YOLOv8实例分割实战-Android手机部署

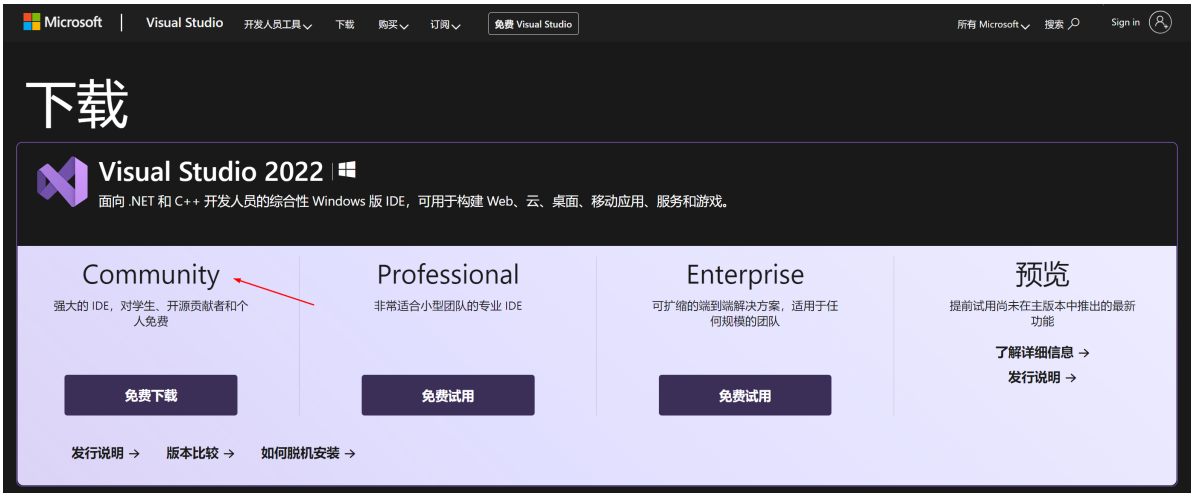
课程演示环境：Windows10, cuda 11.8, cudnn8.9

1 软件安装

1) 安装Visual Studio 2022

下载Visual Studio 社区版

下载链接：https://visualstudio.microsoft.com/zh-hans/downloads/



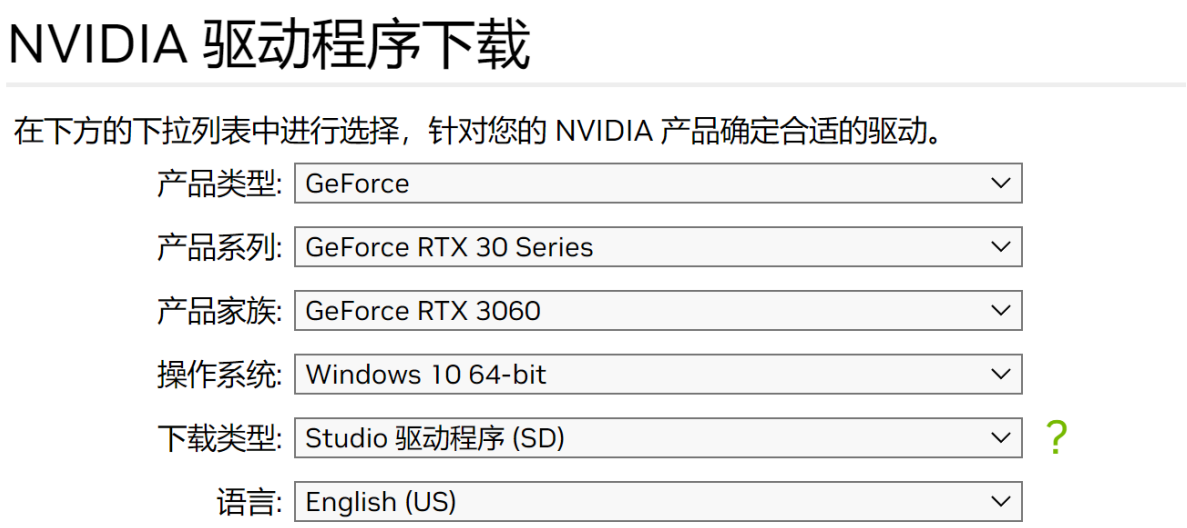
注意：安装时可勾选“Python开发”和“C++开发”

