北京晶视智能科技有限公司

Auto Tuning Tool Guide

文档版本 v1.1

发布时间 2019-09-25

商标声明

TBD

注意

TBD

北京晶视智能科技有限公司

地址：

邮编：

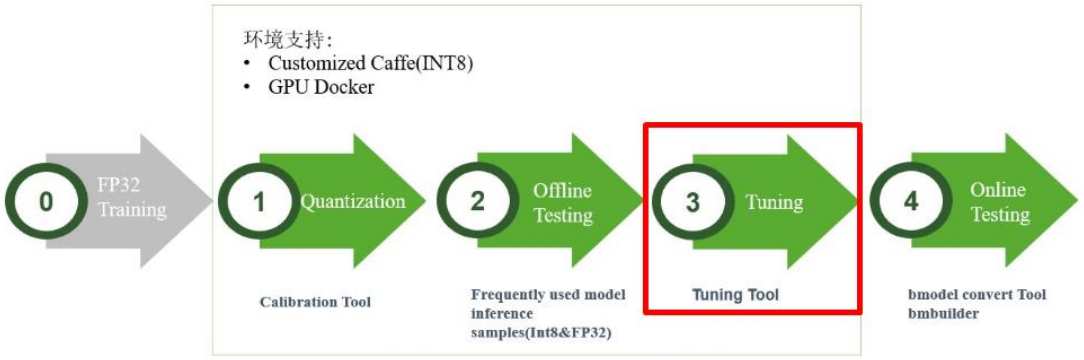
网址：

目录

1. 前言

概述

本文档说明精度调整工具 Auto Tuning Tool 的使用方法和注意事项。Auto Tuning Tool 用于将量化后的 INT8 模型做精度提升.



读者对象

本文档主要适用于以下工程师：

- 算法工程师
- 系统整合工程师

修订记录

版本号	作者	审阅人	时间	修改描述
1.1	褚洪君		2019.09.26	•
1.0	汤道言、褚洪君	朱晓军、王亮	2019.09.20	初版

2. 准备工作

我们**强烈建议**您安装 GPU docker 环境，这样会大幅度提升模型处理的效率。相关的工作(Tuning)，用 CPU 几乎无法完成。

2.1. Host PC 环境

- ubuntu 18.04 or 16.04
- python2.7
- docker installed
- nvidia-docker installed (if you want to setup GPU Docker)

2.2. GPU Docker 环境建立(推荐)

目前我们只提供基于 CUDA9.0 的 prebuilt Caffe 库。所以请确认您主机上的 GPU 显卡驱动是与 CUDA9.0 兼容的。请按如下步骤建立环境:

1. 请把 cafffe_gpu 目录里的库文件覆盖到 caffe 目录。
2. build & run docker image

```
1. sudo nvidia-docker build -t bmcalibration:2.0_gpu -
   f docker/Dockerfile_GPU .
2. sudo nvidia-docker run -v /workspace:/workspace -it bmcalibration:2.0_gpu
```

这样，GPU Docker 环境即建立完成。请调用如下的 caffe api 来使能 GPU caffe.

```
1. caffe.set_mode_gpu()
2. caffe.set_device(device_id)    #device_id means the GPU id on your machine.
```

2.3. CPU Docker 环境建立

请按如下命令建立环境:

```
sudo docker build -t bmcalibration:2.0_cpu -f docker/Dockerfile_CPU .
sudo docker run -v /workspace:/workspace -it bmcalibration:2.0_cpu
```

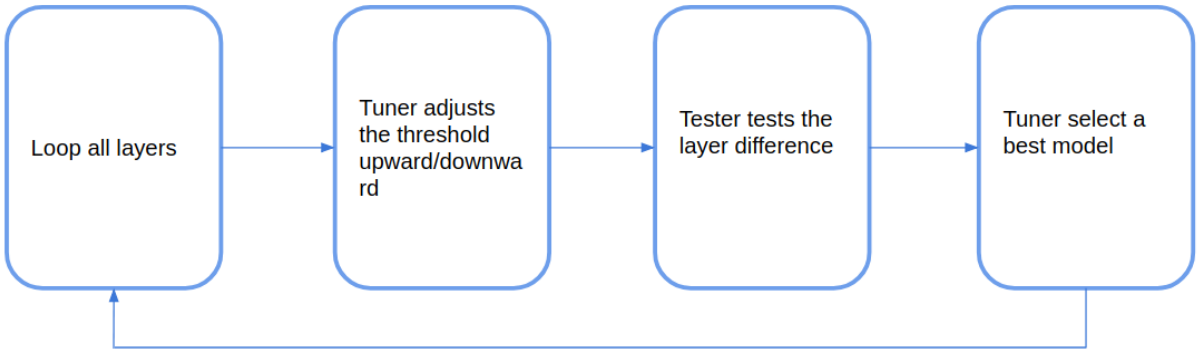
这样，CPU Docker 环境即建立完成。

3. Auto Tuning Tool 说明

3.1. 介绍

Auto Tuning tool 用于调整 calibration table 的 threshold. 通过分析 FP32 与 INT8 的 feature map 的差异，来逐层调整每个 layer 的 threshold 达到整体提升模型精度的目的。最终输出调整好的 INT8 模型。

3.2. 流程图



INT8 模型进到 Tuning tool 利用提供的图片样本计算从上到下模型每一个 layer 的 feature map 整体上与 FP32 的差异。之后上下调整 threshold 后，确认一个最小的 diff 相对应的 threshold 当做本层的 threshold . 以此类推将每一层调整完毕. 最终输出 model.

