CNStream Docker Images

Build docker images for CNStream.

Directory tree

```
.
├── build-cnstream-image.sh
├── Dockerfile.16.04
├── load-image-cnstream.sh
├── rm-all-docker-container.sh
└── run-container-cnstream.sh
```

Clone

```
git clone https://github.com/CambriconKnight/cnstream-docker-image.git
```

```
cam@cam-3630:/data/github$ git clone https://github.com/CambriconKnight/cnstream-docker-
Cloning into 'cnstream-docker-image'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 14 (delta 3), reused 13 (delta 2), pack-reused 0
Unpacking objects: 100% (14/14), done.
Checking connectivity... done.
cam@cam-3630:/data/github$ cd cnstream-docker-image
cam@cam-3630:/data/github/cnstream-docker-image$ ls
build-cnstream-image.sh  Dockerfile.16.04  load-image-cnstream.sh  README.md  rm-all-doc
cam@cam-3630:/data/github/cnstream-docker-image$
```

Build

```
./build-cnstream-image.sh
```

```
cam@cam-3630:/data/github/cnstream-docker-image$ ./build-cnstream-image.sh
Cloning into 'cnstream'...
remote: Enumerating objects: 1005, done.
remote: Counting objects: 100% (1005/1005), done.
remote: Compressing objects: 100% (684/684), done.
remote: Total 5920 (delta 422), reused 710 (delta 292), pack-reused 4915
Receiving objects: 100% (5920/5920), 192.19 MiB | 3.02 MiB/s, done.
Resolving deltas: 100% (3113/3113), done.
Checking connectivity... done.
File(neuware-mlu270-1.5.0-1_Ubuntu16.04_amd64.deb): Not exist!
Copy your neuware package(neuware-mlu270-1.5.0-1_Ubuntu16.04_amd64.deb) into the directo
eg:cp -v /data/ftp/v1.5.0/neuware/neuware-mlu270-1.5.0-1_Ubuntu16.04_amd64.deb ./cnstrea
cam@cam-3630:/data/github/cnstream-docker-image$ cp -v /data/ftp/v1.5.0/neuware/neuware-
'/data/ftp/v1.5.0/neuware/neuware-mlu270-1.5.0-1_Ubuntu16.04_amd64.deb' -> './cnstream/r
cam@cam-3630:/data/github/cnstream-docker-image$ ./build-cnstream-image.sh
Directory(cnstream): Exists!
File(neuware-mlu270-1.5.0-1_Ubuntu16.04_amd64.deb): Exists!
==================== build image ====================
Sending build context to Docker daemon  159.3MB
Step 1/11 : FROM ubuntu:16.04
 ---> c522ac0d6194
......
......
......
Step 11/11 : WORKDIR /root/CNStream/samples/demo
 ---> Running in 3d594d45f6b8
Removing intermediate container 3d594d45f6b8
 ---> 080fdb4ca3c3
Successfully built 080fdb4ca3c3
Successfully tagged ubuntu16.04_cnstream:v1.5.0
==================== save image ====================
-rw------- 1 cam cam 1351632896 12月 15 22:03 ubuntu16.04_cnstream-v1.5.0.tar.gz
cam@cam-3630:/data/github/cnstream-docker-image$
```

Load

```
#加载Docker镜像
./load-image-cnstream.sh
```

Run

```
#启动Docker容器
./run-container-cnstream.sh
```

Test

YOLOv3

```
#硬件平台：MLU270、MLU220-M.2
#软件环境：Docker（ubuntu16.04_cnstream-v1.5.0.tar.gz）
#运行实例：基于CNStream的YOLOv3运行实例
#业务流程：读取视频文件 --> MLU硬件解码 --> MLU硬件推理 --> 叠加OSD信息 --> RTSP推流输出
#所用插件：DataSource；Inferencer；Osd；RtspSink
#离线模型：http://video.cambricon.com/models/MLU270/yolov3/yolov3_offline_u4_v1.3.0.cambr
#启动脚本：/root/CNStream/samples/demo/detection/mlu270/run_yolov3_mlu270.sh
#        /root/CNStream/samples/demo/detection/mlu220/run_yolov3_mlu220.sh
#配置文件：/root/CNStream/samples/demo/detection/mlu270/yolov3_mlu270_config.json
#        /root/CNStream/samples/demo/detection/mlu220/yolov3_mlu220_config.json
#推流地址：执行run_yolov3_mlu270.sh后，会把RTSP推流地址写入到本地文件RTSP_url_names.txt中。
#        /root/CNStream/samples/demo/detection/mlu270/RTSP_url_names.txt
```

