一、下载代码

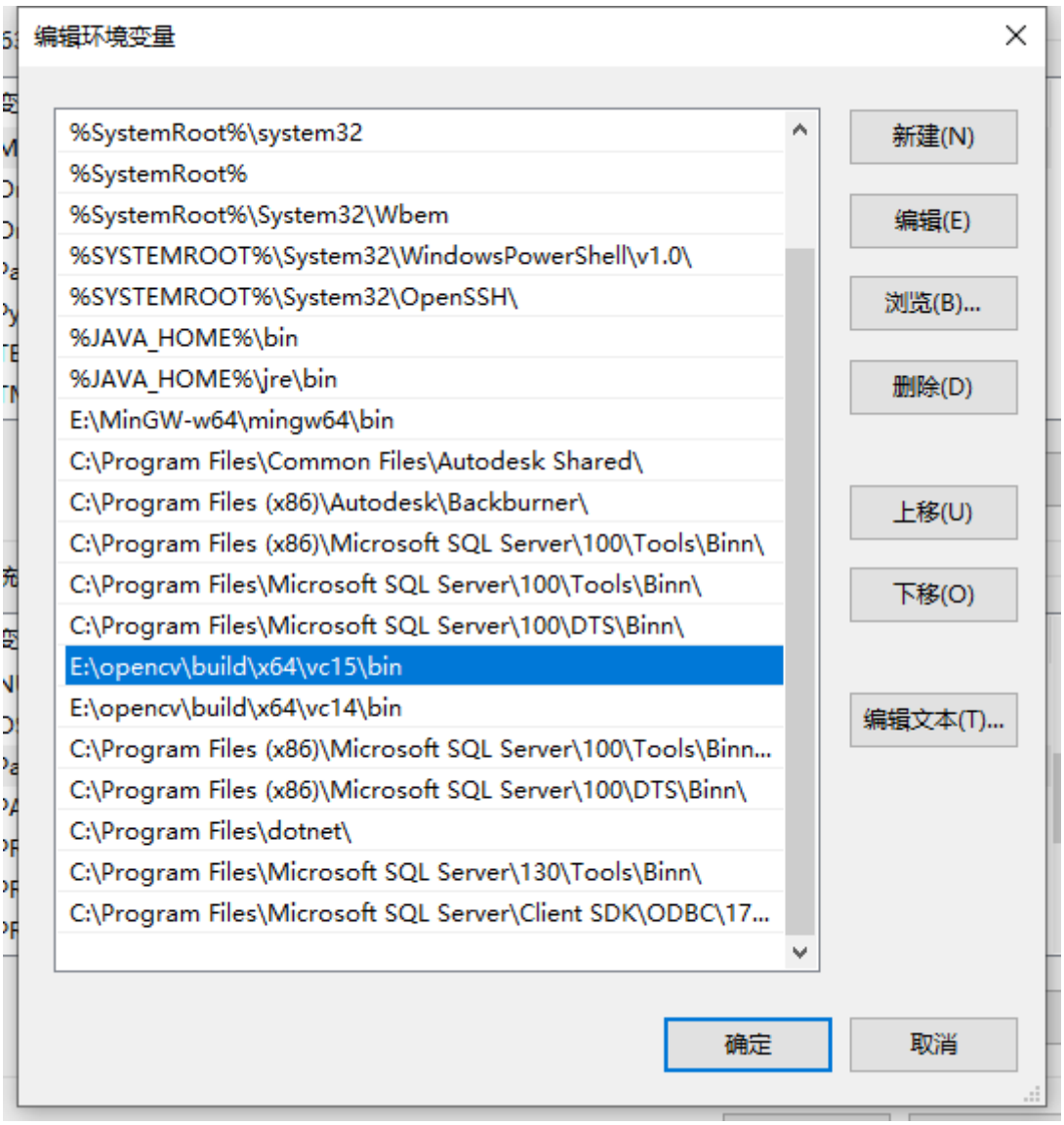
链接：https://pan.baidu.com/s/1nvgtKIRVPOEa0pJazW8GHQ提取码：qhkp