2) 下载和安装nvidia显卡驱动

首先要在设备管理器中查看你的显卡型号，比如在这里可以看到我的显卡型号为RTX 3060。

NVIDIA 驱动下载：https://www.nvidia.cn/Download/index.aspx?lang=cn

下载对应你的英伟达显卡驱动。

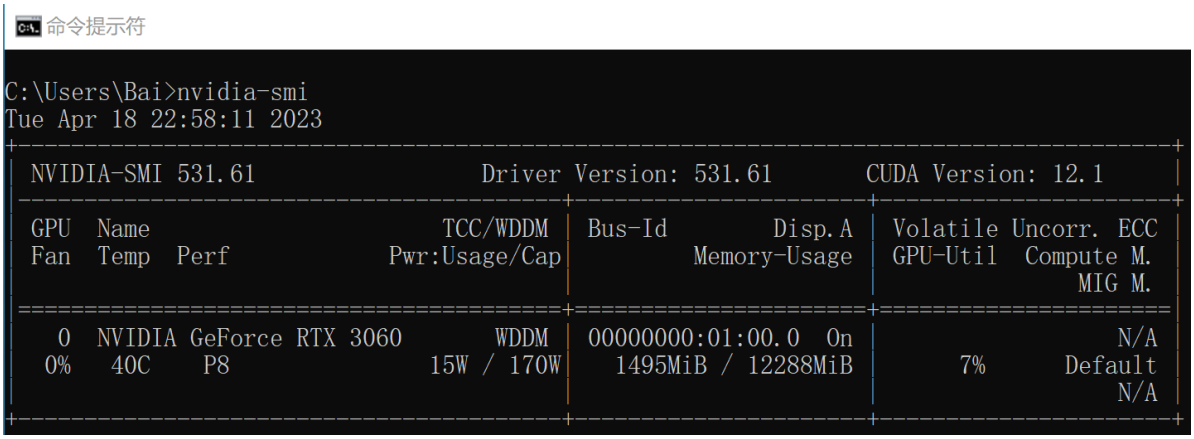


下载之后就是简单的下一步执行直到完成。

完成之后，在cmd中输入执行：

```
nvidia-smi
```

如果输出下图所示的显卡信息，说明你的驱动安装成功。

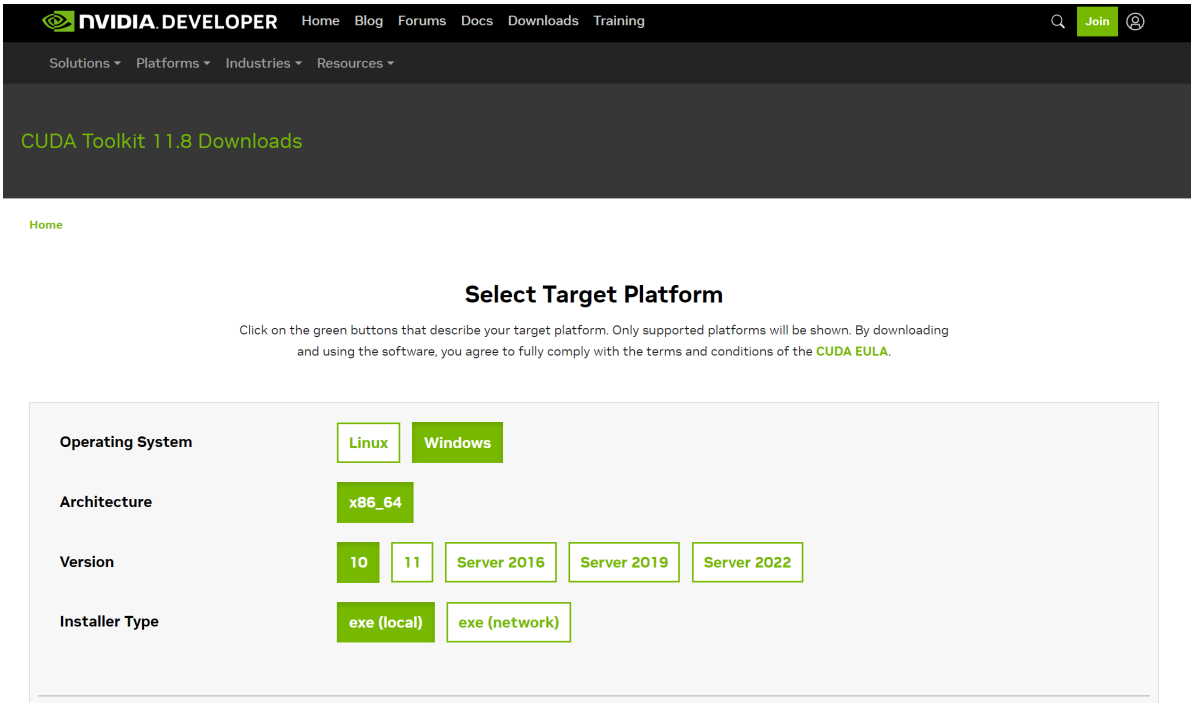


注：图中的 CUDA Version是当前Driver版本能支持的最高的CUDA版本

3) 下载CUDA

CUDA用的是11.8版本

cuda下载链接：https://developer.nvidia.com/cuda-downloads?target_os=Windows&target_arch=x86_64&target_version=10&target_type=exelocal