推理结果摘选：

```
root@cam-3630:~# cd /root/CNStream/samples/demo/detection/mlu270/
root@cam-3630:~/CNStream/samples/demo/detection/mlu270# ./run_yolov3_mlu270.sh
--2020-12-15 14:24:42--  http://video.cambricon.com/models/MLU270/yolov3/yolov3_offline_
Resolving video.cambricon.com (video.cambricon.com)... 182.92.212.157
Connecting to video.cambricon.com (video.cambricon.com)|182.92.212.157|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 98652692 (94M) [text/plain]
Saving to: 'yolov3_offline_u4_v1.3.0.cambricon'
yolov3_offline_u4_v1.3.0.cambricon              100%[==============================
2020-12-15 14:25:28 (2.07 MB/s) - 'yolov3_offline_u4_v1.3.0.cambricon' saved [98652692/9
--2020-12-15 14:25:28--  http://video.cambricon.com/models/MLU270/yolov3/label_map_coco.

......
......
......


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
=================================[ osd Performance ]=============================
[stream id] stream_0  -- [latency] avg:    4.3ms, min:    0.9ms, max:    33.6ms, [frame c
[stream id] stream_1  -- [latency] avg:    4.5ms, min:    0.7ms, max:    29.7ms, [frame c
----------------------------------------------------------
Throughput over the last 2s
Total :        -- [frame count]: 16, [processing time(s)]:   0.038891, [fps]:   411.5
----------------------------------------------------------
Average throughput over the process
Total :        -- [frame count]: 3568, [processing time(s)]:   7.968857, [fps]:   447.8
=================================[ Pipeline Performance ]=========================
End node of pipeline: rtsp_sink
[stream id] stream_0  -- [latency] avg:  271.8ms, min:   84.1ms, max:  510.7ms, [frame c
[stream id] stream_1  -- [latency] avg:  271.1ms, min:   82.1ms, max:  527.8ms, [frame c
----------------------------------------------------------
Throughput over the last 2s
[stream id] stream_0  -- [frame count]: 9, [processing time(s)]:   0.246877, [fps]:    36
[stream id] stream_1  -- [frame count]: 7, [processing time(s)]:   0.289882, [fps]:    24
(* Note: Performance info of pipeline is slightly delayed compared to that of each strea
Pipeline :     -- [frame count]: 17, [processing time(s)]:   0.289882, [fps]:   58.7
----------------------------------------------------------
Average throughput over the process
[stream id] stream_0  -- [frame count]: 1784, [processing time(s)]:  59.782535, [fps]:
[stream id] stream_1  -- [frame count]: 1784, [processing time(s)]:  59.804728, [fps]:
(* Note: Performance info of pipeline is slightly delayed compared to that of each strea
Pipeline :     -- [frame count]: 3568, [processing time(s)]:  59.813077, [fps]:   59.7

Total : 59.700000
CNSTREAM CORE I1215 14:26:30.512209 27883] [MyPipeline] Stop
WARNING: Logging before InitCNStreamLogging() is written to STDERR
CNSTREAM CORE I1215 14:26:30.512708 27887] [MyPipeline] stop updating stream message
I1215 14:26:30.513303 27883 mlu_context.cpp:82] Cambricon runtime destroy
root@cam-3630:~/CNStream/samples/demo/detection/mlu270#
```

