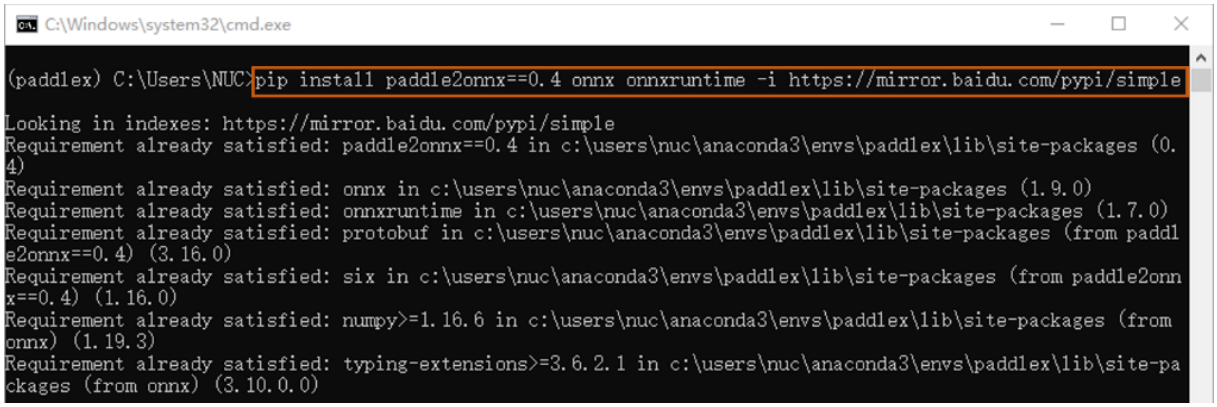


图 7-1 安装 Paddle2ONNX 和 ONNX

7.2 将 PaddleX 模型转换成 OpenVINO 模型

将 PaddleX 模型转换成 OpenVINO 模型可以分为四个步骤。

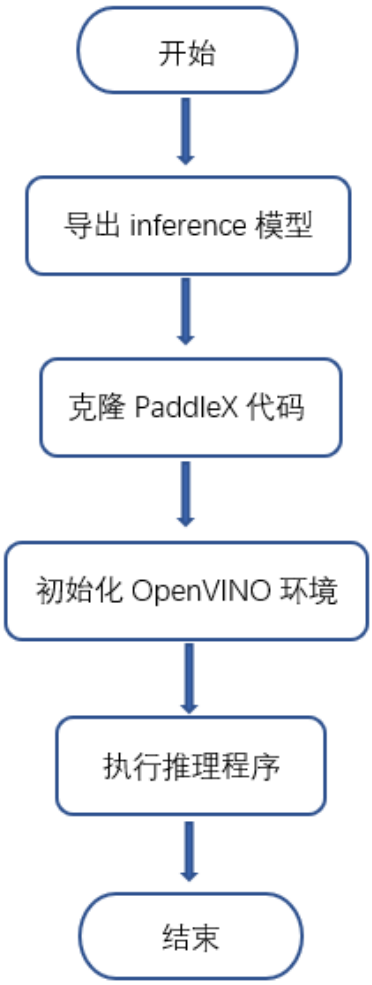
第一步，将 Paddle 模型导出为 inference 模型。

第二步，通过 git 克隆 PaddleX 代码仓到本地。

第三步，初始化 OpenVINO 环境，转换代码。

第四步，加载 OpenVINO 模型，执行推理程序。

模型训练步骤如流程图 7-1 所示。



流程图 7-1 PaddleX 模型转换 OpenVINO 模型步骤

7.2.1 导出 inference 格式模型

将 paddle 模型导出为 inference 格式模型步骤如下。

第一步，之前通过 PaddleX 训练得到 YOLOv3 模型后，导出模型 best_model，新建一个用于存放推理文件的文件夹，本文建立文件夹 pp_ov，文件路径为 D:\pp_ov，将 best_model 复制到 pp_ov 文件夹中，如图 7-2 所示。