下载后得到文件：cuda_11.8.0_522.06_windows.exe

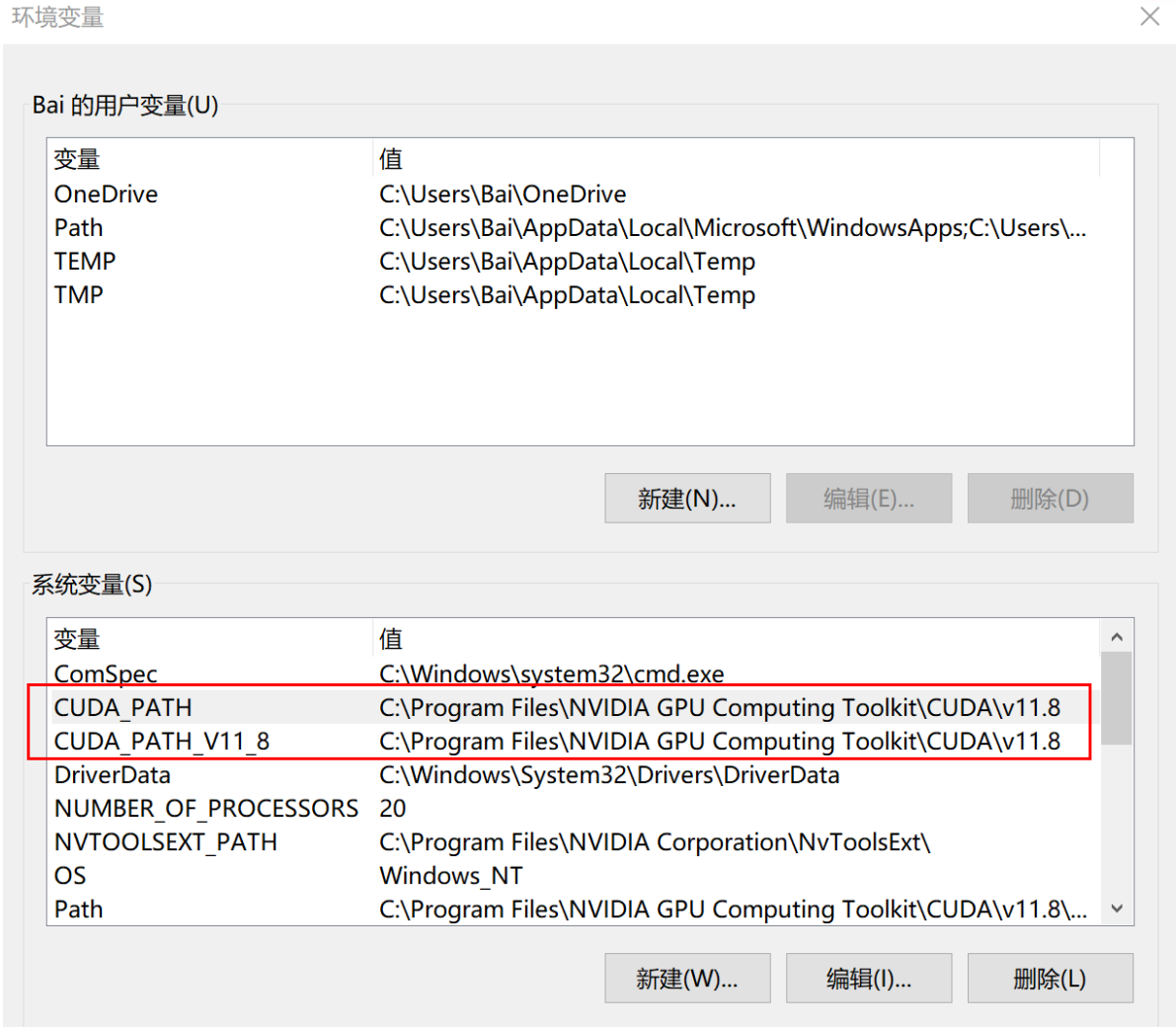
执行该文件进行安装。

4) 安装cuda

(1) 将cuda运行安装，建议默认路径

安装时可以勾选Visual Studio Integration

(2) 安装完成后设置环境变量



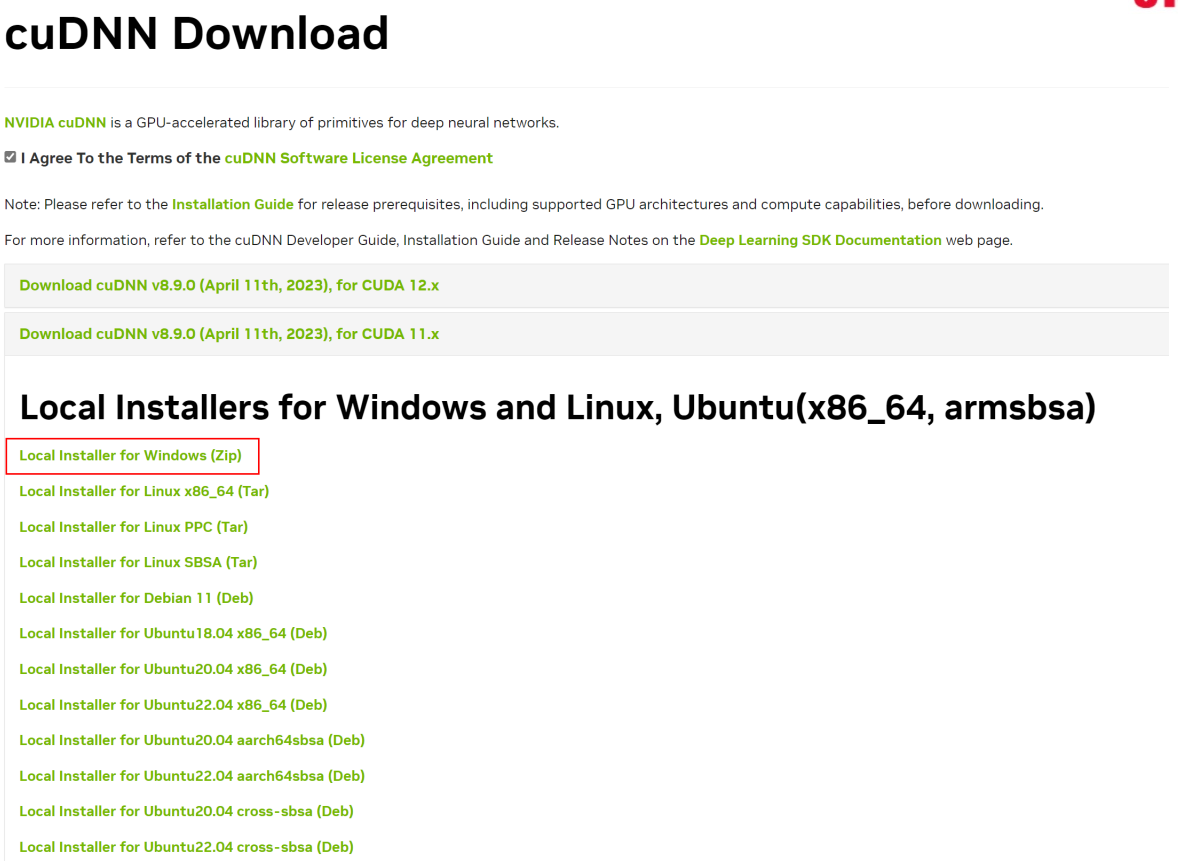
看到系统中多了CUDA_PATH和CUDA_PATH_V11_8两个环境变量。

5) 下载cuDNN

cudnn下载地址：https://developer.nvidia.com/cudnn

需要有NVIDIA账号

注意：cudnn版本要和cuda版本匹配



下载后得到文件：cudnn-windows-x86_64-8.9.0.131_cuda11-archive.zip

6) 安装cuDNN

复制cudnn文件

对于cudnn直接将其解开压缩包，然后需要**将bin,include,lib中的文件复制粘贴到cuda的文件夹下**

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8

注意：对整个文件夹bin,include,lib选中后进行复制粘贴

7）CUDA安装测试

最后测试cuda是否配置成功：

打开CMD执行：

```
nvcc -V
```

即可看到cuda的信息

命令提示符

```
Microsoft Windows [版本 10.0.19045.2006]
(c) Microsoft Corporation。保留所有权利。

C:\Users\Bai>nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2022 NVIDIA Corporation
Built on Wed_Sep_21_10:41:10_Pacific_Daylight_Time_2022
