windows+libtorch configuration

2020-03-05 | | 本文总阅读量 8623 次

Intro

This blog aims to teach how to deploy a pytorch model in Windows platform. The deployed model in this article has only inference ability for torch::jit does not support back propagation of some layers or operations. Of course, reasoning only is enough for many projects. The tools used for deployment include visual studio，opencv，libtorch。

Environment

win10 platform

cuda10.2+cudnn7.6.5

Gtx 1080Ti

visual studio 2017 community version

opencv 4.5.0

libtorch 1.7

Actually，except that libtorch version must be higher than or equal to pytorch version(for possible api problem) and visual studio must be no less than 2015. Other dependencies have no rigid version requirment. Enven graphics card is not a must if cpu speed is a satisfactory.

visual studio

visual studio download link here, just download it and install c++ parts.

opencv

Official opencv website provides what you want。Download the executable file and unzip it to the desired dir.

libtorch

Down load the libtorch 1.7x release.

PyTorch Build	Stable (1.7.1)	Preview (Nightly)		
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python	C++ / Java		
CUDA	9.2	10.1	10.2	11.0	None

Run this Command:

```
Windows binaries do not support Java. Support is only available for Linux and MacOS. Download here for C++ (Release version):
https://download.pytorch.org/libtorch/cu102/libtorch-win-shared-with-deps-1.7.1.zip

Download here for C++ (Debug version):
https://download.pytorch.org/libtorch/cu102/libtorch-win-shared-with-deps-debug-1.7.1.zip
```

Then unzip the file to destination just like the following setting

Example

generate .pt file

I chose ResNet34 as an example to show the deployment. Prepare a picture to test if succeeded. The picture in the blog is here:

The generate torchscript file:

```
from torchvision.models import resnet34
import torch.nn.functional as F
import torch.nn as nn
import torch
import cv2

#read a picture, convert to [1,3,224,224] float tensor
image = cv2.imread("flower.jpg")
```

```
image = cv2.resize(image,(224,224))
input_tensor = torch.tensor(image).permute(2,0,1).unsqueeze(0).float()/225.0

#define resnet34 and load ImageNet pretrained
model = resnet34(pretrained=True)
model.eval()
#check outputs
output = model(input_tensor)
output = F.softmax(output,1)
print("predicted class:{}th, prob:{}".format(torch.argmax(output),output.max()))

#generate .pt
model=model.to(torch.device("cpu"))
model.eval()
var=torch.ones((1,3,224,224))
traced_script_module = torch.jit.trace(model, var)
traced_script_module.save("resnet34.pt")
```

the outputs should be a .pt file and:

```
predicted class: 723th, prob: 0.5916505455970764
```

Visual Studio config

new a Visual Studio project。

Open Visual Studio 2017. Click file->new->project and new a empty c++ project.

