**导出c++：**

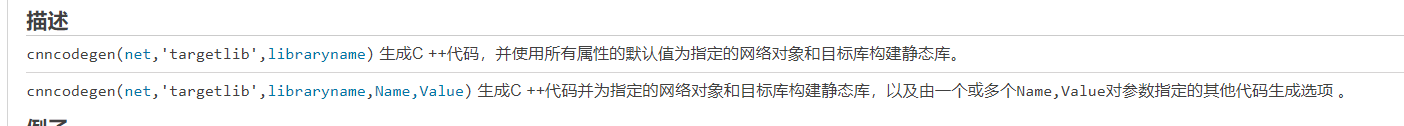
函数：

cnncodegen(‘net’,'targetlib',‘libraryname’)

cnncodegen(net,'targetlib',libraryname,Name,Value)

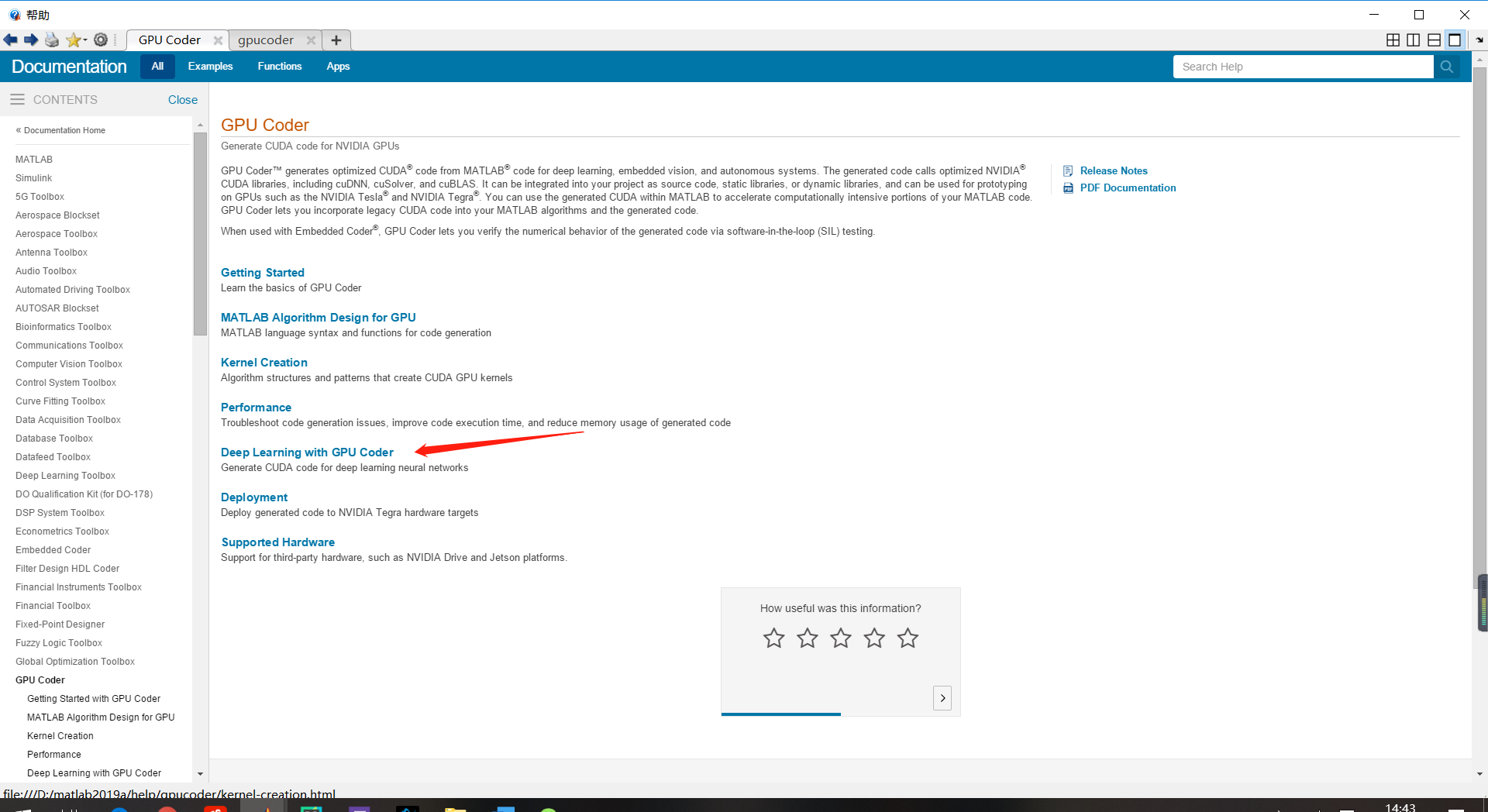
libraryname决定生成什么代码，‘cudnn’生成的是cuda代码，

‘arm-compute’生成的是用在arm上的c++

