

win10+VS2017+Opencv3.4.1+libtorch開發環境建置(2)

原創 sdf20201029 萌萌噠程序猴 2021-04-20 22:03

在前文我們講過VS2017+Opencv3.4.1的配置，本文就讓我們來講libtorch的環境配置。在配置完成之後，我們寫一個簡單的VS2017工程，透過呼叫libtorch的函數來驗證我們的配置是否成功。

win10+VS2017+Opencv3.4.1+libtorch開發環境建置(1)

首先，講一下我的liborch版本以及相關依賴函式庫的版本的下載與安裝，這裡必須要版本互相對應，否則很可能會出錯：

1. liborch

從官網下載對應的libtorch庫：

<https://pytorch.org/get-started/locally/>

如下圖所示，讀者需要注意根據自己的配置環境選擇對應的版本：

The screenshot shows the PyTorch 'Get Started Locally' page with several options selected and highlighted with red boxes:

- PyTorch Build:** Stable (1.8.1)
- Your OS:** Windows
- Package:** LibTorch
- Language:** C++ / Java
- Compute Platform:** CUDA 11.1

Below the selection grid, the text states: "Windows stable binaries do not support Java, but nightly binaries do. Support is only available for Linux and MacOS. Download here for C++ (Release version):"

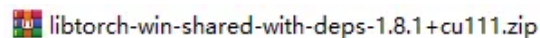
Two download links are provided, both highlighted with red boxes:

- Release version: <https://download.pytorch.org/libtorch/cu111/libtorch-win-shared-with-deps-1.8.1%2Bcu111.zip>
- Debug version: <https://download.pytorch.org/libtorch/cu111/libtorch-win-shared-with-deps-debug-1.8.1%2Bcu111.zip>

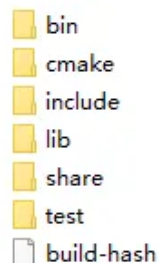
Annotations on the left side of the image:

- A red arrow points from the text "Release版本" to the first link.
- A red arrow points from the text "Debug版本" to the second link.

我選擇的是Release版本，下載下來的libtorch庫如下圖：



把以上壓縮檔案解壓縮到某一目錄，解壓縮出來之後包含以下檔案：

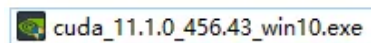


2. CUDA

去NVIDIA官網下載CUDA：

<https://developer.nvidia.com/cuda-downloads>

我使用的版本是CUDA 11.1，讀者如果使用與我不同版本的libtorch，則需要另外選擇對應的CUDA版本。



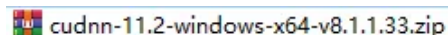
要注意的是，CUDA的安裝必須在VS2017之後，否則VS2017將無法辨識CUDA程序，且無法新建CUDA工程。不過如果讀者是先安裝了CUDA再安裝的VS2017，也不必卸載並重裝VS2017，直接重裝CUDA即可。而且，安裝CUDA的過程中只需點選next即可，最好不要修改預設路徑，否則很可能會出錯。