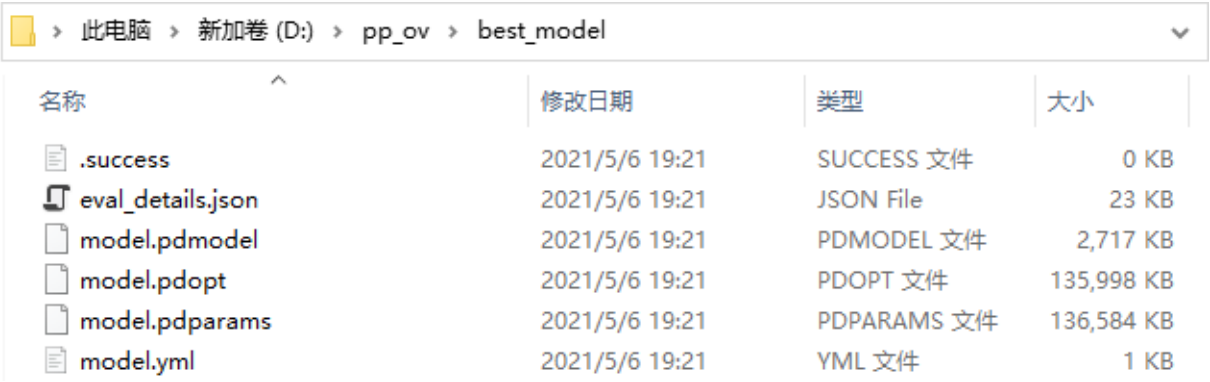


图 7-2 best_model 文件

第二步，进入 D:\pp_ov 路径的 Windows 命令行窗口，通过命令<activate paddlex>进入 paddlex 虚拟环境，输入命令<PaddleX –export_inference –model_dir=./best_model –save_dir=./inference_model –fixed_input_shape=[608,608]>，将模型导出为 inference 模型，inference 模型文件中包括.success、__model__、__params__和 model.yml 四个文件。如图 7-3 所示。

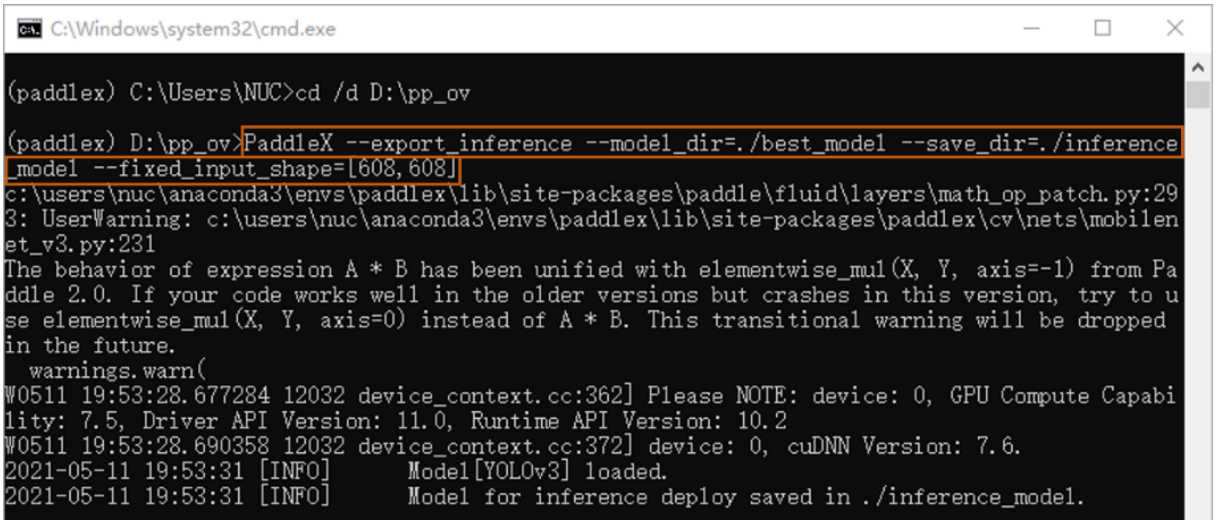


图 7-3 导出 inference 模型

7.2.2 初始化 OpenVINO 环境

初始化 OpenVINO 运行环境步骤如下。

第一步，从 GitHub 克隆代码到本地，在 D:\pp_ov 文件空白处右键打开 Git Bash Here，输入命令<git clone https://github.com/PaddlePaddle/PaddleX.git>下载 PaddleX 代码仓到 pp-ov 文件内。如图 7-4 所示。