Cuda compilation tools, release 11.8, V11.8.89
Build cuda_11.8.r11.8/compiler.31833905_0
```

8） 安装Anaconda

Anaconda 是一个用于科学计算的 Python 发行版，支持 Linux, Mac, Windows, 包含了众多流行的科学计算、数据分析的 Python 包。

1) 下载安装包

Anaconda下载Windows版：https://www.anaconda.com/

2) 然后安装anaconda

9）安装pytorch

创建虚拟环境，环境名字可自己确定，这里本人使用mypytorch作为环境名:

```
conda create -n mypytorch python=3.9
```

Anaconda Prompt

```
(base) C:\Users\Bai>conda create -n mypytorch python=3.9
```

安装成功后激活mypytorch环境：

```
conda activate mypytorch
```

在所创建的mypytorch环境下安装pytorch, 执行命令：

```
conda install pytorch torchvision torchaudio pytorch-
cuda=11.8 -c pytorch -c nvidia
```

注意：11.8处应为自己电脑上的cuda版本号

离线安装：

下载网址: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/pytorch/win-64/

安装pytorch2.0版本: pytorch-2.0.0-py3.9_cuda11.8_cudnn8_0.tar.bz

```
conda install --offline pytorch-2.0.0-
py3.9_cuda11.8_cudnn8_0.tar.bz
```