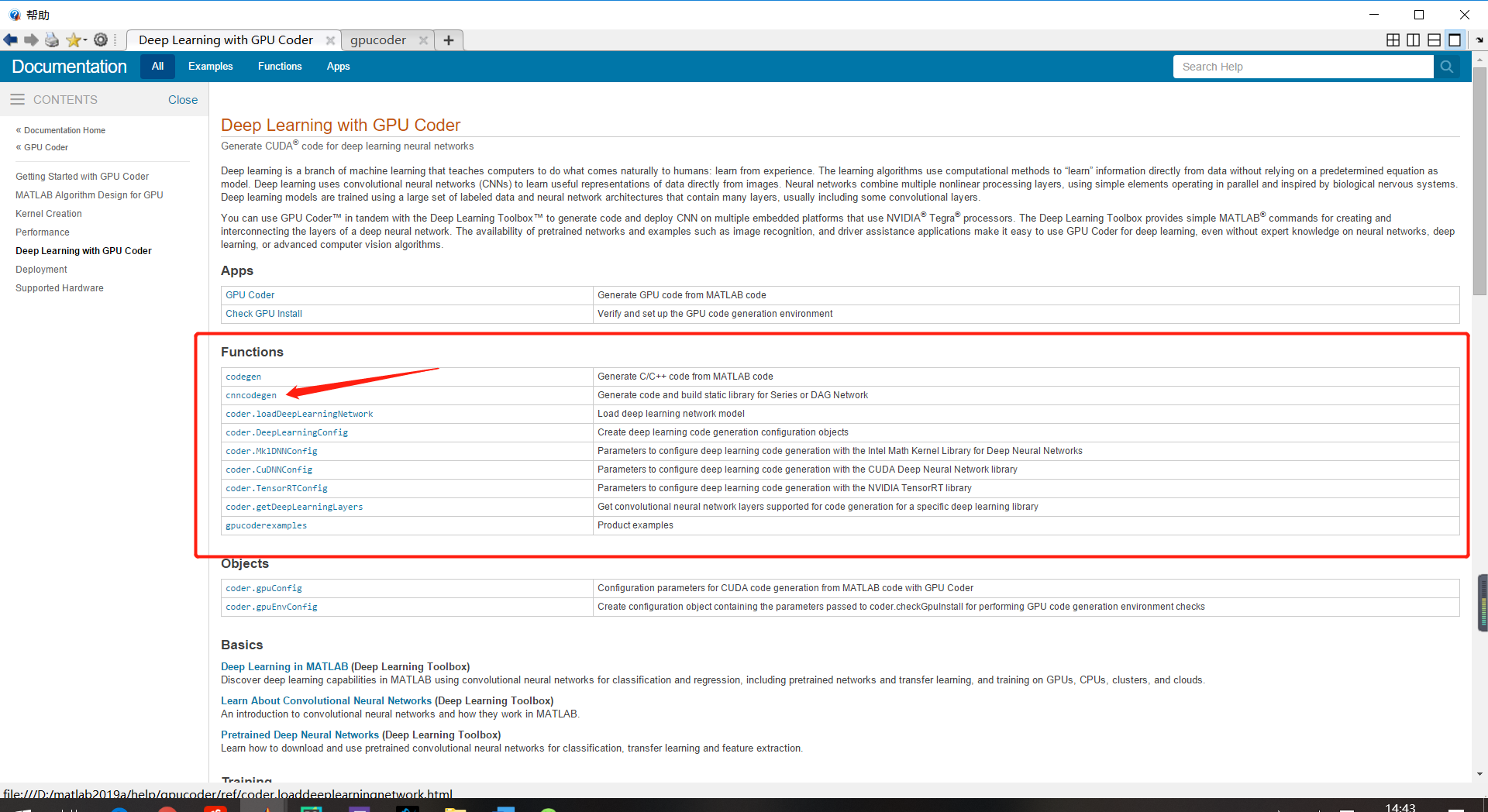


**如何找到这些变量应该填什么：**

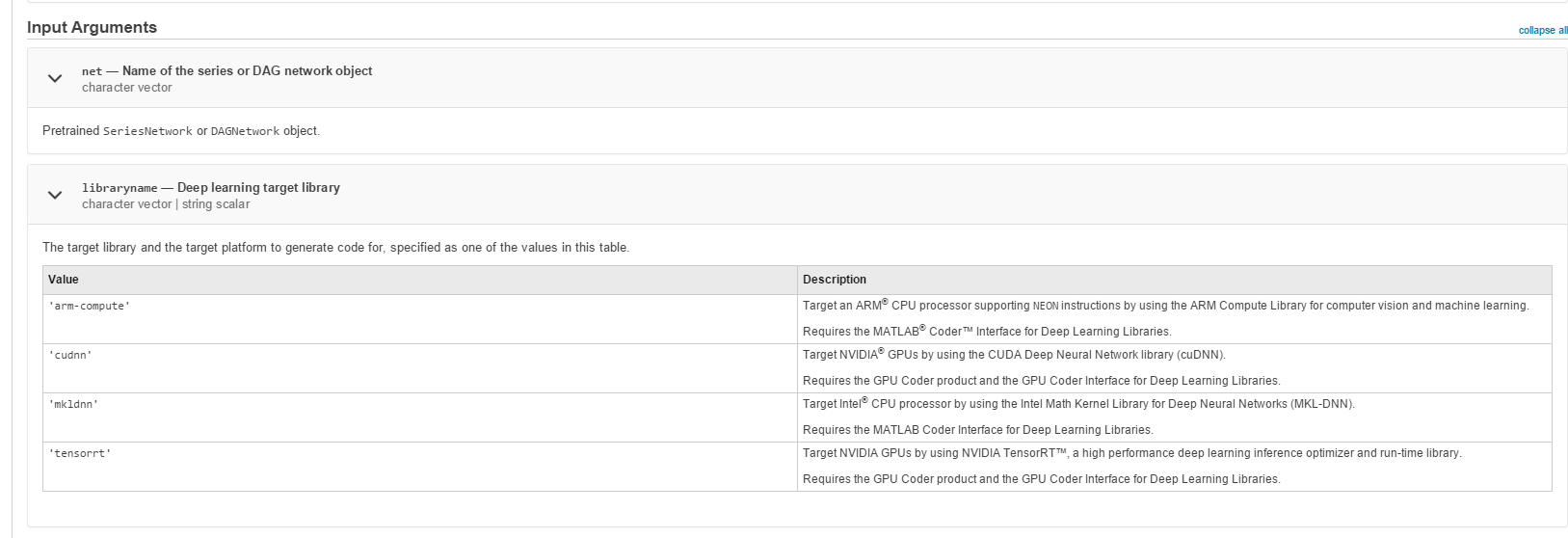
文档里面的gpu coder



这些是其他的导出代码，我们主要用红箭头指向的（cnncodegen）：

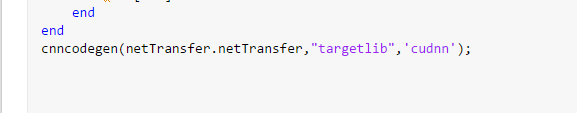
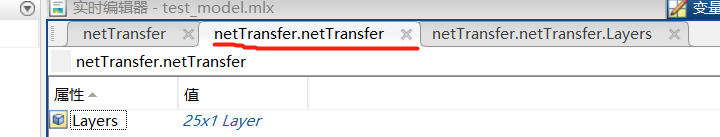


