1. 安装caffe各种依赖库

之前已在Ubuntu14.04上源码编译了各种所需依赖库，所有include和lib文件都存放在/caffelib文件夹。坑一：ubuntu14.04自带gcc是4.8.x，ubuntu16.04自带gcc5.6.x，直接将依赖库移植过来编译caffe时报错undefined reference to protobuf, 解决办法：gcc降级

sudo apt-get install g++-4.9

sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.9 20

sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-5 10

sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-4.9 20

sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-5 10

sudo update-alternatives --install /usr/bin/cc cc /usr/bin/gcc 30

sudo update-alternatives --set cc /usr/bin/gcc

sudo update-alternatives --install /usr/bin/c++ c++ /usr/bin/g++ 30

sudo update-alternatives --set c++ /usr/bin/g++

以上操作在docker内的话需去掉sudo。

(以后也可在ubuntu16.04的gcc5.x环境下源码编译所有依赖库并保留一份）

1. 安装Mvapich2

下载mvapich2-2.2.tar.gz，由于cuda9.0计算能力较高，所以要将mvapich2的Makefile文件里的cm\_20改为cm\_30或以上。

再按如下编译：

./configure --prefix=/caffelib --with-cuda=/usr/local/cuda --enable-cuda --enable-threads=multiple --enable-cxx--enable-rdma-cm --with-rdma

坑二：fresh的ubuntu系统可能会提示一些库文件没有安装，按提示安装如下文件：

apt-get install gfortran

apt-get install byacc

apt-get install librdmacm

apt-get install libibverbs-dev

再 make -j8 & make install

1. 安装nccl2.0

上nvidia官网下载nccl\_2.1.15-1+cuda9.0\_x86\_64.txz解压到/nccl2.0

1. 编译caffe，git clone <https://github.com/Caffe-MPI/Caffe-MPI.github.io.git>

进入文件夹修改其Makefile.config如下：

USE\_CUDNN:=1

USE\_NCCL:=1

# CPU\_Only:=1

CUDA\_ARCH加上-gencode arch=compute\_30,code=sm\_30和

-gencode arch=compute\_35,code=sm\_35

OPENCV\_VERSION:=3

INCLUDE\_DIR和LID\_DIR加上/caffelib和/nccl2.0中的include和lib路径即可，以及分别加上 /usr/include和 /usr/lib/x86\_64-linux-gnu

WITH\_PYTHON\_LAYER=1

再make all -j32； make pycaffe，其中make pycaffe时可能会报错找不到libboost\_python，坑三：由于是新系统，发现python都没装，因此执行如下操作：

apt-get install python

apt-get install aptitude

aptitude install python-dev

apt-get install python-pip

pip install numpy

apt-get install python-numpy

apt-get install libboost-python-dev

再重新make pycaffe即可。

1. 测试

（1）Run data/cifar10/get\_cifar10.sh to get cifar10 data.

该步需下载cifar-10-python.tar.gz ，由于网速问题可提前下好传到宿主机后再docker cp 到容器中的相应路径下，然后注释脚本中的wget语句

（2）Run examples/cifar10/create\_cifar10.sh

（3）Run mpi\_train\_quick.sh to train the net.

Example of mpi\_train\_quick.sh script. mpirun -host node1,node2 --mca btl\_openib\_want\_cuda\_gdr 1 --mca io ompio -np 2 -npernode 1 ./build/tools/caffe train --solver=examples/cifar10/cifar10\_quick\_solver.prototxt --gpu=0,1,2,3

该步在单机单gpu上无法运行，需装infiniband和无线网卡驱动。

自定义测试脚本如下：

mpirun -np 1 ./build/tools/caffe train --solver=examples/cifar10/cifar10\_quick\_solver.prototxt --gpu=0

由于只有一块显卡因此np只能设为1（运行时间42s）, 若设置CPU\_ONLY=1重新编译caffe-mpi在运行后设置np>1也能跑（我设置的np=2），但时间大大增加（运行程序时中间会卡住，时间大概3h20m），还可令TEST\_FP16：=1重编译后再跑程序查看其性能（好像也是42s）。