2 YOLOv8项目克隆和安装

1） 克隆YOLOv8并安装

安装Git软件（https://git-scm.com/downloads），克隆项目到本地（如d:）

项目repo网址: https://github.com/ultralytics/ultralytics

在 Git CMD窗口中执行:

```
git clone https://github.com/ultralytics/ultralytics
```

在mypytorch虚拟环境下执行:

```
cd ultralytics
```

```
pip install -e .
```

3） 下载预训练权重文件

下载yolov8预训练权重文件，并放置在新建立的weights文件夹下

例如：D:\ultralytics\ultralytics\weights

百度网盘下载链接：

链接：https://pan.baidu.com/s/1cyE1ACYFN-ZDrG_bHnUqgQ
提取码：s6zn

4） 安装测试

预测图片：

```
yolo segment predict
model=D:/ultralytics/ultralytics/weights/yolov8s-seg.pt
source=D:/ultralytics/ultralytics/assets/bus.jpg
```

预测摄像头：

```
yolo segment predict
model=D:/ultralytics/ultralytics/weights/yolov8s-seg.pt
source=0 show
```

命令参数说明：https://docs.ultralytics.com/modes/predict/

3 导出ONNX模型

1）修改文件1：

D:\ultralytics\ultralytics\nn\modules\block.py中

class C2f(nn.Module)改动如下

```
def forward(self, x):
    """Forward pass through C2f layer."""
    #y = list(self.cv1(x).chunk(2, 1))
    #y.extend(m(y[-1]) for m in self.m)
    #return self.cv2(torch.cat(y, 1))

    x= self.cv1(x)
    x =[x,x[:,self.c:, ...]]
    x.extend(m(x[-1]) for m in self.m)
    x.pop(1)
    return self.cv2(torch.cat(x,1))
```

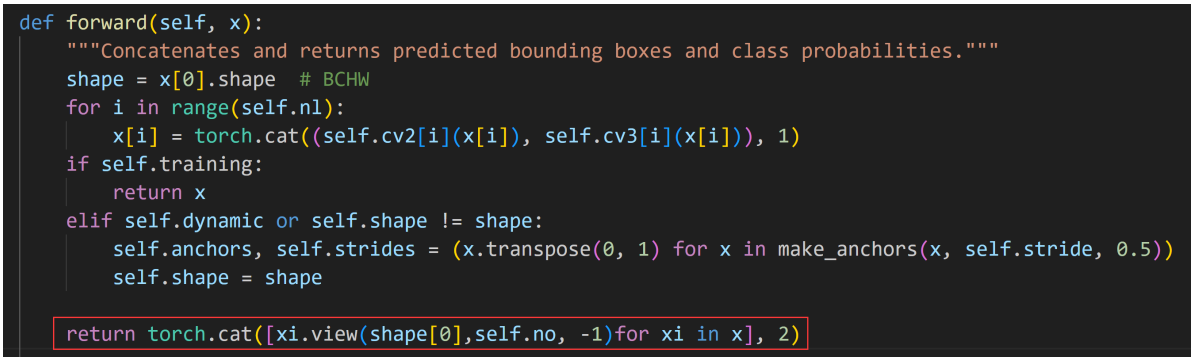
```
    def forward(self, x):
        """Forward pass through C2f layer."""
        #y = list(self.cv1(x).chunk(2, 1))
        #y.extend(m(y[-1]) for m in self.m)
        #return self.cv2(torch.cat(y, 1))

        x= self.cv1(x)
        x =[x,x[:,self.c:, ...]]
        x.extend(m(x[-1]) for m in self.m)
        x.pop(1)
        return self.cv2(torch.cat(x,1))
```

2）修改文件2：

D:\ultralytics\ultralytics\nn\modules\head.py中

class Detect(nn.Module)改动如下



```
def forward(self, x):
    """Concatenates and returns predicted bounding boxes and class probabilities."""
    shape = x[0].shape  # BCHW
    for i in range(self.nl):
        x[i] = torch.cat((self.cv2[i](x[i]), self.cv3[i](x[i])), 1)
    if self.training:
        return x
    elif self.dynamic or self.shape != shape:
        self.anchors, self.strides = (x.transpose(0, 1) for x in make_anchors(x, self.stride, 0.5))
        self.shape = shape

    return torch.cat([xi.view(shape[0],self.no, -1)for xi in x], 2)
```

```
    def forward(self, x):
        """Concatenates and returns predicted bounding
boxes and class probabilities."""
        shape = x[0].shape  # BCHW
        for i in range(self.nl):
            x[i] = torch.cat((self.cv2[i](x[i]),
self.cv3[i](x[i])), 1)
        if self.training:
            return x
        elif self.dynamic or self.shape != shape:
            self.anchors, self.strides = (x.transpose(0,
1) for x in make_anchors(x, self.stride, 0.5))
            self.shape = shape

        return torch.cat([xi.view(shape[0],self.no, -1)for
xi in x], 2)
```

