**Model Infer在Linux的编译过程记录**

为了能够有一个可以调试的model\_infer程序，我尝试把现有paddlex/deploy/cpp/model\_deploy的东西移植到IDE中去。

- 一开始尝试用clion发现对cmakelist的错误提示不够智能，不好用，改用qt creator发现不错。于是把paddlex的cmakelist稍作修改，保留deploy/cpp的形式生成两段式的2个cmakelists，思路是首先生成exe文件看是否能跑通，然后再生成so文件，但这整个过程浪费在了很多坑上，实际上完全没必要。使用paddlex自带的脚本已经可以非常简单生成exe和so文件了，虽然填坑过程学了不少东西，但导致开发进度及其缓慢。

采用的环境“

- 系统：ubuntu16.04

- 硬件：1080Ti

- cuda10.2 /usr/local/cuda/lib64

- cudnn7.6.5 /usr/local/cuda/lib64

- tensorRT6.0.1.5 /media/ubuntu/4430C54630C53FA2/SuLiang/Mysoftwares/TensorRT-6.0.1.8

- paddle inference /media/ubuntu/4430C54630C53FA2/SuLiang/Mysoftwares/paddle\_inference/paddle\_inference

- opencv3.4.6 /home/ubuntu/suliang\_git/PaddleX-develop/deploy/cpp/deps/opencv3.4.6gcc4.8ffmpeg

- openssl /home/ubuntu/suliang\_git/PaddleX-release-2.0.0/deploy/cpp/deps/openssl-1.1.0k

参考的编译步骤：deploy/cpp.docs/compile/paddle/linux.md

**任务1：编译exe**

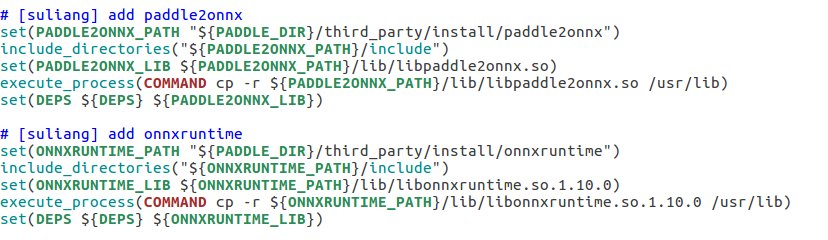
- 下载paddlex, paddleinference

- 修改build.sh

- 执行：sh script/build.sh

理论上整个过程就结束了，比windows更简单。

但由于使用了paddleinference2.3，增加了2个第三方库，而在paddlex2.0的cmakelists中则没有包含这部分，导致编译出错。解决办法就是手动在demo/cmakelist.txt中增加这两个库的链接，如下所示：

**任务2：编译动态库so**

- 把修改过的model\_deploy文件夹放到指定位置：里边针对trt的auto tune进行修改

- 把修改过的demo文件夹下的model\_infer.cpp放到相应位置：里边针对接口进行了调整，增加了trt的部分。

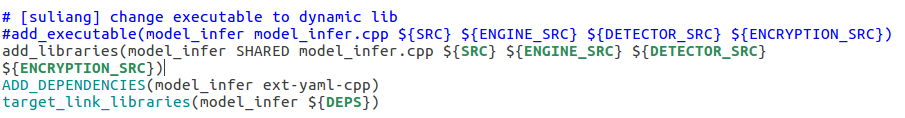
- 还需要修改部分代码：

- 函数导出部分去掉\_\_declspec(dllexport)因为这是专属windows的；把UCHAR替换为unsigned char因为这个在linux找不到；

- 然后修改demo/cmakelist：把可执行文件调整为动态库文件

- 然后修改源码中路径的\\改为根据系统不同决定用\\还是/做路径拼接

最后重新执行sh script/build.sh即可。相当于仅生成lib/libmodel\_infer.so，其他不变。

**任务3：调用动态库**

在qt中通过UI引入libmodel\_infer.so，同时也需要搞定依赖的那些a文件和so文件。

需要拷到/usr/lib/中去的动态库so文件包括：

- opencv：libopencv\_xx.so

- paddle\_inference: libpaddle\_inference.so(如果用trt则一般链接so动态库)

- mkldnn: libdnn.so, libmkldnn.so, libiomp5.so, libmklml\_intel.so

- tensorRT: nvinfer.so, nvinfer\_plugin.so, myelin.so

- model\_infer: libmodel\_infer.so

- onnxruntime: libonnxruntime.so.1.10.0

- paddle2onnx: libpaddle2onnx.so

- yaml-cpp: libyaml-cpp.so

相关so文件都需要逐个拷贝到/usr/lib，采用sudo cp lib/nvinfer.so\* /usr/lib完成

问题：TensorRT在qt中编译通过但运行找不到so文件，即使已经拷贝到/usr/lib也不行。最后解决方案是在/etc/ld.so.conf中添加trt/lib路径然后ldconfig后才找到

问题：paddleinference编译采用的protobuf版本跟本机默认版本不匹配：paddle inference2.3使用的protobuf版本为3.18.0(在include/google/protobuf/stubs/common.h看得到)，但本机还是protobuf2.6.1。解决办法参考：https://www.cnblogs.com/zhehan54/p/9983794.html，删掉了qt中的libgtk3.0.so，同时用sudo 启动sudo ./qtcreator

### 调试总结：

1. 导入cmakelists的qt项目，默认是采用cmake而不是qt自己的qmake
2. qt creator可以在Projects/Buld/Cmake/Key中输入相关环境变量

### 错误总结：

1. 报错：cc1plus: fatal error: .cpp.dd: No such file or directory compilation terminated. 该错误导致.o文件无法生成

主要是因为优化等级错误造成，原来写成了-o3，改为-O3即可

1. 报错：libdl.so: error adding symbols: DSO missing from command line。该错误导致主程序无法完成链接

主要原因据说是缺少相应的库，看了下发现paddle\_inference用的是.a的静态库，改为使用动态库后，发现打印出来的还是.a静态库。于是重新clear冲洗scan project，这样就是动态库

1. 报错：yaml-cpp找不到参考

仔细查看找不到参考的yaml-cpp的来源，发现居然是从/usr/include里边取到yaml-cpp的头文件，但我从来没往那放头文件阿，而且这样的头文件跟我实时编译的库文件就不匹配了，于是果断sudo rm -rf删掉/usr/include里边的yaml-cpp文件夹，重新clear重新scan，一切，build的一切，都绿了，瞬间high上天

日期记录

4.26 开始配置ubuntu环境，配置opencv/qt环境

4.27 开始整理cmakelists，在qt中搭建合成的cmakelists，反复尝试始终报错，怀疑qt有问题不适合这种方式

4.28 重新用paddlex源码来编译测试，经过多次尝试编译成功，说明源码没问题，改用clion尝试替换qt作为调试工具，安装破解，重新测试，还是报错

4.29 重新捡回qt，因为他的cmake过程提示比clion好很多。重新搭建简单版本的多cmakelists项目结构进行测试，重新加入yaml-cpp检查测试效果，在简单版本验证好以后再次挪回主程序。发现1号报错依旧，改为bing一下看看错误，偶然bing搜索发现一个建议，检查找到-o3写错改为-O3一号错误消失，出现2号错误，于是尝试改为动态库，2号错误消失，出现3号错误，仔细检查发现路径问题，去掉/usr/include的代码后，build就全绿了。