点进去，里面的Input Arguments（输入参数）

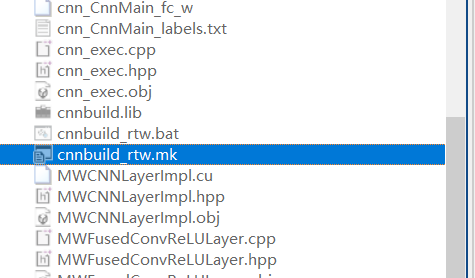


有cnncodegen()的参数设置说明。

如果是导出的模型，第一个参数应该到Layer的前一层

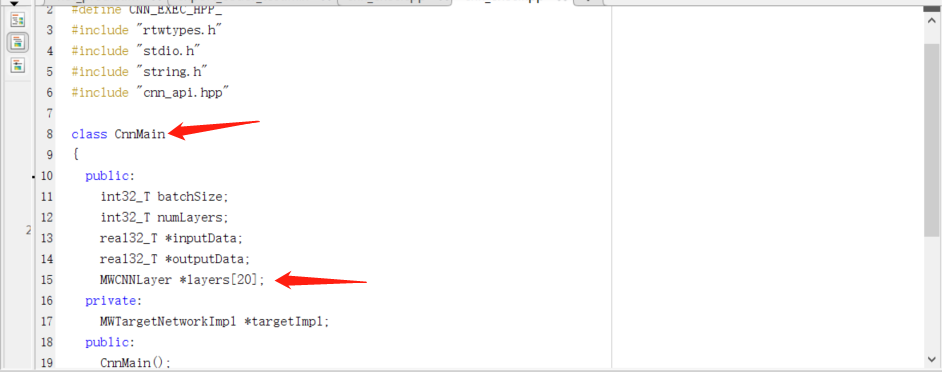
生成的新文件夹里面，有各种头文件和源文件，Cnnbuild\_rtw.mk是生成静态链接库的文件



这个txt文件时标签文件



Cnn\_exec.hpp头文件里面的CnnMain类，有神经层数量和



**用这几行代码生成mex文件**

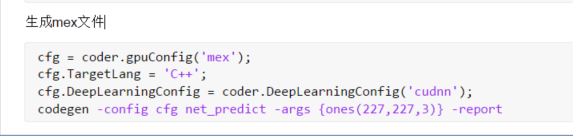
cfg = coder.gpuConfig('mex');

cfg.TargetLang = 'C++';

cfg.DeepLearningConfig = coder.DeepLearningConfig('cudnn');

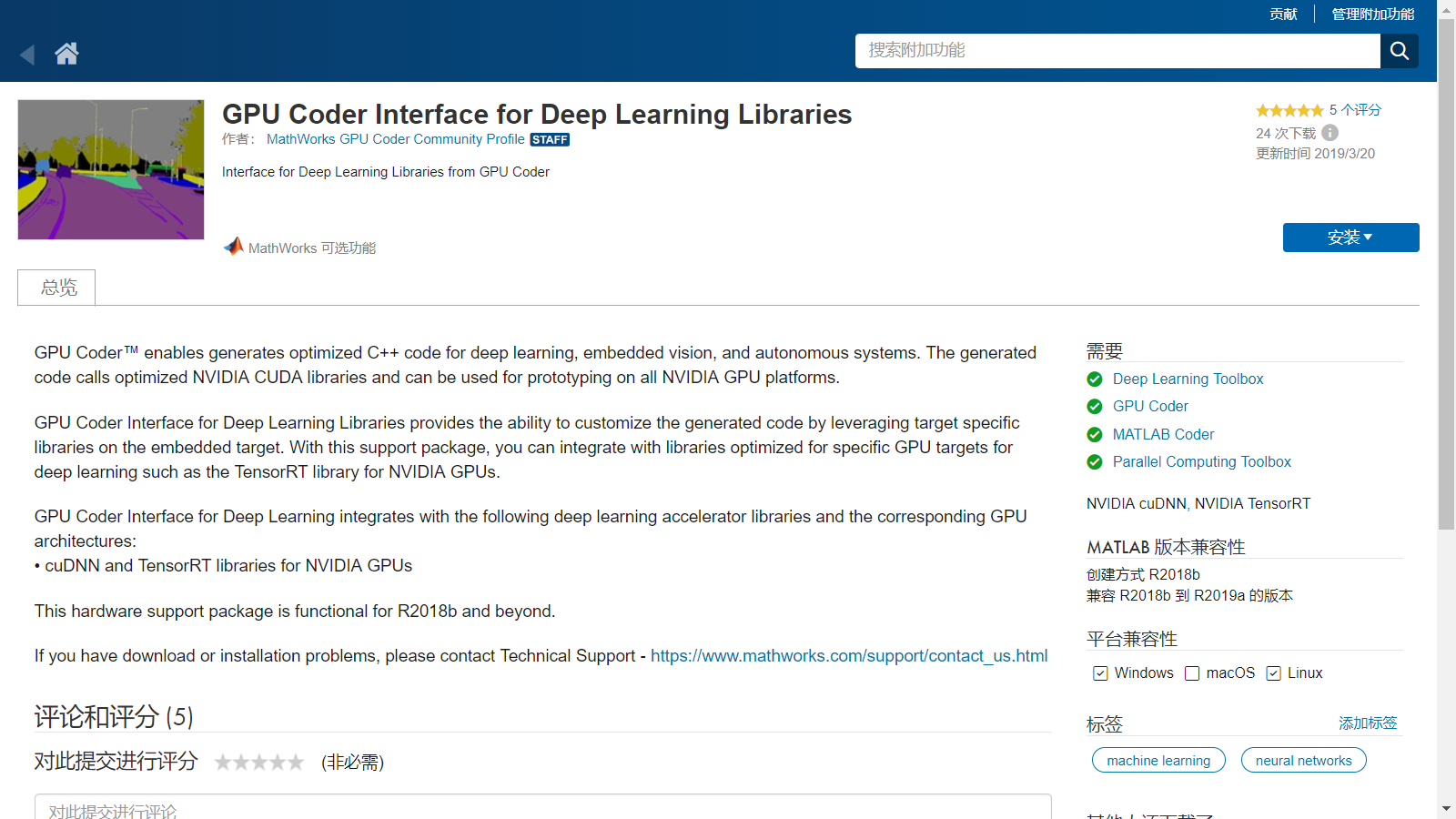
codegen -config cfg net\_predict -args {ones(227,227,3)} -report