注意1：旧版本的YOLOv8两个改动处都在D:\ultralytics\ultralytics\nn\modules.py中

注意2：训练YOLOv8时不需要这两个改动

3）执行命令

```
yolo export
model=D:/ultralytics/ultralytics/weights/yolov8n-seg.pt
format=onnx simplify=True opset=12
```

```
yolo export
model=D:/ultralytics/ultralytics/weights/yolov8s-seg.pt
format=onnx simplify=True opset=12
```

导出后得到文件yolov8n-seg.onnx和yolov8s-seg.onnx

自己训练出的权重文件导出：

```
yolo export model=path/to/best-seg.pt format=onnx
simplify=True opset=12
```

4 onnx转换成NCNN文件

一键生成：

https://convertmodel.com/

或采用自己安装软件按下面的步骤生成：

1）安装protobuf

下载protobuf-3.19.4安装包，并解压；

在VS2022的X64命令行下执行以下命令

注： 为解压的protobuf-3.19.4文件夹的根目录。课程中是D:/protobuf-3.19.4

```
cd <protobuf-root-dir>

mkdir build-vs2022

cd build-vs2022

cmake -G"NMake Makefiles" -DCMAKE_BUILD_TYPE=Release -
DCMAKE_INSTALL_PREFIX=%cd%/install -
Dprotobuf_BUILD_TESTS=OFF -
Dprotobuf_MSVC_STATIC_RUNTIME=OFF  -
DNCNN_BUILD_WITH_STATIC_CRT=ON  ../cmake

nmake

nmake install
```

编译后可执行检查安装是否成功

```
protoc.exe --version
```

2）克隆和安装ncnn

首先克隆ncnn

```
git clone https://github.com/Tencent/ncnn.git
```

打开VS2022的X64命令行（进入到ncnn根目录下）执行以下语句

注意：cmake -G...这条命令有三个需要换成D:/protobuf-3.19.4的根目录

```
cd <ncnn-root-dir>

mkdir -p build-vs2022

cd build-vs2022

cmake -G"NMake Makefiles" -DCMAKE_BUILD_TYPE=Release -
DCMAKE_INSTALL_PREFIX=%cd%/install -DProtobuf_INCLUDE_DIR=
<protobuf-root-dir>/build-vs2022/install/include -
DProtobuf_LIBRARIES=<protobuf-root-dir>/build-
vs2022/install/lib/libprotobuf.lib -
DProtobuf_PROTOC_EXECUTABLE=<protobuf-root-dir>/build-
vs2022/install/bin/protoc.exe -DNCNN_VULKAN=OFF -
DNCNN_BUILD_WITH_STATIC_CRT=ON ..

nmake

nmake install
```

其中

```
cmake -G"NMake Makefiles" -DCMAKE_BUILD_TYPE=Release -
DCMAKE_INSTALL_PREFIX=%cd%/install -
DProtobuf_INCLUDE_DIR=D:/protobuf-3.19.4/build-
vs2022/install/include -DProtobuf_LIBRARIES=D:/protobuf-
3.19.4/build-vs2022/install/lib/libprotobuf.lib -
DProtobuf_PROTOC_EXECUTABLE=D:/protobuf-3.19.4/build-
vs2022/install/bin/protoc.exe -DNCNN_VULKAN=OFF  -
DNCNN_BUILD_WITH_STATIC_CRT=ON ..
```

编译后

D:\ncnn\build-vs2022\tools\onnx下有onnx2ncnn.exe

3） 生成ncnn文件

拷贝yolov8n-seg.onnx和yolov8s-seg.onnx文件到D:\ncnn\build-vs2022\tools\onnx

执行命令生成ncnn相应的param和bin文件

```
onnx2ncnn.exe yolov8n-seg.onnx yolov8n-seg.param yolov8n-seg.bin
```

4）使用ncnn_optimize优化ncnn文件

产生新的param和bin文件:

D:\ncnn\build-vs2022\tools路径下执行

先拷贝yolov8n-seg.bin和yolov8n-seg.param文件，以及yolov8s-seg.bin和yolov8s-seg.param文件到此路

径下

执行命令：

```
ncnnoptimize.exe yolov8n-seg.param yolov8n-seg.bin yolov8s-seg.param yolov8s-seg.bin
```