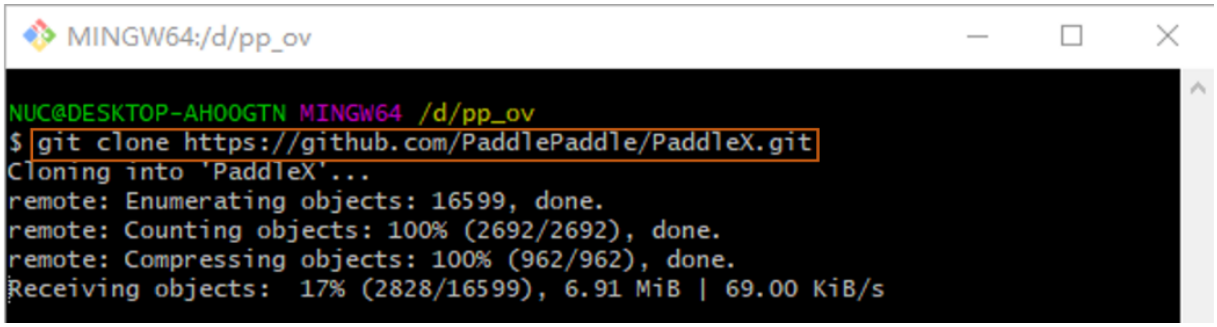


图 7-4 克隆 PaddleX 代码到本地

第二步，通过命令

<"c:\Program Files (x86)\Intel\openvino_2021.2.185\bin\setupvars.bat">，初始化 OpenVINO 环境，如图 7-5 所示。

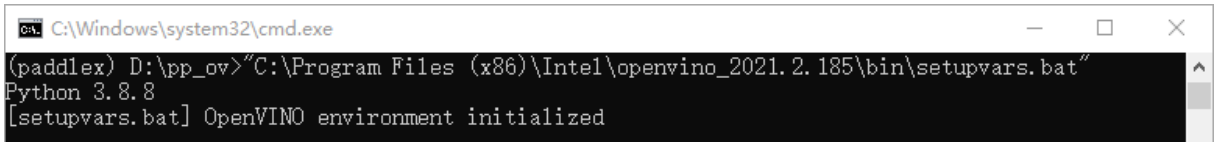


图 7-5 初始化 OpenVINO 环境

第三步，转换代码，通过输入命令<python converter.py --model_dir inference_model --save_dir openvino_model --fixed_input_shape [608,608]>将 inference 格式模型转换为 OpenVINO 模型。如图 7-6 所示。