生成的是codegen下还会有一个mex文件夹，生成的mex文件



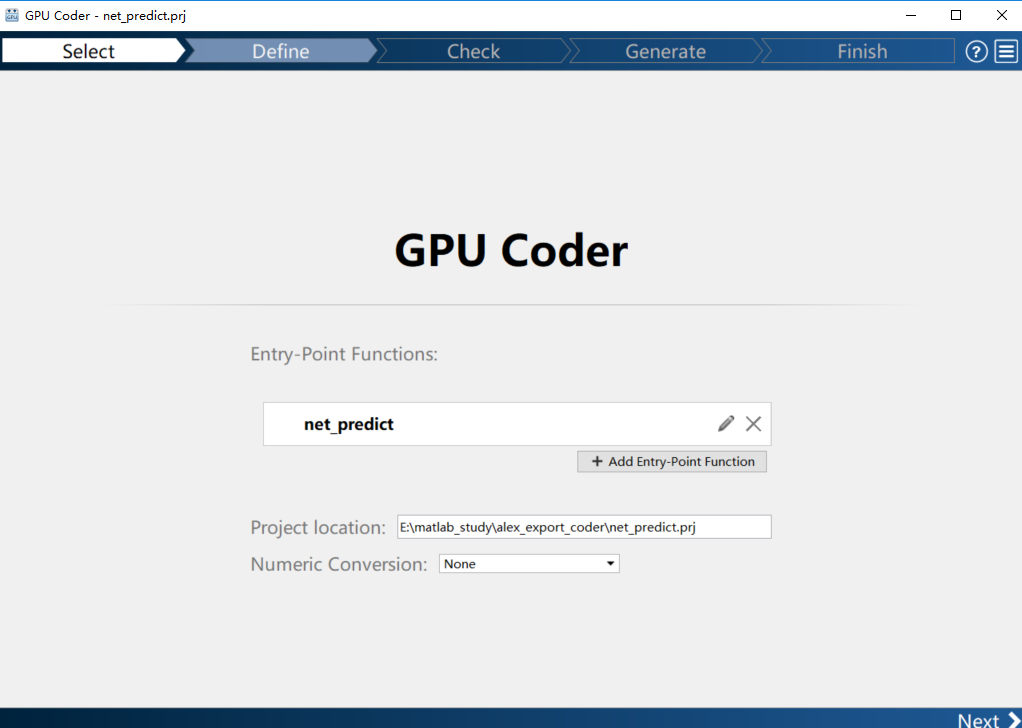
**导出cuda**

需要安装APP