二、环境配置

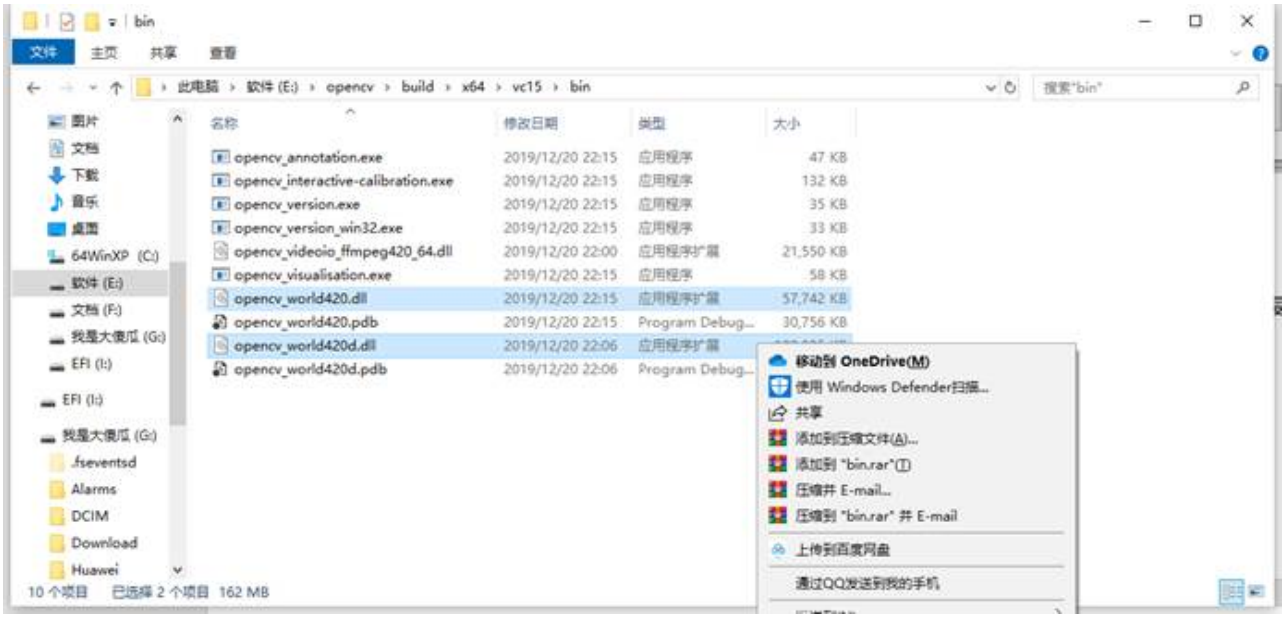
1、opencv环境配置

> opencv版本：opencv340
> 可以使用下面的教程（也可以参考：VS+OpenCV+OpenVINO2022详细配置 - 知乎 (zhihu.com)）

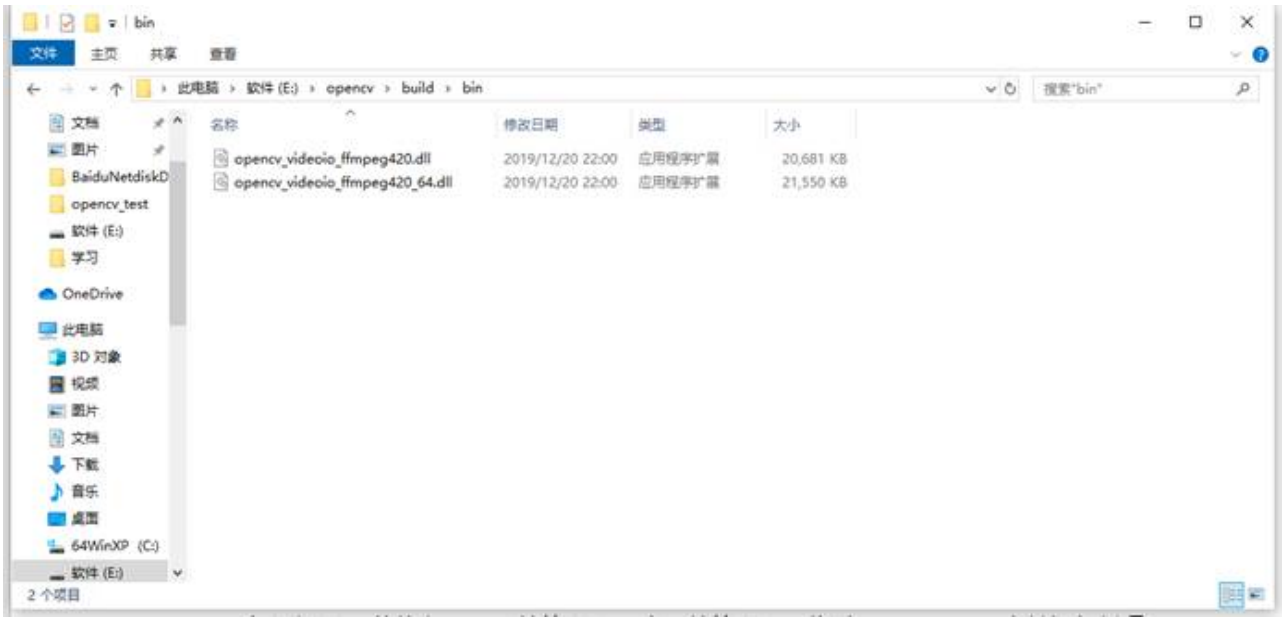
（1）我的电脑右键->属性->高级系统设置->环境变量->系统变量->Path->添加D:\OpenCV\build\x64\vc15\bin（注意是按照你自己的OpenCV的路径，这里给出的是我的路径，不可直接复制使用）添加进的是opencv安装目录的build中的bin文件夹路径）