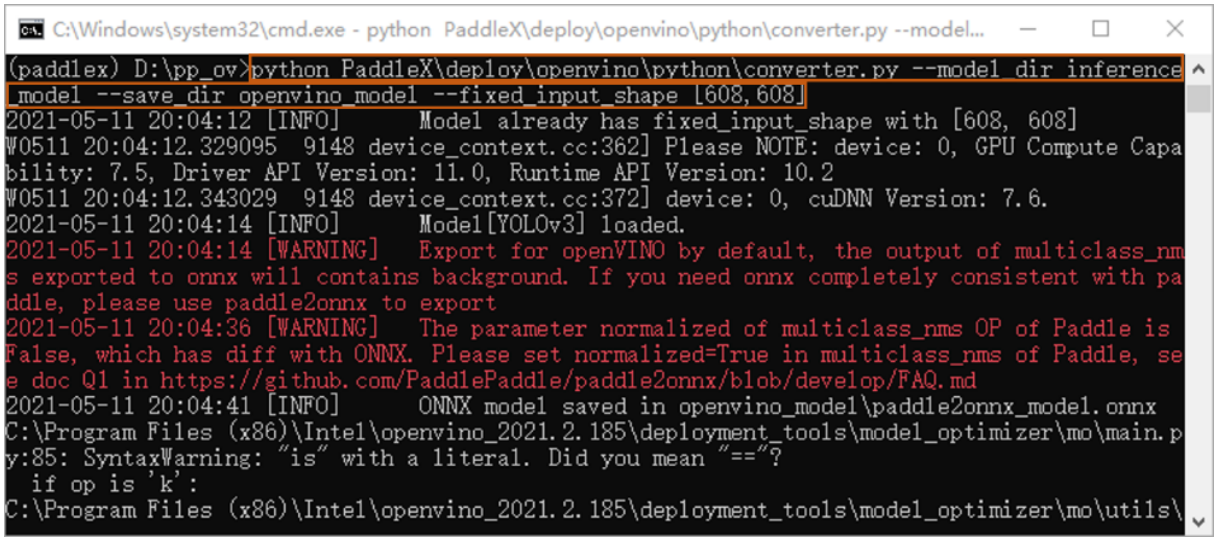


图 7-6 转换代码

7.2.3 执行推理程序

通过在 paddlex 虚拟环境中输入命令

<python PaddleX\deploy\openvino\python\demo.py -m openvino_model\paddle2onnx_model.xml -i D:\MyDataset\JPEGImages\1140.jpg -c inference_model\model.yml>，加载 OpenVINO 模型，执行推理程序。如图 7-7 所示。