3. CUDNN

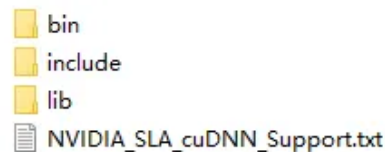
同樣去NVIDIA官網下載CUDNN：

<https://developer.nvidia.com/rdp/cudnn-download>

我使用的是CUDNN 11.2 x64版本，讀者如果使用與我不同版本的CUDA，則需要另外選擇對應的CUDNN版本。



解壓縮以上壓縮包，得到以下檔案：



這時，我們需要找到CUDA的安裝路徑，我電腦上的路徑是：

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1

如果讀者安裝CUDA時沒有修改預設路徑，基本上可以認為路徑跟我的是一樣的（可能有差別的是最後的v11.1版本不一樣）。

接著，需要將解壓縮CUDNN壓縮包得到的一些檔案拷貝到CUDA的安裝目錄：

(1)將以上bin資料夾下面的檔案拷貝到以上CUDA安裝路徑的bin資料夾下；

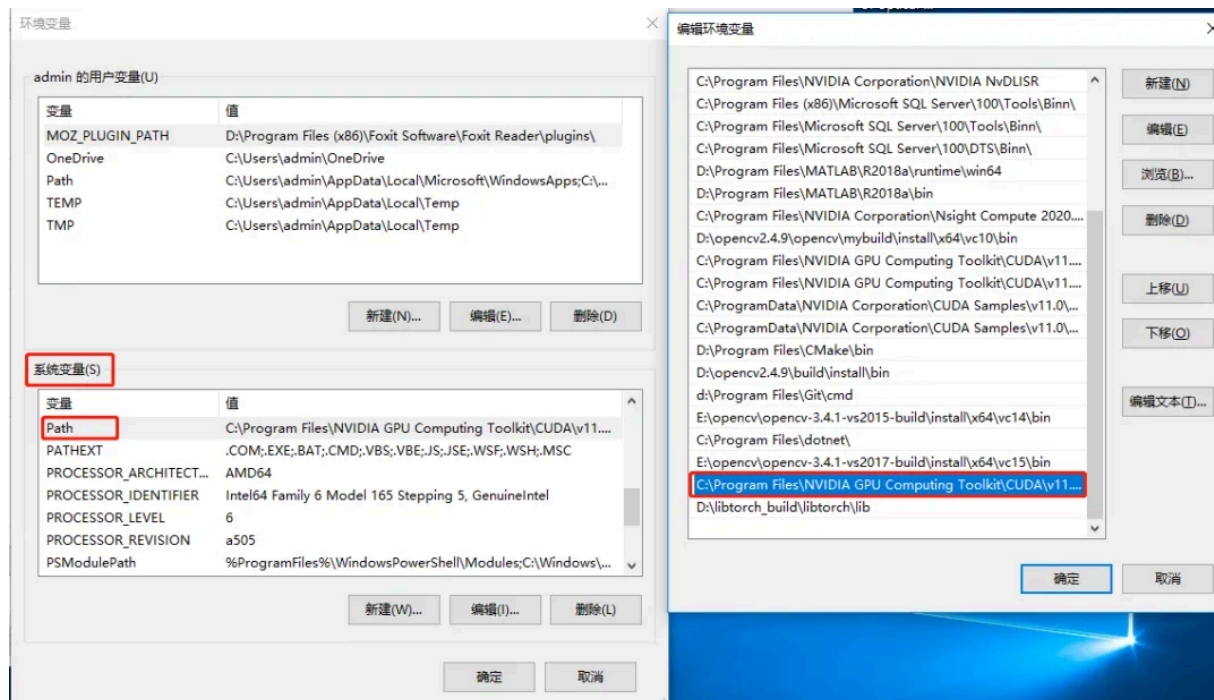
(2)將以上include資料夾下方的檔案拷貝到以上CUDA安裝路徑的include資料夾下；

(3)將以上lib/x64資料夾下方的檔案拷貝到以上CUDA安裝路徑的lib/x64資料夾下；

(4)把CUDA安裝路徑的lib/x64目錄加入到系統環境變數path中，我電腦上的完整路徑為：

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64

那麼將以上路徑加入系統環境變數path：



經過上述安裝與配置，接下來我們可以開始VS2017工程的配置了。首先新建一個空的VS2017控制台程序，設定為Release/x64版本，並新增一個main檔案（前文配置Opencv的時候已經詳細說明怎麼新建和設定工程，此處不再重複）：





1. 設定頭檔包含目錄

專案-->屬性-->VC++目錄-->包含目錄-->選擇頭檔目錄：

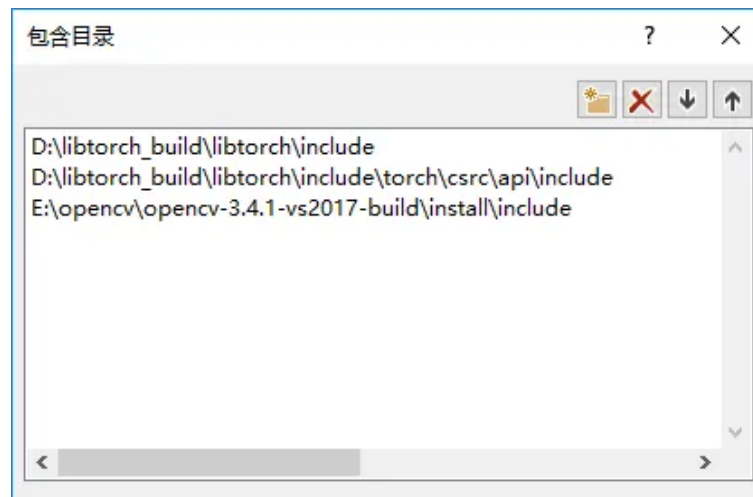
(1)設定Opencv3.4.1頭檔目錄如下，opencv-3.4.1-vs2017-build也就是我們前面編譯Opencv3.4.1時所設定的編譯目錄：

E:\opencv\opencv-3.4.1-vs2017-build\install\include

(2)設定libtorch頭檔目錄如下，這裡的目錄也就是我們把libtorch庫下載下來之後的解壓縮目錄：

D:\libtorch_build\libtorch\include

D:\libtorch_build\libtorch\include\torch\csrc\api\include



2. 設定庫目錄

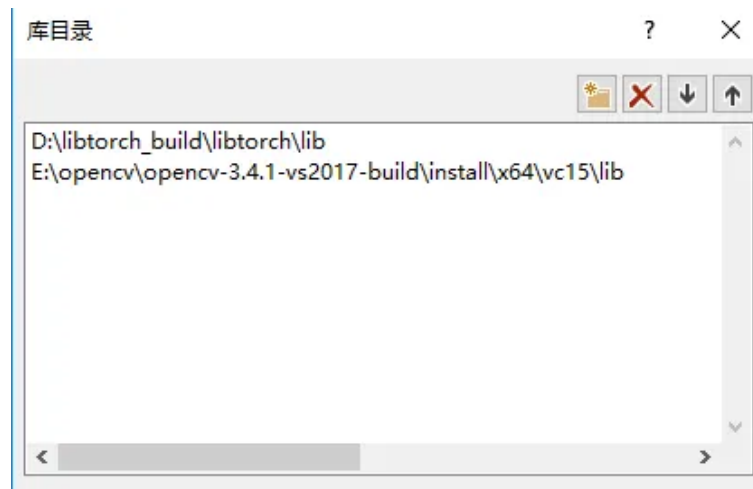
專案-->屬性-->VC++目錄-->庫目錄-->選擇庫檔案目錄：

(1)設定Opencv3.4.1函式庫檔案目錄如下，opencv-3.4.1-vs2017-build也就是我們前面編譯Opencv3.4.1時所設定的編譯目錄：