5 安装Android Studio

官网：https://developer.android.google.cn/studio/

注意：建议使用课程网盘中的Android Studio，否则可能有版本兼容问题

安装时会提示安装SDK

同意licenses

注意：Android SDK安装路径中不要有空格

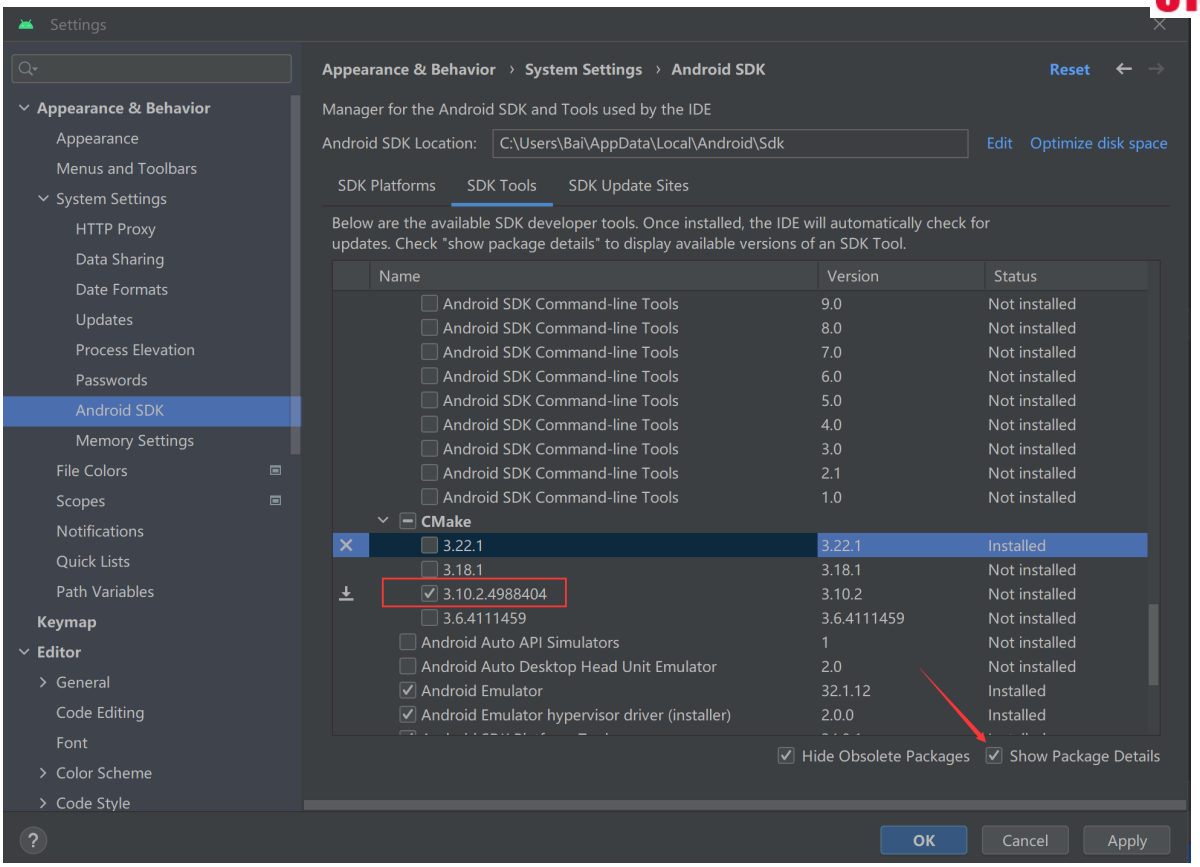
注意配置：

File->Settings->Appearance & Behavior ->System Settings->Android SDK

SDK Platforms选中面向手机的Android版本

SDK Tools选中NDK, CMake

注意：cmake的版本选择不要太高



检查**build.gradle(app)**文件

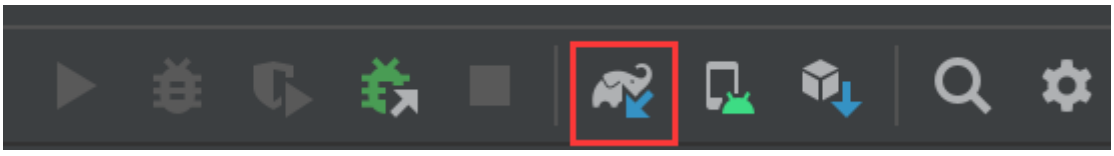
```
ndkVersion '24.0.8215888'
```

添加 `CMake` 到文件 `local.properties`

```
sdk.dir=C\:\\Users\\Bai\\AppData\\Local\\Android\\Sdk
ndk.dir=C\:\\Users\\Bai\\AppData\\Local\\Android\\Sdk\\ndk\\24.0.8215888
cmake.dir=C\:\\Users\\Bai\\AppData\\Local\\Android\\Sdk\\cmake\\3.10.2.4988404
```