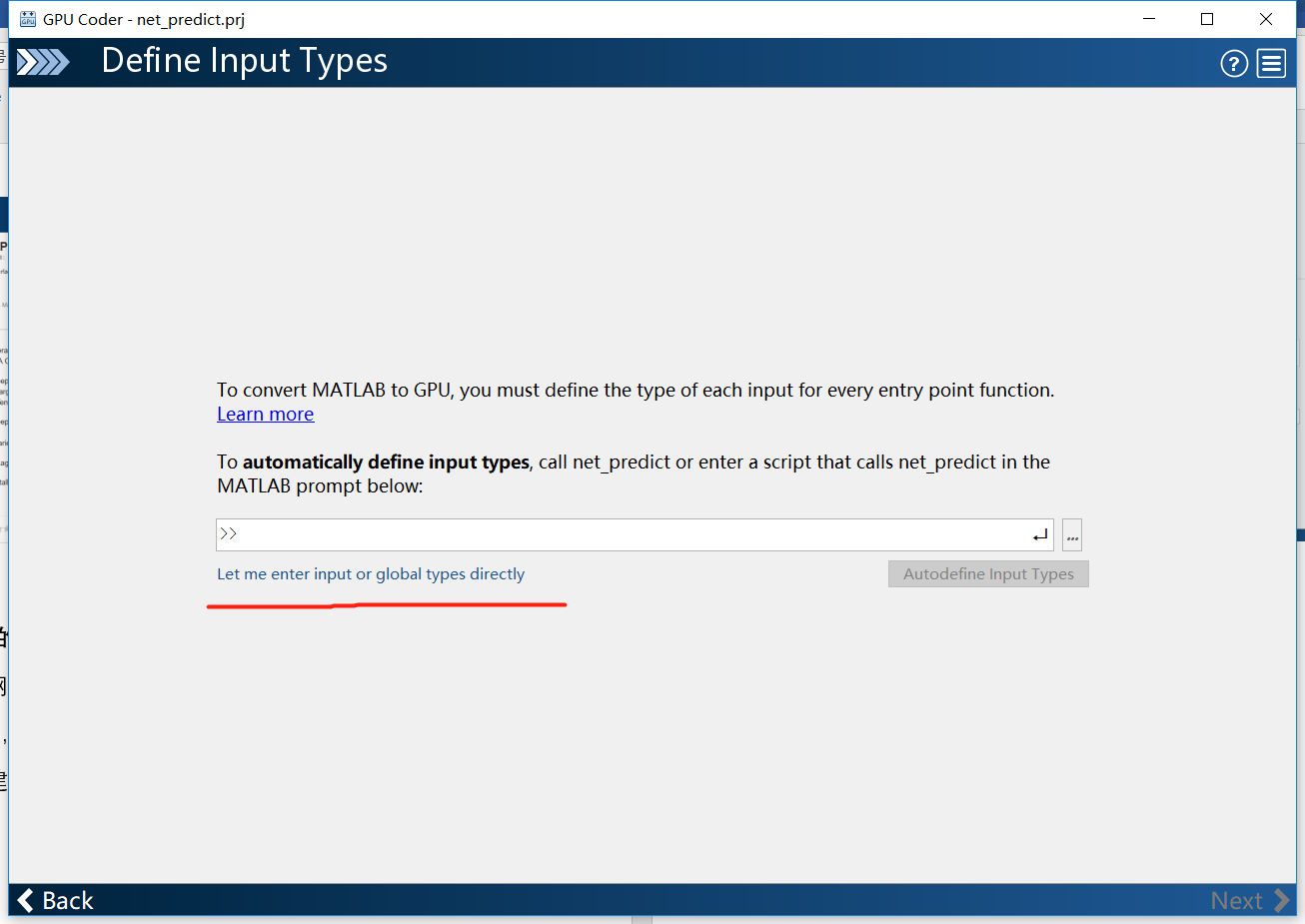
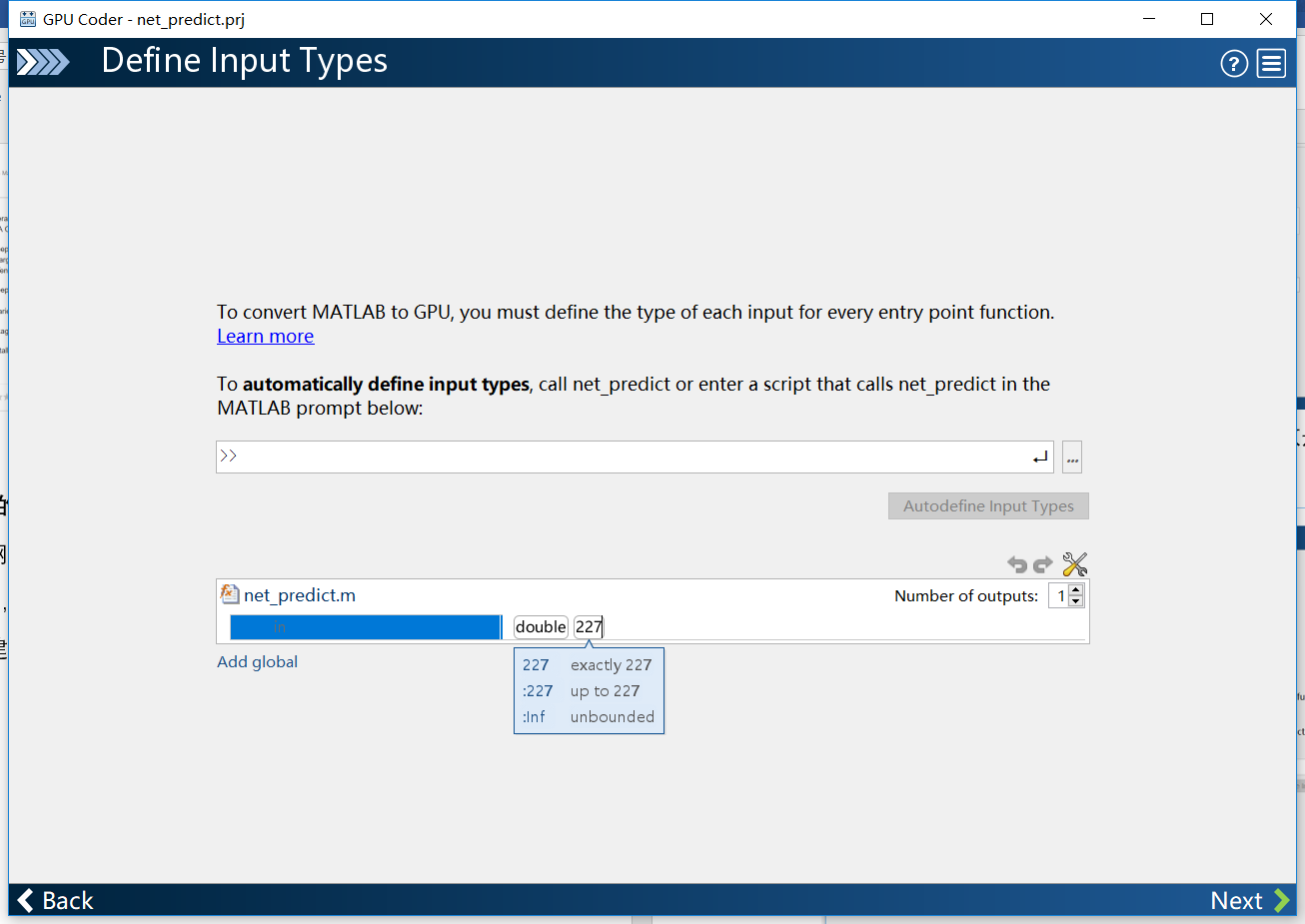