（2）将路径：D:\OpenCV\build\x64\vc15\bin\中的opencv_world420.dll、opencv_world420d.dll 复制到C:\Windows\SysWOW64目录下



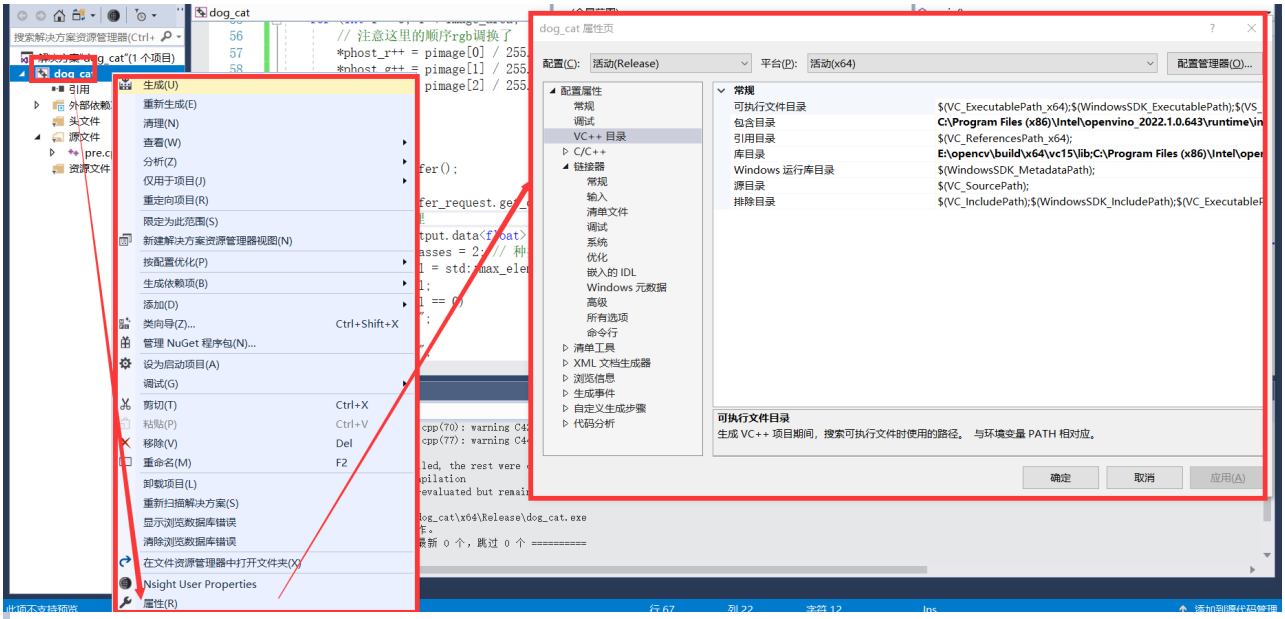
（3）将路径：D:\OpenCV\build\bin\中的opencv_videoio_ffmpeg420_64.dll 复制到C:\Windows\System32目录下



如果找不到Windows可以点查看把隐藏项目勾上

2、VS新建项目，配置项目

> openvino环境配置可参考：Openvino安装命令



配置VC++ 目录的”包含目录“和”库目录“

”包含目录“:

C:\Program Files (x86)\Intel\openvino_2022.1.0.643\runtime\include

C:\Program Files (x86)\Intel\openvino_2022.1.0.643\runtime\include\openvino

C:\Program Files (x86)\Intel\openvino_2022.1.0.643\runtime\include\ngraph

C:\Program Files (x86)\Intel\openvino_2022.1.0.643\runtime\include\ie

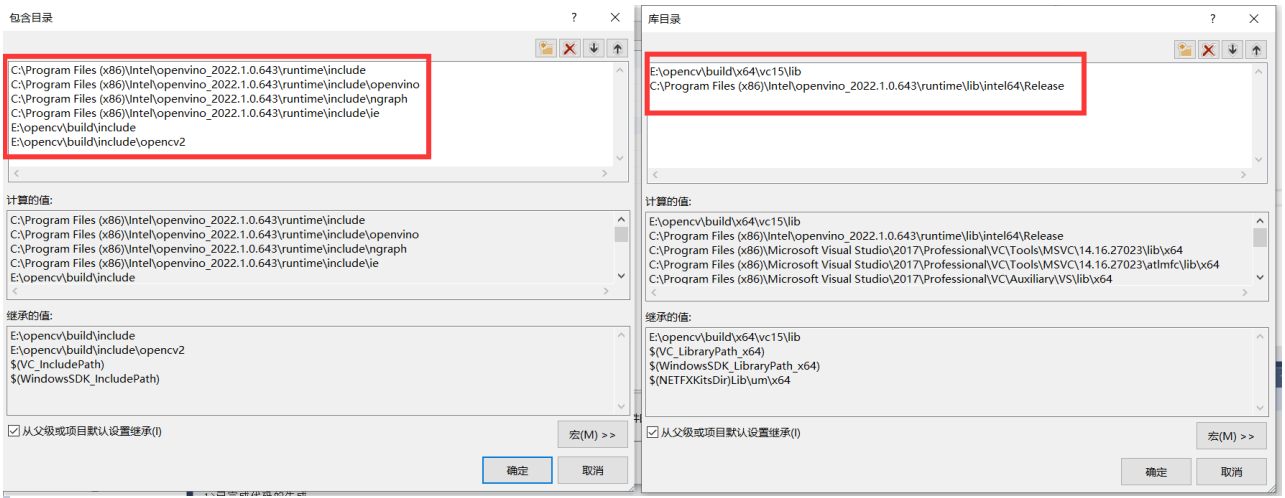
E:\opencv\build\include

E:\opencv\build\include\opencv2

”库目录“:

E:\opencv\build\x64\vc15\lib

C:\Program Files (x86)\Intel\openvino_2022.1.0.643\runtime\lib\intel64\Release

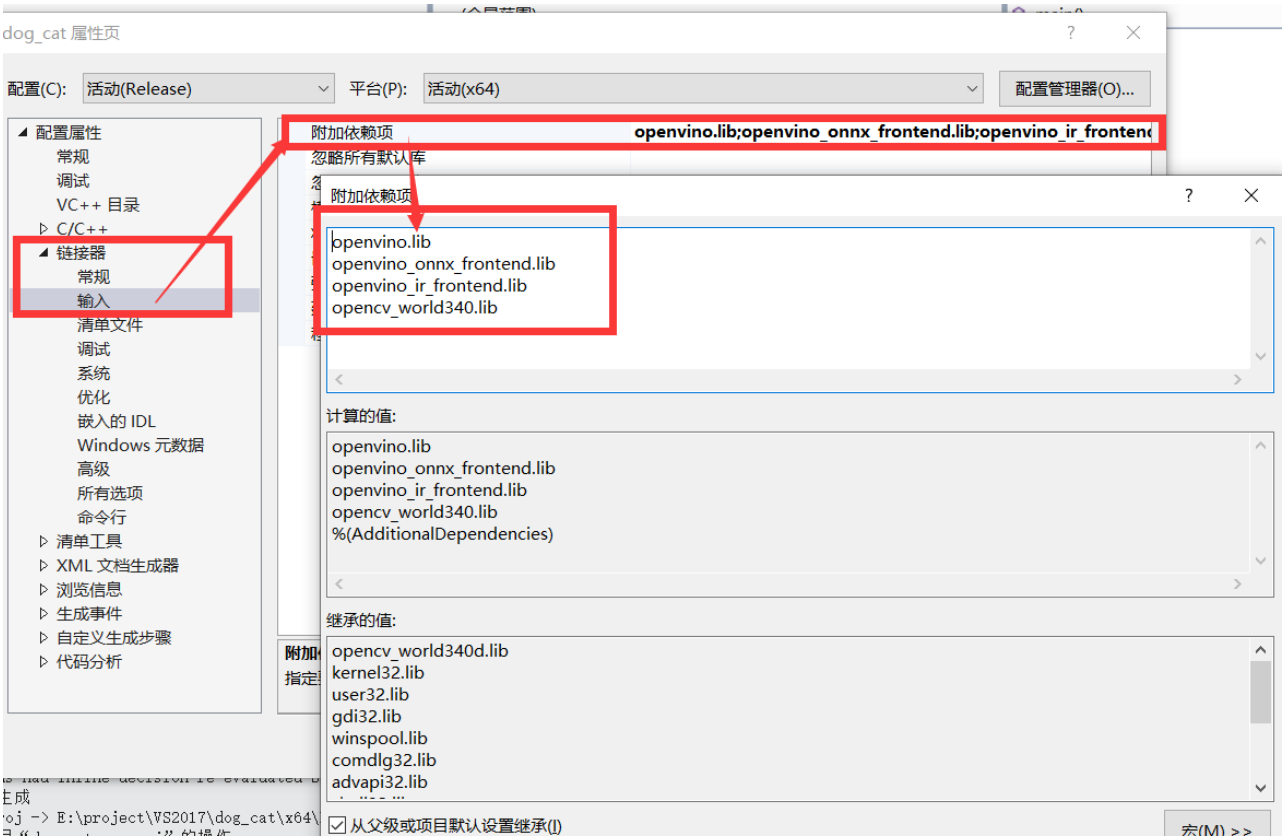


配置”链接器“的输入:

openvino.lib

openvino_onnx_frontend.lib

> openvino_ir_frontend.lib
> opencv_world340.lib



三、运行代码

```cpp
#include <openvino/openvino.hpp>
#include <iostream>
#include <vector>
#include <opencv2/opencv.hpp>


int main() {
        std::string img_path = "E:/project/VS2017/dog_cat/img/dog.jpg"; // 预测图片
        std::string onnx_path = "E:/project/VS2017/dog_cat/model/Cat_dog.onnx"; // 预测模型
        size_t input_batch_size = 1; // 输入图片的batch_size
        size_t num_channels = 3;                // 输入通道
        size_t h = 224;                                 // 输入图片的高
        size_t w = 224;                                 // 输入图片的宽
        clock_t startTime, endTime;             // 推理时间记录变量

 // 0、创建IE插件，查询支持硬件设备
```

```
        ov::Core core;
//获取当前支持的所有的AI硬件推理设备
        std::vector<std::string> devices = core.get_available_devices();
for (int i = 0; i < devices.size(); i++) {
                std::cout << devices[i] << std::endl;
}
// 1、加载检测模型
// 模型加载并编译
        ov::CompiledModel compiled_model = core.compile_model(onnx_path, "AUTO");
// 创建用于推断已编译模型的推理请求对象  创建的请求分配了输入和输出张量
        ov::InferRequest infer_request = compiled_model.create_infer_request();

// 2、请求网络输入
auto input_tensor = infer_request.get_input_tensor(0);

// 3、指定shape的大小
        input_tensor.set_shape({ input_batch_size, num_channels, w, h });
// 4、获取输入的地址，并传递给指针input_data_host
float* input_data_host = input_tensor.data<float>();

// 对应于pytorch的代码部分
// 推理开始时间
        startTime = clock();
// opencv读取图片
        cv::Mat src = cv::imread(img_path);
int image_height = src.rows;
int image_width = src.cols;
// 修改图片大小
        cv::Mat image;
        cv::resize(src, image, cv::Size(w, h));
int image_area = image.cols * image.rows;
unsigned char* pimage = image.data;
float* phost_b = input_data_host + image_area * 0;    // input_data_host和phost_*进行地址关联
float* phost_g = input_data_host + image_area * 1;
float* phost_r = input_data_host + image_area * 2;
// BGR->RGB
float mean[] = { 0.406, 0.456, 0.485 };
float std[] = { 0.225, 0.224, 0.229 };
for (int i = 0; i < image_area; ++i, pimage += 3) {
```