Config env

Choose Release mode and x64 platform：

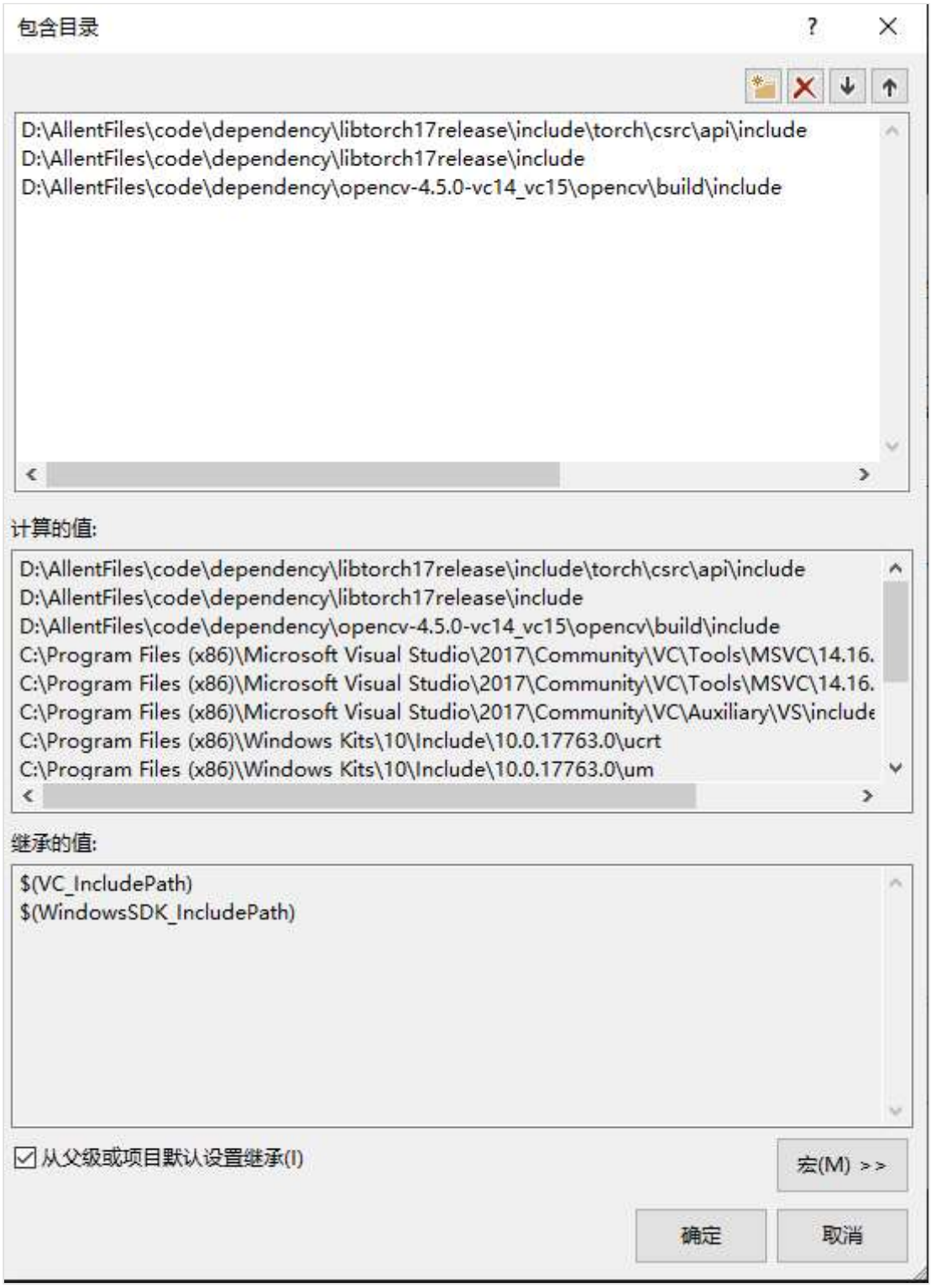
Property

include

Right clike the project and go to property configuration page. Choose VC++ include and add include and lib path.

```
your path to libtorch\include\torch\csrc\api\include
your path to libtorch\include
your path to opencv\build\include
```