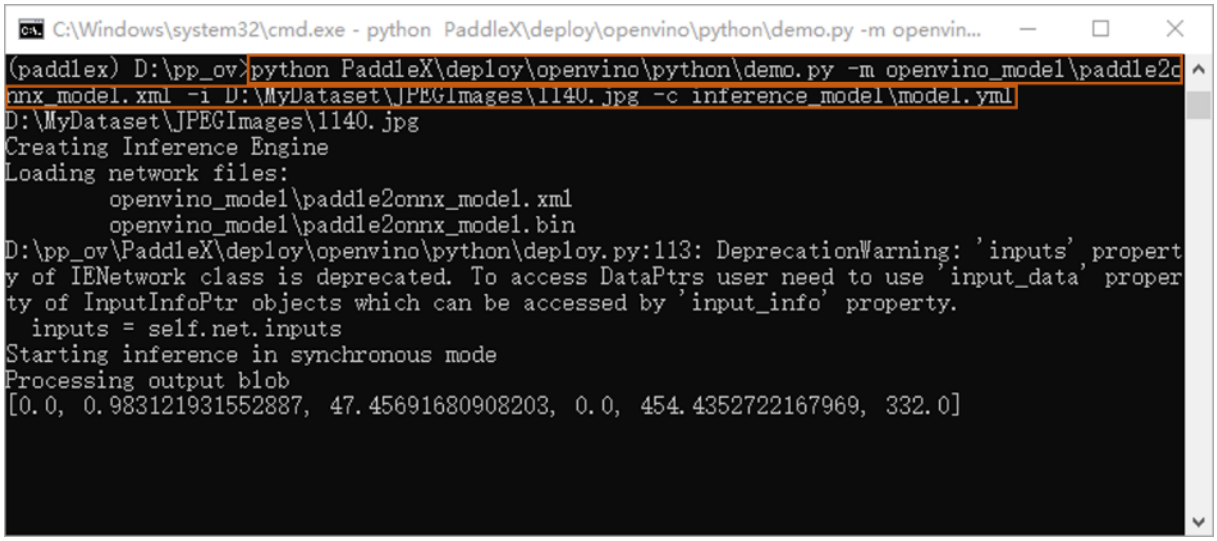


图 7-7 执行推理程序

输出结果如图 7-8 所示。

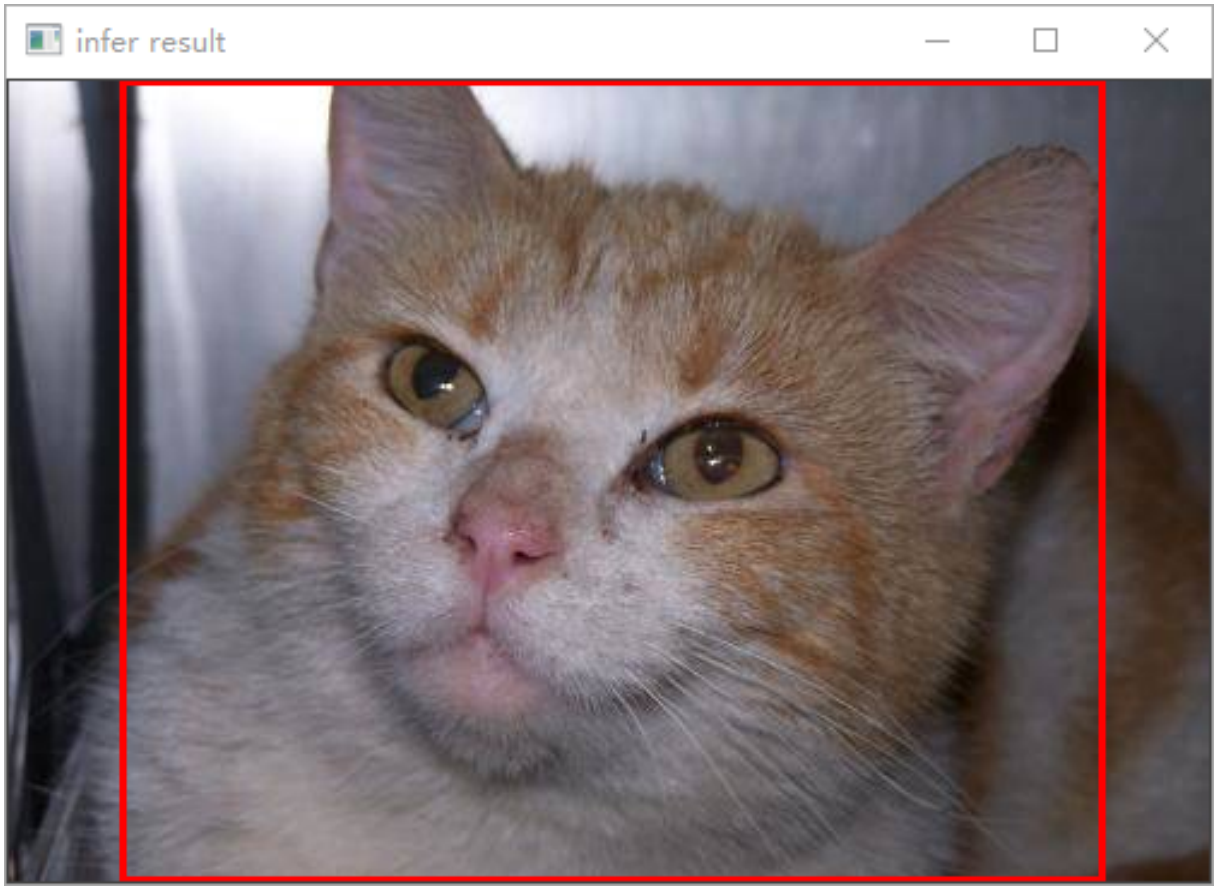


图 7-8 推理结果

7.3 YOLOv3 IR 模型性能测试

7.3.1 推理计算性能评价指标

响应延迟(Latency)和吞吐量(Throughput)是评价 AI 模型推理优化性能的两个重要指标，响应延迟和吞吐量通常用于测量网络性能和提高加载时间。响应延迟可视为执行一次推理计算所需的时间，而吞吐量可视为在一个单位时间内执行的推理计算次数。换句话说，延迟衡量的是推理计算的速度，响应延迟越小，推理计算的越快；吞吐量是处理多少数据，在单位时间内，吞吐量越高性能越好。

7.3.2 性能测试

本文将使用 YOLOv3 模型为 5.3.5 节训练并发布的，YOLOv3-MobileNetV3_large-608◊608，其转换的 YOLOv3 IR 模型中 paddle2onnx_model.xml 和 paddle2onnx_model.bin 来做性能测试。具体步骤如下。

第一步，进入 openvino 默认安装路径的 C:\Program File (x86)\Intel\openvino_2021.2.185\deployment_tools\tools\benchmark_tool 的 Windows 命令行窗口，通过命令<activate paddlex>命令打开 paddlex 虚拟环境，再通过命令<"c:\Program Files (x86)\Intel\openvino_2021.2.185\bin\setupvars.bat">初始化 OpenVINO 环境，如下图 7-9 所示。

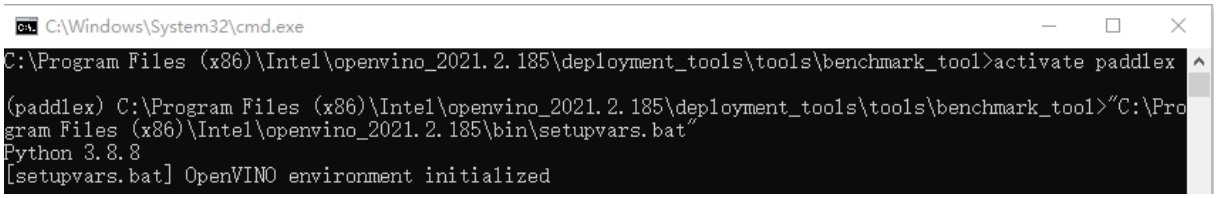


图 7-9 初始化 OpenVINO 环境

第二步，在 Windows 命令行窗口输入命令<python benchmark_app.py -m D:\pp_ov\openvino_model\paddle2onnx_model.xml -d CPU>指定 CPU 进行性能测试，如图 7-10 所示。