**App里面的GpuCoder:**

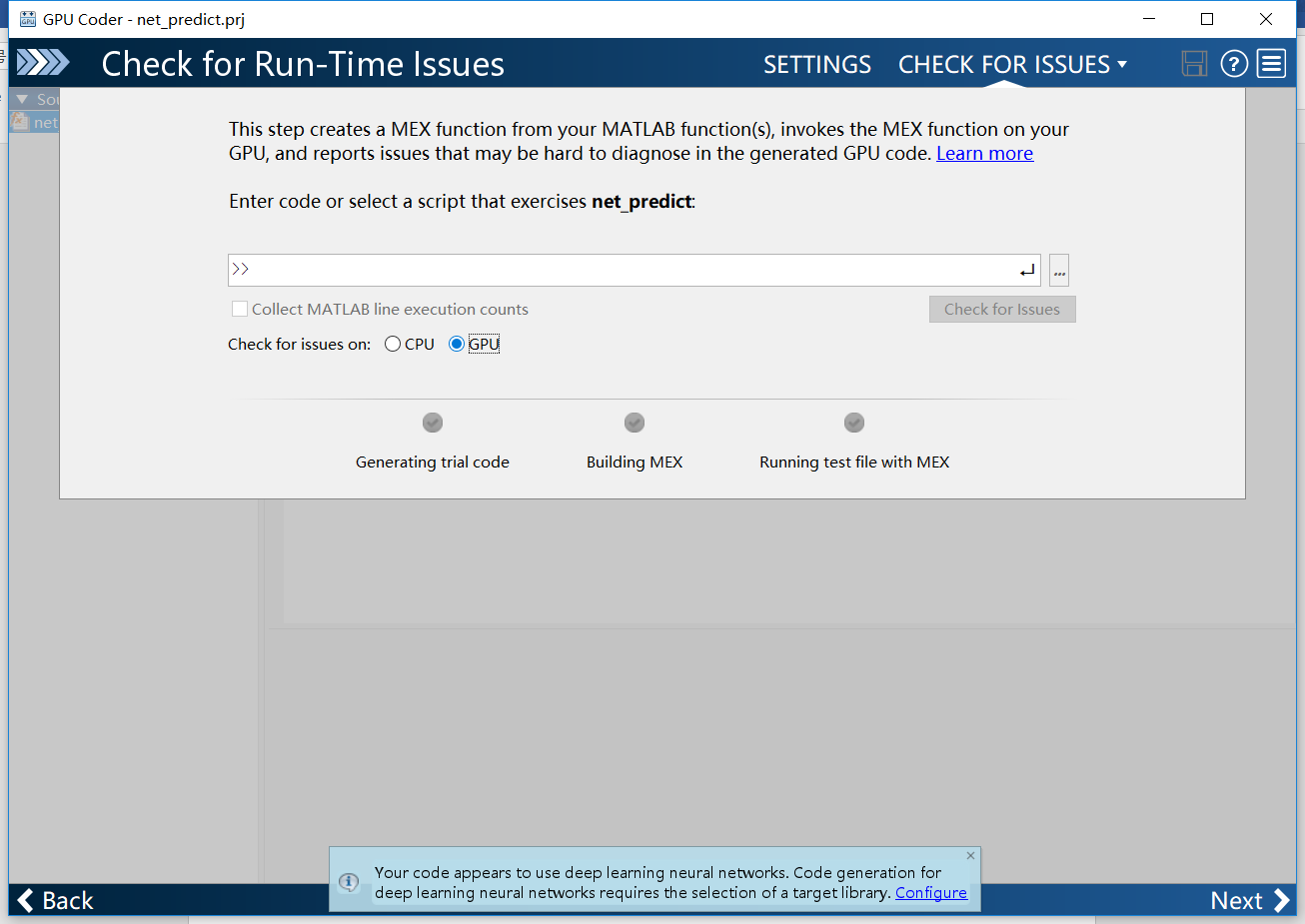
这里选择刚才建立的.m函数脚本，冉恒点下一步，会生成.prj的文件，是gpucoder的工程文件，如果以前有的话，这里是不能建立的



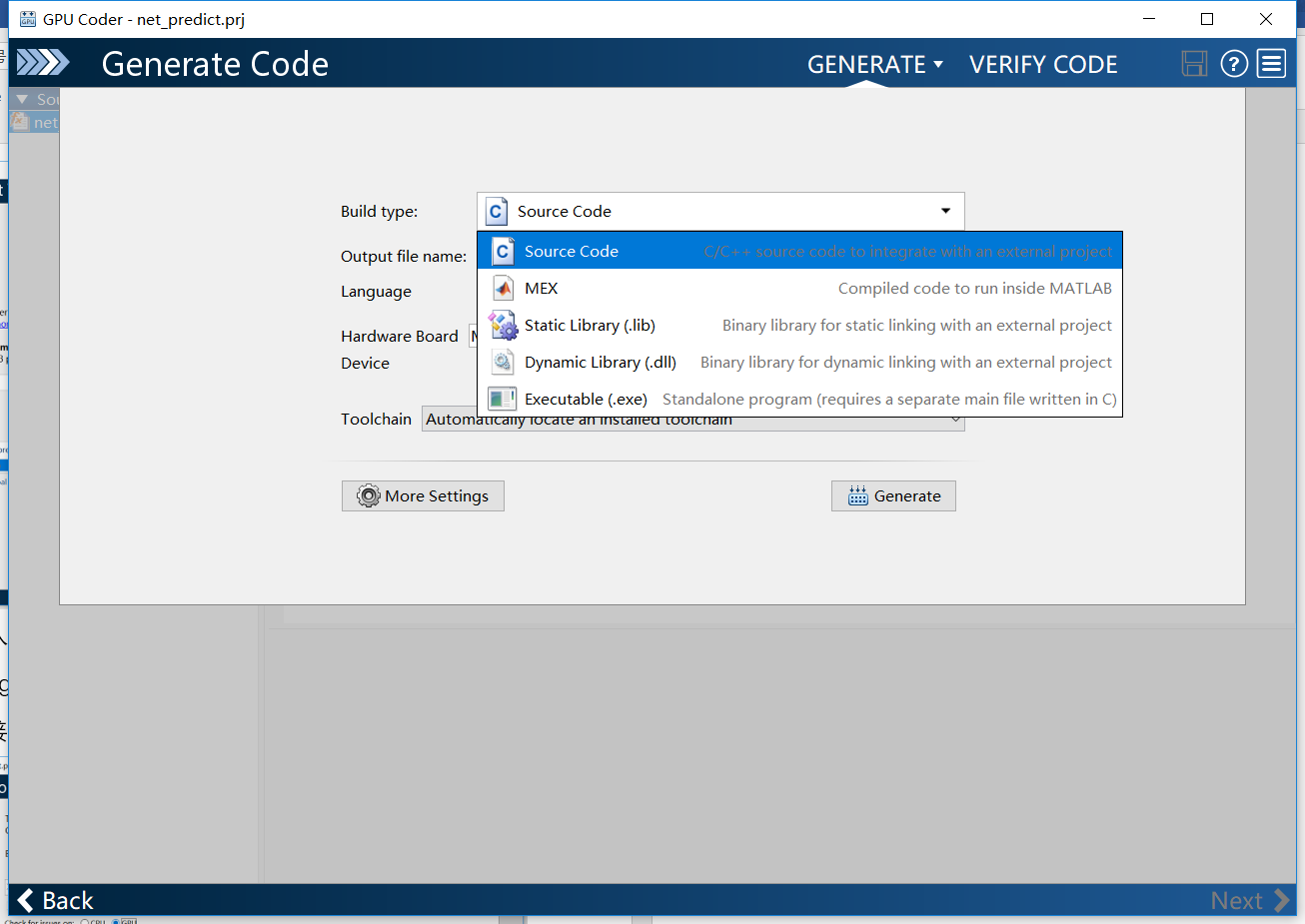
然后点击这个蓝连接，进行参数的设置，参数是你神经网络输入的尺寸，比如alexnet是227\*227\*3