SSD

```
#硬件平台：MLU270
#软件环境：Docker（ubuntu16.04_cnstream-v1.5.0.tar.gz）
#运行实例：基于CNStream的SSD运行实例
#业务流程：读取视频文件 --> MLU硬件解码 --> MLU硬件推理 --> 叠加OSD信息 --> 本地显示（带有GUI界面
#所用插件：DataSource；Inferencer；Osd；Displayer
#离线模型：http://video.cambricon.com/models/MLU270/Primary_Detector/ssd/resnet34_ssd.cam
#启动脚本：/root/CNStream//samples/demo/run.sh
#配置文件：/root/CNStream/samples/demo/detection_config.json
#推流地址：执行run.sh后，会把检测的实时画面显示到屏幕上。
#注意事项：1.需在GUI终端运行； 2.需设置-Dbuild_display=ON重新编译CNStream；
#          3.需修改detection_config.json文件中字段"show"值为"true"；
```

推理结果摘选：


```
#1.请确保在GUI终端运行
cd /root/CNStream/build
#2.设置-Dbuild_display=ON重新编译CNStream
cmake -Dbuild_display=ON ../
cd /root/CNStream/samples/demo
#3.修改detection_config.json文件中字段"show"值为"true"；
vim detection_config.json
#运行实例
./run.sh
```

```
root@cam-3630:~/CNStream/samples/demo# ./run.sh
~/CNStream/data/models/MLU270/Primary_Detector/ssd ~/CNStream/samples/demo
resnet34_ssd.cambricon offline model exists.
label_voc.txt exists.
~/CNStream/samples/demo
/root/CNStream/samples/demo
CNRT: 4.6.0 e158c88
Register object named [PreprocYolov3]
Register object named [PreprocCpu]
Register object named [ObjPreprocCpu]
Register object named [PostprocSsd]
Register object named [PostprocYolov3]
Register object named [PostprocFakeYolov3]
Register object named [PostprocClassification]
Register object named [ObjPostprocClassification]
Register object named [CarFilter]
WARNING: Logging before InitGoogleLogging() is written to STDERR
I1215 15:46:50.022884 23753 demo.cpp:328] CNSTREAM VERSION:v5.3.0
CNSTREAM CORE I1215 15:46:50.023119 23753] Add Module source to pipeline
CNSTREAM CORE I1215 15:46:50.023214 23753] Add Module detector to pipeline
CNSTREAM CORE I1215 15:46:50.023227 23753] Add Module osd to pipeline
CNSTREAM CORE I1215 15:46:50.023239 23753] Add Module displayer to pipeline
CNSTREAM CORE I1215 15:46:50.023257 23753] Link Module osd-->displayer
CNSTREAM CORE I1215 15:46:50.023265 23753] Link Module detector-->osd
CNSTREAM CORE I1215 15:46:50.023269 23753] Link Module source-->detector
CNSTREAM CORE I1215 15:46:50.023288 23753] Directory [perf_database/] exists.
before init
before create window
before create texture
I1215 15:46:50.474404 23753 mlu_context.cpp:99] Cambricon runtime init success.
I1215 15:46:50.500952 23753 model_loader.cpp:191] Load function from offline model succe
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
===============================[ displayer Performance ]============================
......
......
......
[stream id] stream_0  -- [frame count]: 1491, [processing time(s)]:  59.740551, [fps]:
[stream id] stream_1  -- [frame count]: 1491, [processing time(s)]:  59.740547, [fps]:

(* Note: Performance info of pipeline is slightly delayed compared to that of each strea
Pipeline :      -- [frame count]: 2982, [processing time(s)]:  59.740566, [fps]:   50.0

Total : 50.000000
CNSTREAM CORE I1215 15:46:16.409847 20398] [MyPipeline] Stop
WARNING: Logging before InitCNStreamLogging() is written to STDERR
CNSTREAM CORE I1215 15:46:16.410425 20402] [MyPipeline] stop updating stream message
I1215 15:46:16.420593 20398 mlu_context.cpp:82] Cambricon runtime destroy
root@cam-3630:~/CNStream/samples/demo#
```