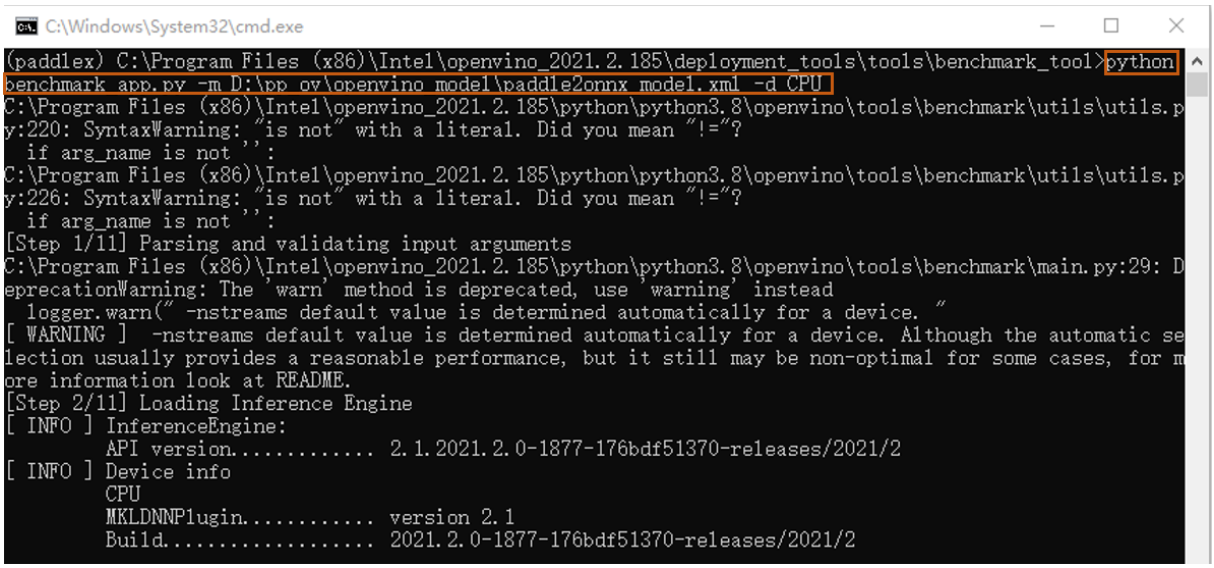


图 7-10 指定 CPU 进行性能测试

测试结果如图 7-11 所示。

图 7-11 CPU 性能测试结果

第三步，在 Windows 命令行窗口输入命令<python benchmark_app.py -m D:\pp_ov\openvino_model\paddle2onnx_model.xml -d GPU>指定 iGPU 进行性能测试，如图 7-12 所示。