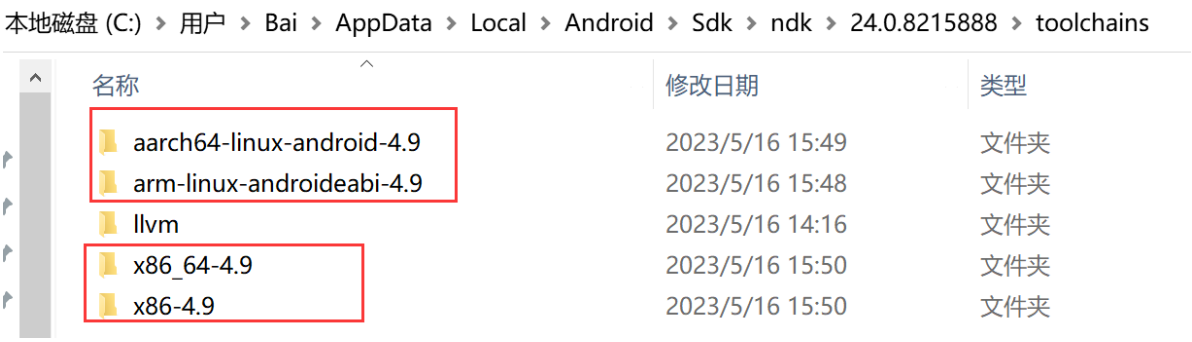
注意：上述路径和版本应为自己电脑上的真实路径

Press the button `Sync project with Gradle Files` in the upper right.



如果出现错误信息：No toolchains found in the NDK toolchains folder for ABI with prefix: arm-linux-androideabi

解决方案：同时安装低版本的ndk（如version=21.0.6113669），将低版本ndk中toolchains 文件夹下的arm-linux-androideabi等文件复制到高版本ndk的toolchains 文件夹中



6 准备Android项目文件

1)下载项目文件ncnn-android-yolov8-seg.zip，并解压。

课程中是放置在D:\ncnn-android-yolov8-seg

百度网盘下载链接：

链接：https://pan.baidu.com/s/1cyE1ACYFN-ZDrG_bHnUqgQ
提取码：s6zn

2) 放置ncnn模型文件

ncnn-android-yolov8-seg\app\src\main\assets中放入ncnn文件(即模型的param和bin文件)

修改param和bin文件名为yolov8n-seg.param，yolov8-seg.bin, yolov8s-seg.param, yolov8s-seg.bin

3) 放置ncnn和opencv的android文件

(1) 放置ncnn的安卓文件

https://github.com/Tencent/ncnn/releases

下载ncnn-YYYYMMDD-android-vulkan.zip

课程中使用ncnn-20230223-android-vulkan.zip

解压ncnn-YYYYMMDD-android-vulkan.zip后放置到**app/src/main/jni** 并修改

app/src/main/jni/CMakeLists.txt中的**ncnn_DIR**

(2) 放置opencv的安卓文件

https://github.com/nihui/opencv-mobile

下载opencv-mobile-XYZ-android.zip

课程中使用opencv-mobile-4.6.0-android.zip

解压opencv-mobile-XYZ-android.zip后放置到**app/src/main/jni**并修改

app/src/main/jni/CMakeLists.txt中的 **OpenCV_DIR**

(3) 修改CMakeLists.txt文件

D:\ncnn-android-yolov8-seg\app\src\main\jni下面

```
project(yolov8ncnn)

cmake_minimum_required(VERSION 3.10)

set(OpenCV_DIR ${CMAKE_SOURCE_DIR}/opencv-mobile-4.6.0-android/sdk/native/jni)
find_package(OpenCV REQUIRED core imgproc)

set(ncnn_DIR ${CMAKE_SOURCE_DIR}/ncnn-20230223-android-vulkan/${ANDROID_ABI}/lib/cmake/ncnn)
find_package(ncnn REQUIRED)

add_library(yolov8ncnn SHARED yolov8ncnn.cpp yolo.cpp ndkcamera.cpp)

target_link_libraries(yolov8ncnn ncnn ${OpenCV_LIBS} camera2ndk mediandk)
```

7 手机连接电脑并编译软件

1) 安装投屏软件(optional)

https://www.apowersoft.com.cn/phone-mirror-pinzhuan?apptype=aps-pin

手机和电脑都要安装

2）手机连接电脑

以小米11手机为例

(1)设置开发者模式

设置->我的设备->全部参数

点击MIUI版本三次

(2)设置USB调试和安装

设置->更多设置->开发者选项

打开USB调试；USB安装

(3)手机通过USB数据线（或WiFi）连接电脑

3）编译和调试

打开已经存在的项目，选择build.gradle

4）导出签名apk

Build->Generate Signed APK

8 自己数据集训练模型的部署

1）使用YOLOv8模型训练自己的数据集，可参考本人的课程《YOLOv8实例分割实战：训练自己的数据集》

2）ncnn模型文件的替换

3）修改yolo.cpp文件中的class_names

yolo.cpp文件generate_proposals函数中

```
const int num_class = 80;
```

改为自己数据集中的类别数

参考：

https://github.com/Tencent/ncnn

https://github.com/nihui/opencv-mobile

https://github.com/FeiGeChuanShu/ncnn-android-yolov8

声明

本课程的数据集、程序文件以及课件的演示文稿、视频由讲师白勇拥有知识产权的权利。只限于学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造 。