3.3. 推理数据选择与处理

推理数据用于计算 FP32 与 INT8 的 feature map 差异。Tuning tool 会利用全部由 data_list 参数指定的图片最终算出差异. 推理数据的选择应该尽量覆盖所有实际的应用场景。数据的前处理方法与 Calibration Tool 类似，具体实现在 prepare_image function in test_utils.pyj. 这里我们提供了 image_params.json 用于数据的前处理。具体的参数说明如下：

Parameter	Description

size	The target image size([h w])
padding	Specify padding when resize or not
mirror	The mirror method. Using the traditional opencv flip. 1: horizontal flip 0: vertical flip -1: horizontal and vertical flip Ignore when other value.
color_format	The color format order. Only support 'rgb' or 'bgr'
r/g/b_mean	The mean value of r/g/b channel.
scale	The target image scale

3.4. Feature Map Difference

Feature map difference 的计算公式同样对结果产生影响。我们在 evaluation_utils.py 中提供了两种计算 difference 的方法。一种是计算平方距离 (square distance)的方法，以我们的经验这种方法对一些应用场景比较有效，比如图片分类、face align 场景。第二种方法是针对于 yolo，会计算 bbox, class, score 的 feature map 的 diff 平方距离。

3.5. Caffe prototxt input layer 写法

```
layer {
  name: 'data'
  type: 'Input'
  top: 'data'
  input_param {
    shape: { dim: 1 dim: 3 dim: 224 dim: 224 }
    enable_quantize: 1
  }
}
```

请注意红色的 enable_quantize 参数需要给，这样 caffe 会对 INT8 的 model 输入数据默认做量化。

4. 运行 Auto Tuning Tool

Auto Tuning Tool 的入口是 main.sh，具体的内容如下:

```
python main.py \
    --proto /home/tuning/yolov3.prototxt \
    --model /home/tuning/custom.caffemodel \
    --
calibration_proto "/home/tuning/bmnet_calibration_table.1x10.pb2" \
    --calibration_model "/home/tuning/bmnet_int8.1x10.caffemodel" \
    --output_path ./result_tune \
    --data_list /home/tuning/calibration/tuning/input.txt \
    --data_limit 20 \
    --image_params "image_params.json" \
    --ignore_layer_list 'data'
```

具体的参数说明如下:

Parameter	**Description**
model	FP32 caffe model
proto	FP32 caffe prototxt
calibration_model	原始的要去 tuning 的 int8 caffe model
calibration_proto	原始的用 calibration tool 生成的 calibration table pb2
output_path	存放 tune 完 model 的临时目录
data_list	more detail 用于算 diff 的 inference data ，参见 3.3 节．这里写入图片的绝对路径。
data_limit	用来算 diff 的推理图片的数目．并不是写在 data_list 里的图，都参与计算 diff. 以 data_limit 的数量为准 ．

image_params	数据前处理 script 的位置. 数据前处理 script 如何写参考 3.3 节。
ignore_layer_list	Ignore layers that you don’t want to tune.

Main.sh 将 main.py 的内容包装起来，具体处理会包含 test 及 tune 两个部分. Test 和 tune 的部分还有分别的 script test.sh 及. tune.sh. 您可以分别使用。

test.sh 生成 report.log 并记录想要 tuning 的 layer 的 INT8 和 FP32 的 diff. int8_layer 用于指定目标 tuning 的 layer. 如果为空，script 会产生 model 的 diff.

tune.sh 去调整一层的 threshold. tune_layer 指定要调整的 layer，tune_diff 是指定当前的 layer 的 diff 值. 这个 script 会在 output_path 里产生 tuning 过后的 model.

5. Auto Tuning Tool 实例

本章节以 Tuning Resnet50 为例，详细说明每一个步骤，以供您参考。

1. 准备 tuning 数据。

   需要输入数据包括:

   1) 量化后 INT8 模型的 pb2 文件，caffemodel 文件如下(请参见 calibration tool 实例):

      bmnet_Resnet50_calibration_table.pb2
      bmnet_Resnet50_int8.caffemodel

   2) FP32 模型的 prototxt 和 caffemodel，请按要求改 prototxt 的 input layer，如 3.5 章节。

   3) 用于计算的 diff 的图片，并把记录图片路径的 input.txt 路径给到 data_list 参数，并在 image_params.json 文件里修改图片需要做的前处理的参数。

2. 运行 Tuning tool 的。

   souce main.sh

   输入 log 如下:

...

3. 查看 tuning 结果。

Tuning 的过程因为会涉及到多次的 FP32 和 INT8 模型的推理，会持续一段时间才能结束，需耐心等待。

最后输出红框里的 model 即 tune 好的 model.