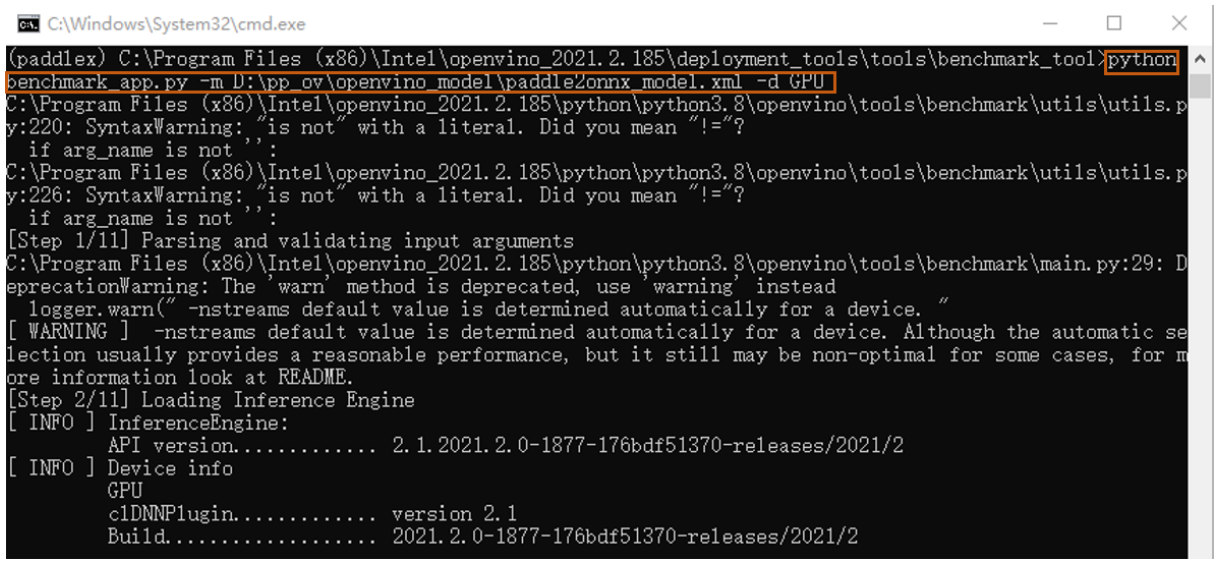


图 7-12 指定 iGPU 进行性能测试

测试结果如图 7-13 所示。

图 7-13 iGPU 性能测试结果

7.3.3 性能对比

本文基于幻影峡谷的中央处理器第 11 代英特尔®酷睿™i7-1165G7 和英特尔® Iris® Xe Graphics 集成显卡，所得到的 YOLOv3 IR 模型性能测试结果见表 7-1。

表 7-1 YOLOv3 IR 模型性能测试结果

执行硬件	吞吐量(FPS)
i7-1165G7	10.56
Xe Graphics	19.56

由上述数据可以看到，YOLOv3 模型经过 OpenVINO™工具套件优化后，在 Iris® Xe Graphics 集成显卡上的吞吐量可以达到 19.56FPS，该速度完全满足大部分 AI 工程实践应用。

8. 总结

本文根据 PaddleX 提供的 Python API 和 GUI 两种模型训练模式分别对猫狗数据集训练并使用 OpenVINO™工具套件进行了优化加速推理。对于相对比较复杂的深度学习模型训练而言，PaddleX Python API 通过定于 transforms、dataset 和模型三个步骤，使用并不复杂的 Python 代码脚本已经可以获得效果比较好的深度学习模型。而且 PaddleX GUI 无代码训练模式把模型训练的复杂度降得更低，PaddleX 将飞桨 CV 方向的工具组件产品进行了集成，并且提供了统一的 API 接口，精选了工程应用中成熟的模型，使深度学习模型开发更为便捷和统一，并且依然能得到精度较好的推理结果。

在模型推理部署方面，本文也详细的介绍了基于 OpenVINO™工具套件的安装和使用，特别是在使用 OpenVINO™工具套件对模型进行加速优化后的 YOLOv3 模型，在英特尔®11 代 i7-1165G7 上可以跑到 10.56FPS 的性能，在 Iris® Xe Graphics 集成显卡上可以跑出 19.56FPS 的性能，模型的推理计算性能有了很显著的提升，完全能满足常见的 AI 应用需求，并且在一定程度上能够摆脱对独立显卡的依赖。

在工程应用中，无代码模型训练模式已经成为了趋势，无代码模型训练方式可以很好的解决深度学习技术学习、调优成本高等问题，作为用户可以完全在本地进行开发流程，解决了数据及算法的安全问题，使企业开发人员能够最快完成模型的开发和部署。而 OpenVINO™工具套件的应用可以在无代码快捷简单训练模型的基础上，加速对模型在嵌入式平台和边缘智能端的推理和部署，在提升性能的同时能够摆脱对独立显卡的依赖。由此可见，像 PaddleX 和 OpenVINO™工具套件这样的深度学习模型训练和部署的工具，必然会在未来的 AI 工程应用中大放异彩。源于产业实践，关注行业需求，节约生产成本，加快创意落地才是王道，才能更快更好的赋能于产业实践。

如欲了解更多 OpenVINO™ 开发资料，

请扫描下方二维码，

我们会把最新资料及时推送给您。