 直接点击输入227\*227\*3，然后点击下一步

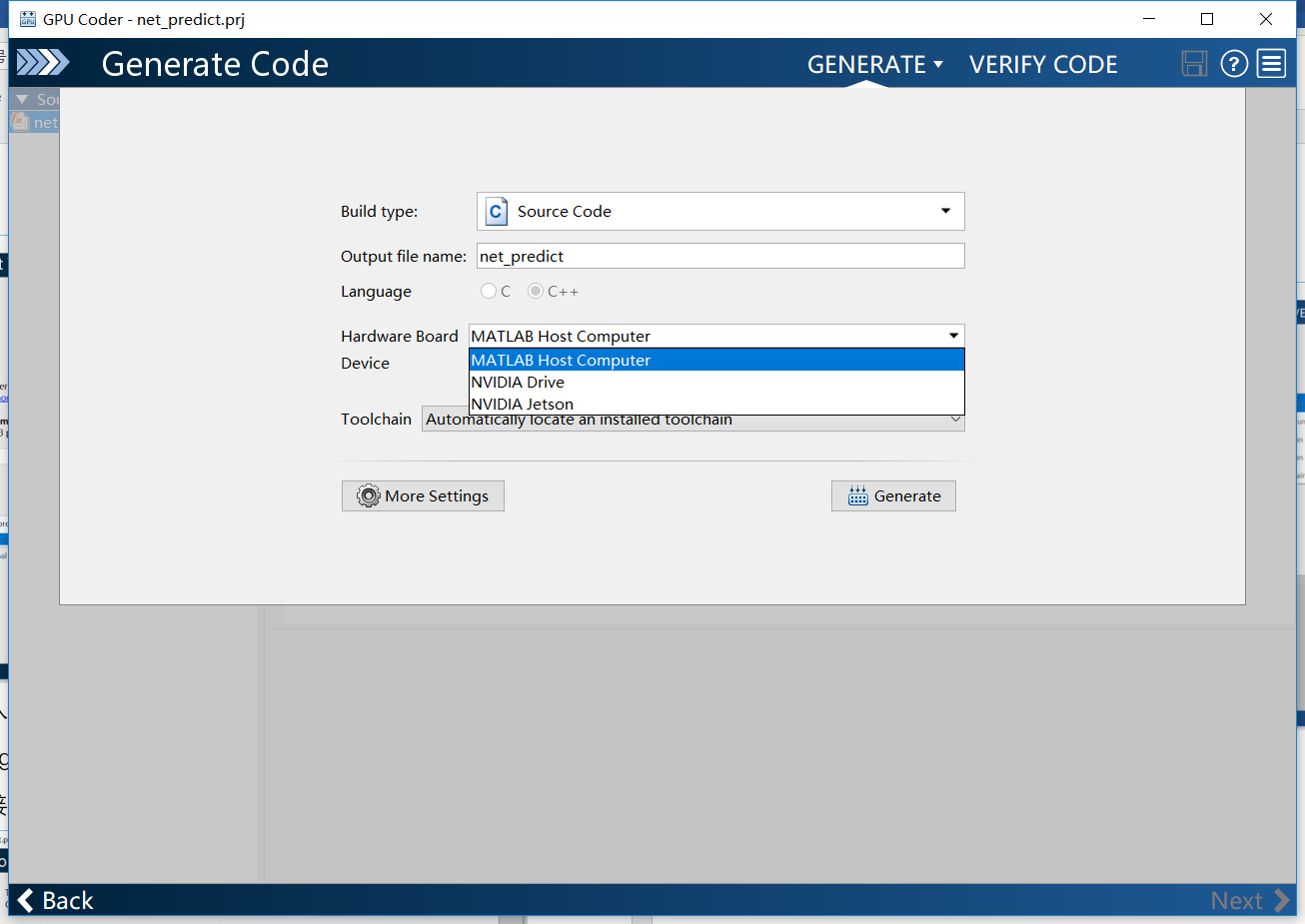
这里选择gpu，上面是干什么的我现在也不知道。。。但是不用管，直接点下一步



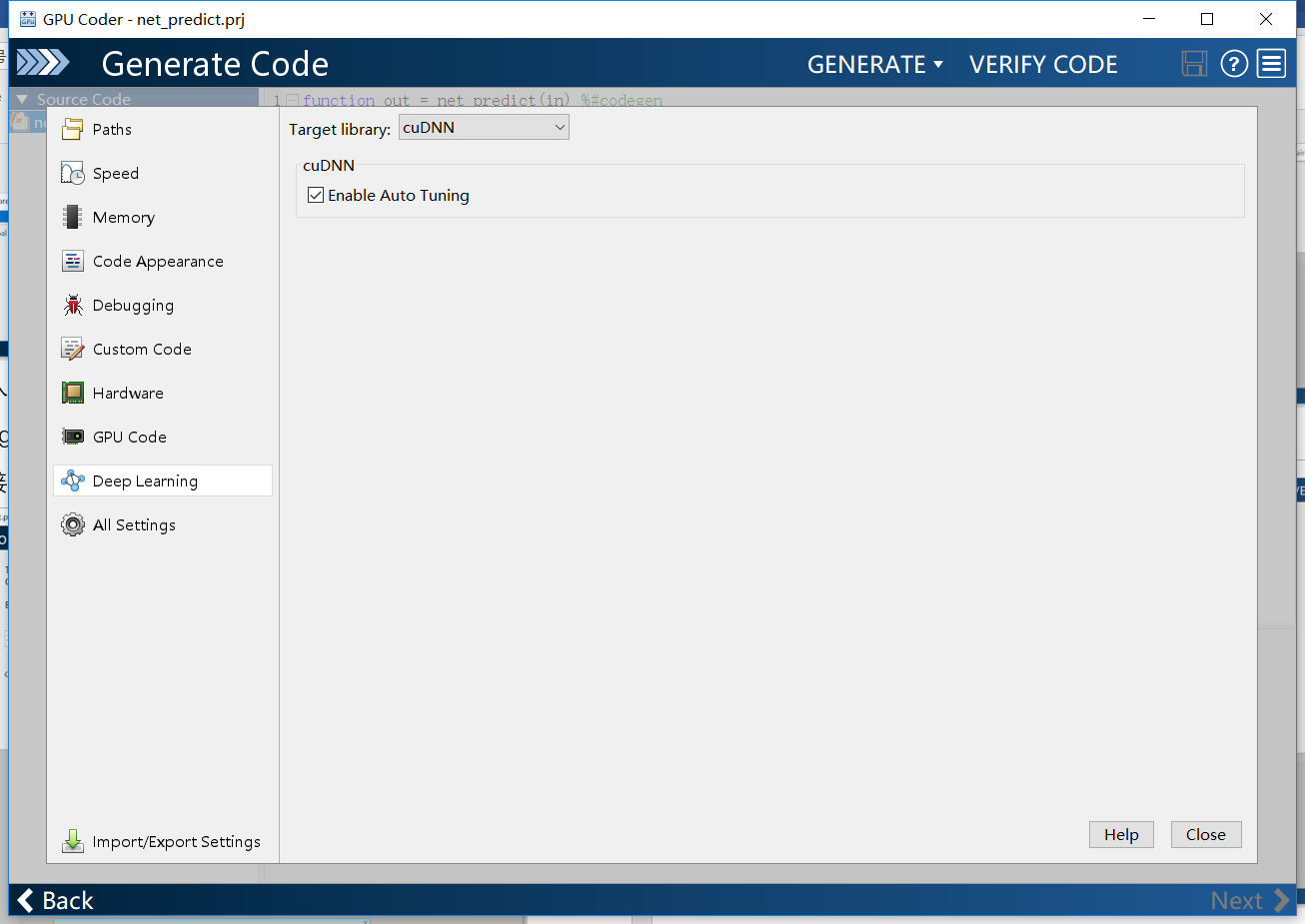