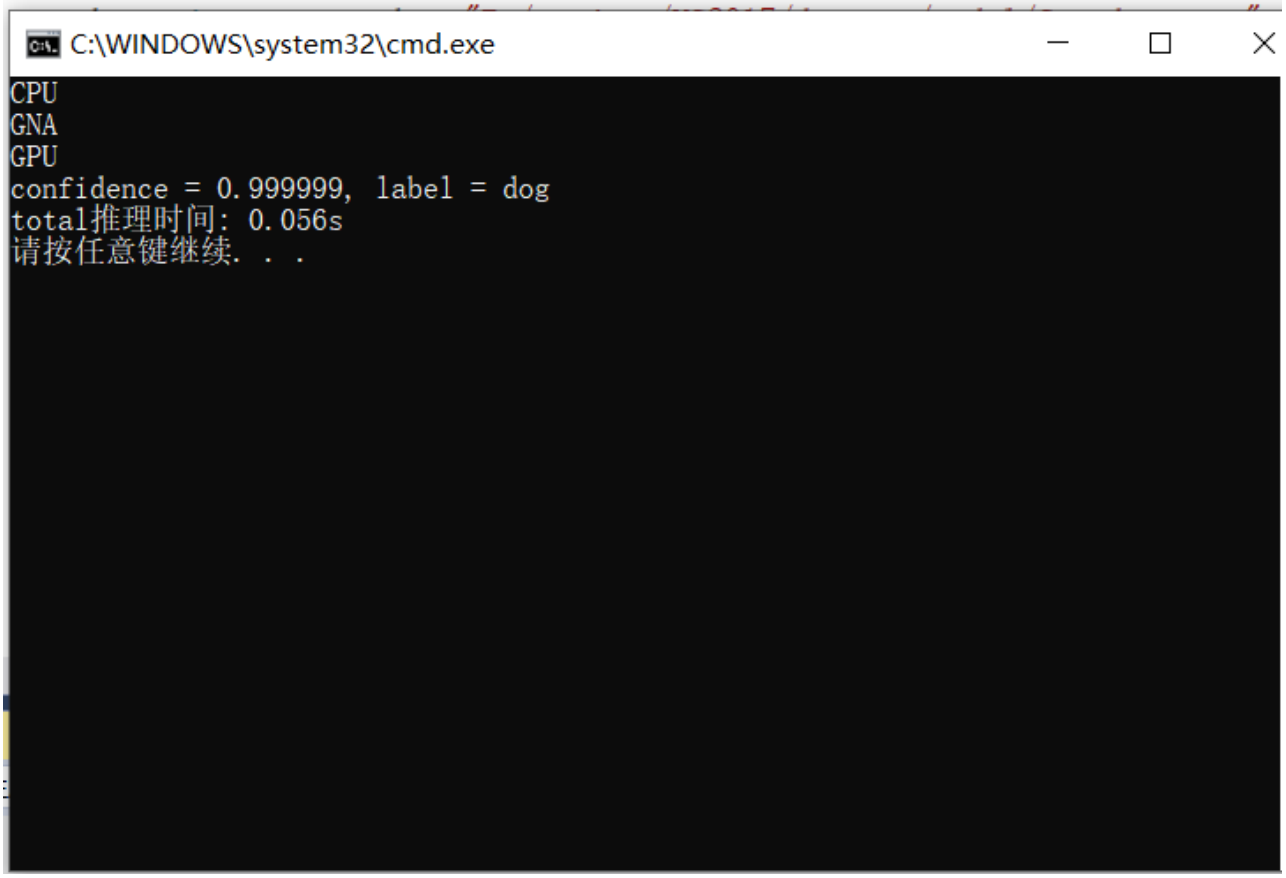
```
                // 注意这里的顺序rgb调换了
                *phost_r++ = pimage[0] / 255.;  // 将图片中的像素点进行减去均值除方差，并赋值给input
                *phost_g++ = pimage[1] / 255.;
                *phost_b++ = pimage[2] / 255.;
 }


 // 5、执行预测
        infer_request.infer();
 // 6、推理结果
 auto output = infer_request.get_output_tensor(0);
 // 对输出结果处理
 float* prob = output.data<float>();
 const int num_classes = 2; // 种类
 int predict_label = std::max_element(prob, prob + num_classes) - prob;  // 确定预测类别的下标
        std::string label;
 if (predict_label == 0)
                label = "cat";
 else
                label = "dog";
 float confidence = prob[predict_label];    // 获得预测值的置信度
 printf("confidence = %f, label = %s\n", confidence, label);
        endTime = clock();//计时结束
        std::cout << "total推理时间: " << (double)(endTime - startTime) / CLOCKS_PER_SEC << "s" << std::endl;

 return 0;
}
```

四、运行结果

```
std::string img_path = "E:/project/VS2017/dog_cat/img/dog.jpg"; // 预测
```




```
std::string img_path = "E:/project/VS2017/dog_cat/img/cat.jpg"; // 预测图
```