E:\opencv\opencv-3.4.1-vs2017-build\install\x64\vc15\lib

(2)設定libtorch函式庫檔案目錄如下，這裡的目錄也就是我們把libtorch函式庫下載下來之後的解壓縮目錄：

D:\libtorch_build\libtorch\lib



3. 新增依賴庫

項目-->屬性-->連結器-->輸入-->附加依賴項-->新增對應的lib檔。

這裡貼出我新增的檔案如下（如果有絕對路徑，讀者需要注意填寫自己電腦上面的對應檔案的實際路徑，不過也大同小異啦）：

```
1  c10.lib
2  C:\Program Files\NVIDIA Corporation\NvToolsExt\lib\x64\nvToolsExt64_1.lib
3  C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64\cudart_static.lib
4  caffe2_nvrtc.lib
5  c10_cuda.lib
6  torch.lib
7  torch_cuda.lib
8  torch_cuda_cu.lib
9  -INCLUDE:?searchsorted_cuda@native@@@YA?AVTensor@2@AEBV32@0_N1@Z
10 torch_cuda_cpp.lib
11 torch_cpu.lib
12 C:\Program Files\NVIDIA Corporation\NvToolsExt\lib\x64\nvToolsExt64_1.lib
```

```
13 C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64\cufft.lib
14 C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64\curand.lib
15 C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64\cublas.lib
16 C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64\cudnn.lib
17 -INCLUDE:?warp_size@cuda@@@YAHXZ
18 opencv_img_hash341.lib
19 opencv_world341.lib
```

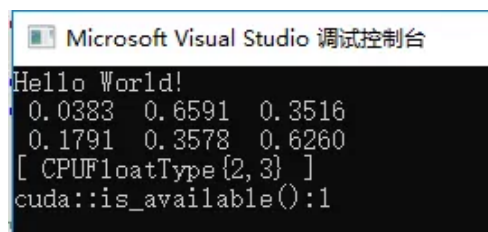
4. 在main檔案中加入程式碼

```
1 #include <iostream>
2 #include <memory>
3 #include <torch/torch.h>
4 #include <torch/script.h>
5 #include <opencv2/opencv.hpp>
6 #include <opencv2/core/core.hpp>
7 #include <opencv2/xfeatures2d.hpp>
8
9 using namespace cv;
10 using namespace std;
11
12 int main()
13 {
14     std::cout << "Hello World!\n";
15     torch::Tensor tensor = torch::rand({ 2,3 });
16     std::cout << tensor << std::endl;
17     torch::save(tensor, "tensor.pt");
18 }
```



```
19     std::cout << "cuda::is_available():" << torch::cuda::is_available() << std::endl;
20
21     return EXIT_SUCCESS;
22 }
```

5. 編譯並執行上述程式碼，得到結果如下：



```
Microsoft Visual Studio 调试控制台
Hello World!
0.0383 0.6591 0.3516
0.1791 0.3578 0.6260
[ CPUFloatType{2,3} ]
cuda::is_available():1
```

如果運行沒有出錯，至此則VS2017+Opencv3.4.1+libtorch環境配置成功，接下來可以開始奇妙的libtorch深度學習之旅啦~

不過要注意的是，如果列印出來的是“cuda::is_available(): 1”，說明可以運行CUDA加速版本的libtorch，如果列印出來的是“cuda::is_available(): 0”，說明不能執行CUDA加速版的libtorch，原因可能是沒有安裝好CUDA、電腦顯示卡本身不支援CUDA，或是沒有設定好VS2017工程等，需要逐一檢查。

如果讀者有安裝過Caffe或Tensorflow等深度學習框架，肯深有體會，它們有好多依賴庫，安裝起來非常麻煩，而且還需要各種採坑，讓人抓狂。一對比下來，發現配置libtorch環境簡單得多。



C++ 70 Opencv 50 人工智慧 27 深度學習 26 機器學習 33

人工智慧 目錄

上一 篇

win10+VS2017+Opencv3.4.1+libtorch開發環境建置(1)

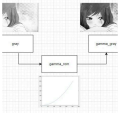
下一 篇

基於libtorch的LeNet-5卷積神經網路實現

閱讀原文

喜歡此內容的人還喜歡

數位影像處理之gamma矯正
FPGA開源工作室



混凝土模板荷載與壓力計算
忒修斯破船



此n非彼n
阿璇教學研究室