这里选择生成代码类型



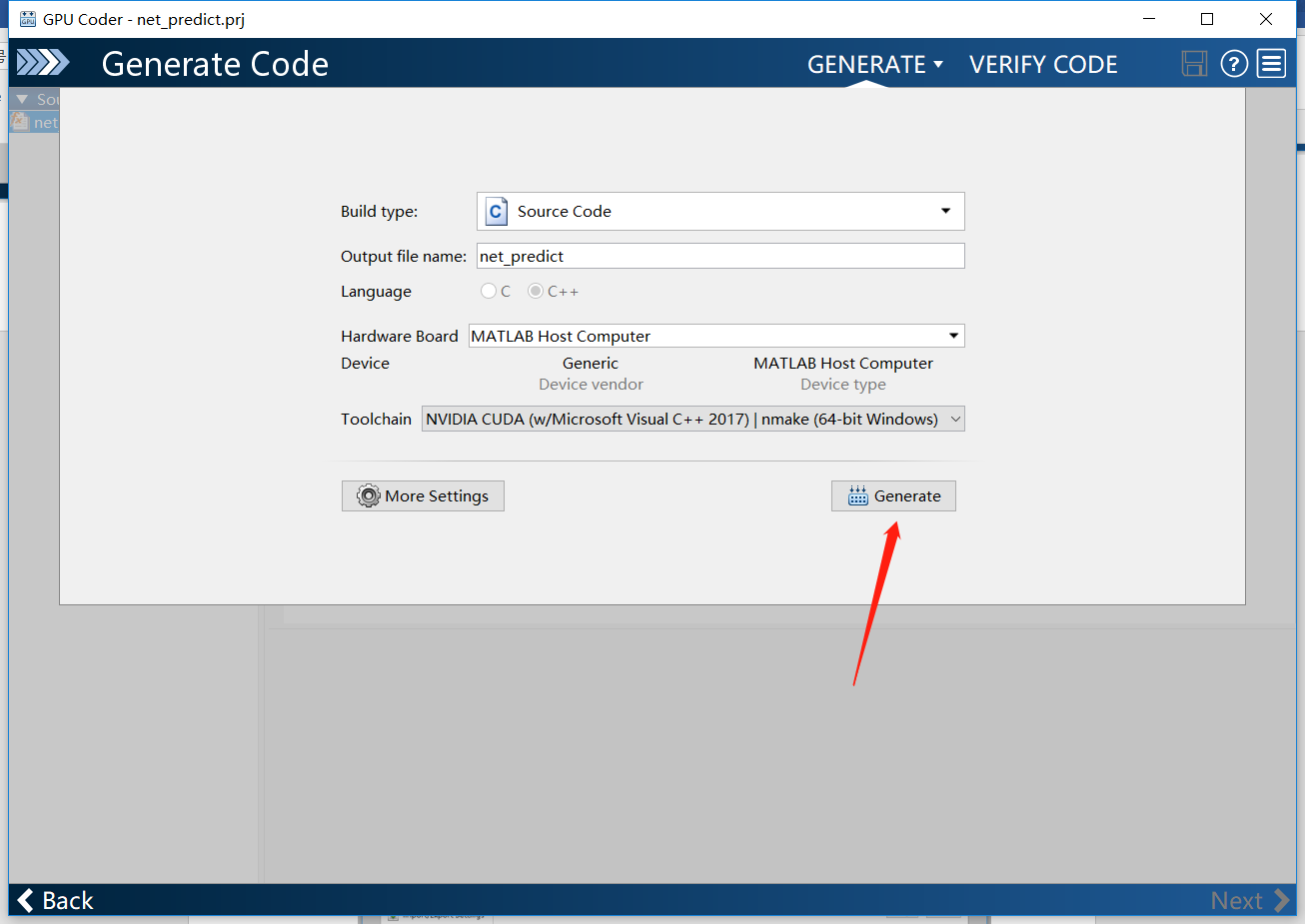
这里选择生成时的gpu类型



设置里面可以选择一些导出的设置



设置完成后，点生成（generate）



生成c++代码成功