Mine is here

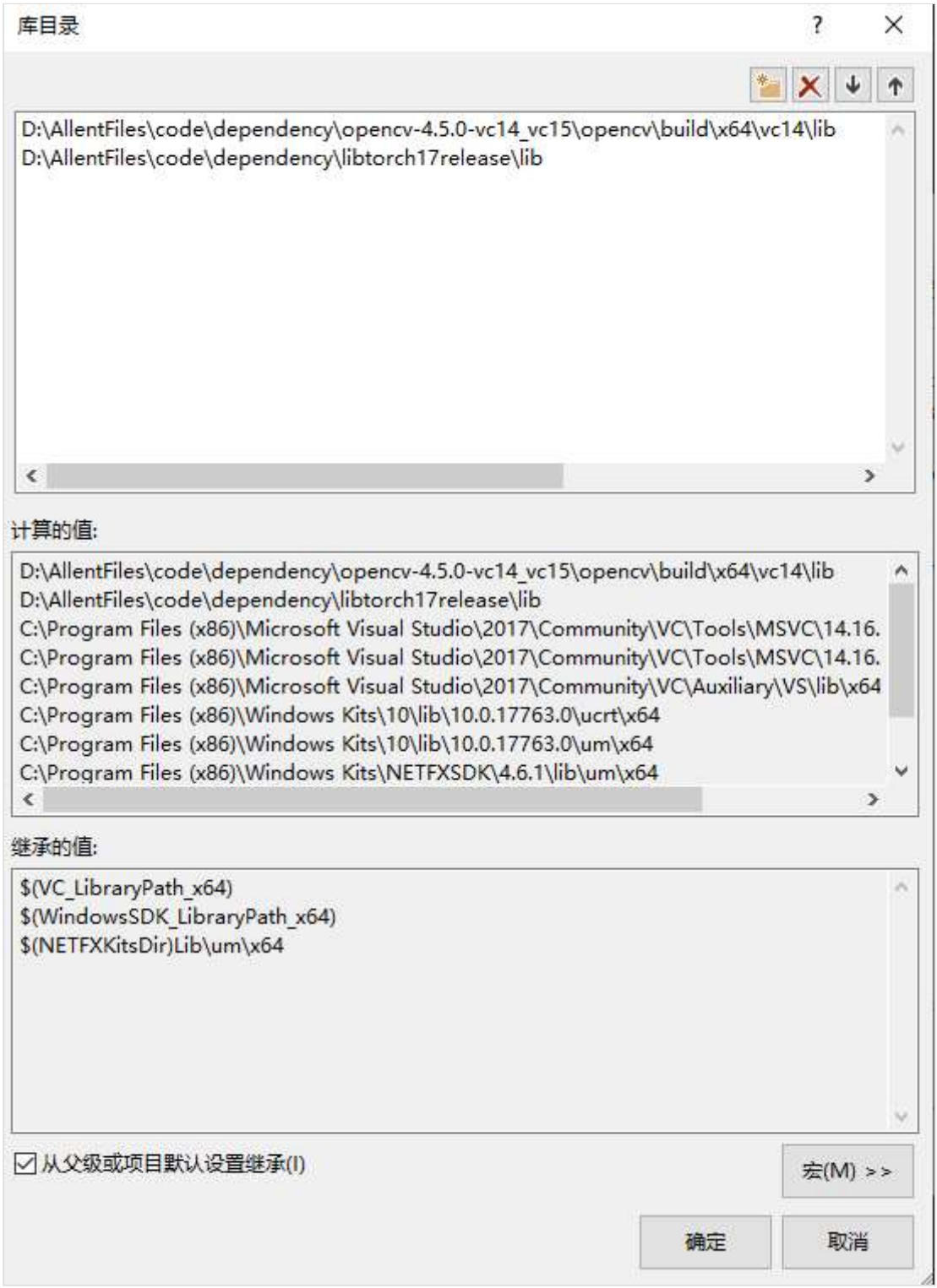


lib

Lib path should be:

```
your path to libtorch\lib
your path to opencv\build\x64\vc14\lib
```

The relationships of VC-VS, opencv-vc are here。VS2017 target vc15, opencv’ \build\x64 includes vc14 and vc15. This blog choose vc14. Lib path configuration like follows:



linker

The last is the linker, clik the linker->input->add dependency and add all the .lib file name in libtorch. Besides, don’ t forget .lib file name under opencv:

```
opencv_world450.lib
asmjit.lib
c10.lib
c10d.lib
c10_cuda.lib
caffe2_detectron_ops_gpu.lib
caffe2_module_test_dynamic.lib
caffe2_nvrtc.lib
clog.lib
cpuinfo.lib
dnnl.lib
fbgemm.lib
gloo.lib
gloo_cuda.lib
libprotobuf-lite.lib
libprotobuf.lib
libprotoc.lib
mkldnn.lib
torch.lib
torch_cpu.lib
torch_cuda.lib
```

dll

Copy all the dll files needed to the excutation path. If you don’ t know which dll is needed, just run the code. Then the windows will remind you....

cpp code

```
#include<opencv2/opencv.hpp>
#include <torch/torch.h>
#include <torch/script.h>

int main()
{
        //cuda
        auto device = torch::Device(torch::kCUDA,0);
        //read pic
        auto image = cv::imread("your path to\\flower.jpg");
        //resize
        cv::resize(image, image, cv::Size(224, 224));
        //convert to tensor
        auto input_tensor = torch::from_blob(image.data, { image.rows, image.cols, 3 }, t
        //load model
        auto model = torch::jit::load("your path to\\resnet34.pt");
        model.to(device);
```

```
        model.eval();
        //forward the tensor
        auto output = model.forward({input_tensor.to(device)}).toTensor();
        output = torch::softmax(output, 1);
        std::cout << "predicted class: " << torch::argmax(output) << ", prob is: " << out
        return 0;
}
```

编译执行，代码的输出结果为：

```
predicted class: 723
[ CUDALongType{} ], prob is: 0.591652
[ CUDAFloatType{} ]
```

You will find that c++ result is a little bit different from python’ s which is pretty trival.

Some errors

error1：cannot use GPU

If you have GPU but cannot use it in C++, try add the following content in the linker：

```
/INCLUDE:?warp_size@cuda@at@@YAHXZ
```

reference.

error2：error “std”：undefined symbol

clik property page->property config->c/c++->language-> set conformance mode no。

error3：miss dll

If build succeeded but xxxx.dll missing when running the code, just paste the dll aside .exe file.

Donate

libtorch

以AlexNet为例分析pytorch

QT Creator + Opencv + Libtorch +CUDA

English