

# Documentation History

## play-with-torch

### 1. Section 1

#### 0. Tools

##### 0.1. Hardware

- + **RaspberryPi 4 Model B** Rev 1 (RPi), RAM: **4 GB** and Processor 4-core @ **1.5 GHz**
- + microSD Card **64 GB**
- + 5M USB Retractable Clip 120 Degrees WebCam Web Wide-angle Camera Laptop U7 Mini or Raspi Camera

##### 0.2. Tested Software

- + Raspbian 10 (Buster) **[armv7l] 32 bit**, install on RPi
- + PyTorch **[armv7l]**: torch 1.6.0 and torchvision 0.7.0
- + Python min. ver. 3.6

#### 1. Install the prerequisites

- + Test using above reference the result is:

In Raspbian, version python is 3.7.3

- + Source: <https://github.com/ljk53/pytorch-rpi>

**torch 1.6.0 [armv7l]**: [https://github.com/ljk53/pytorch-rpi/blob/master/torch-1.6.0a0%2Bb31f58d-cp37-cp37m-linux\\_armv7l.whl](https://github.com/ljk53/pytorch-rpi/blob/master/torch-1.6.0a0%2Bb31f58d-cp37-cp37m-linux_armv7l.whl)

Move file to

```
$ scp torch-1.6.0a0+b31f58d-cp37-cp37m-linux_armv7l.whl pi@xxx:/home/xxx/Downloads
pi@raspberrypi:~/Downloads $ pip3 install torch-1.6.0a0+b31f58d-cp37-cp37m-linux_armv7l.whl
```

- + Install packages

```
$ sudo apt install build-essential make cmake git python3-pip libatlas-base-dev
```

```
$ sudo apt install libssl-dev
```

```
$ sudo apt install libopenblas-dev libblas-dev m4 cython python3-yaml
```

- + Update cmake to > 3.15

```
$ wget https://github.com/Kitware/CMake/releases/download/v3.18.0-rc1/cmake-3.18.0-rc1.tar.gz
```

```
$ sudo apt install libssl-dev
```

```
$ cd cmake-3.18.0
```

```
$ mkdir build && cd build
```

```
$ cmake ..
```

```
$ make
```

```
$ sudo make install
```

```
$ sudo apt remove cmake
```

```
$ sudo ln -s /usr/local/bin/cmake /usr/bin/cmake
```

```
$ sudo ldconfig
```

- + Make WASP to 2048 MB

```
$ nano /etc/dphys-swapfile
```

```
CONF_SWAPFILE=2048M
```

```
$ sudo dphys-swapfile setup
```

```
$ sudo dphys-swapfile swapon
```

```
$ free -h
```

```
pi@raspberrypi:~ $ free -h
```

	total	used	free	shared	buff/cache	available
Mem:	3.7Gi	65Mi	2.0Gi	8.0Mi	1.6Gi	3.5Gi
Swap:	2.0Gi	0B	2.0Gi			

+ Test torch below:

```
pi@raspberrypi:~ $ python3
Python 3.7.3 (default, Jul 25 2020, 13:03:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> import numpy as np
>>> a = torch.from_numpy(np.random.randn(1, 100))
>>> print(a)
tensor([[ -1.5612e+00,  -1.1646e-01,   3.7521e-01,  -9.3571e-01,  -2.7875e-01,
          3.6622e-01,   1.2217e+00,   6.9403e-01,  -5.2061e-01,   1.3009e+00,
          -1.4690e+00,  -1.2326e+00,   4.7218e-01,   4.3152e-01,  -1.2938e-01,
```

it's indicator that works.

```
pi@raspberrypi:~/DetectorPi/test $ python3 test_torch.py
tensor(2.)
tensor(1.)
tensor(1.)
w: Parameter containing:
tensor([[ -0.3264,   0.1669,  -0.0741],
        [  0.3053,   0.1813,   0.3761]], requires_grad=True)
b: Parameter containing:
tensor([ -0.4165,   0.0386], requires_grad=True)
loss: 1.3495376110076904
dL/dw: tensor([[ -0.3721,   0.0205,  -1.1976],
               [  0.3850,   0.3495,   1.0424]])
dL/db: tensor([ -0.7008,  -0.0061])
loss after 1 step optimization: 1.315846562385559
pi@raspberrypi:~/DetectorPi/test $
```

## 2. Build PyTorch from the source (DO THIS INSTRUCTION ON Raspberry Pi 4)

Ref:

+ <https://nmilosev.svbtle.com/compiling-arm-stuff-without-an-arm-board-build-pytorch-for-the-raspberry-pi> [best complete explanation]

+ <https://github.com/nmilosev/pytorch-arm-builds> [complete explanation]

+ <https://gist.github.com/akaanirban/621e63237e63bb169126b537d7a1d979>

+ <https://github.com/Ben-Faessler/Python3-Wheels> [good]

+ <https://github.com/Kashu7100/pytorch-armv7l> [finish]

Build PyTorch from the source to get **torchvision 0.7.0 [armv7l]** wheel.

+ Install package

```
$ sudo apt install libopenblas-dev libblas-dev m4 cmake cython python3-dev python3-yaml
python3-setuptools
```

```
$ sudo apt install libavcodec-dev libavformat-dev
$ sudo apt python3-wheel python3-pillow python3-numpy
```

```
$ cd pytorch
```

```
pi@raspberrypi:~/pytorch $ git clone --recursive https://github.com/pytorch/vision --branch=release/0.7
```

```
pi@raspberrypi:~/pytorch $ cd vision
```

+ Run command below:

```
pi@raspberrypi:~/pytorch/vision $ export NO_CUDA=1
pi@raspberrypi:~/pytorch/vision $ export NO_DISTRIBUTED=1
pi@raspberrypi:~/pytorch/vision $ export NO_MKLDNN=1
pi@raspberrypi:~/pytorch/vision $ export BUILD_TEST=0 # for faster builds
pi@raspberrypi:~/pytorch/vision $ export MAX_JOBS=4 # I have 4 cores
```

```
# pi@raspberrypi:~/pytorch/vision $ export NO_NNPack=1 # update July 19, this is optional, can build with NNPACK
```

```
# pi@raspberrypi:~/pytorch/vision $ export NO_QNNPACK=1 # same as above, can be omitted
```

+ Run command below and On Raspberry Pi 4 4GB with 2294MHz it take round **3 hours**

```
pi@raspberrypi:~/pytorch/vision $ python3 setup.py sdist bdist_wheel
```

**Problem:** ERROR, need install one-by-one packages.

```
e-packages/torch/include -I/home/pi/.local/lib/python3.7/site-packages/torch/include/torch/csrc/api/include -I/home/pi/.local/lib/python3.7/site-packages/torch/include/TH -I/home/pi/.local/lib/python3.7/site-packages/torch/include/THC -I/usr/include/python3.7m -c /home/pi/pytorch/vision/torchvision/csrc/cpu/video_reader/VideoReader.cpp -o build/temp.linux-armv7l-3.7/home/pi/pytorch/vision/torchvision/csrc/cpu/video_reader/VideoReader.o -std=c++14 -DTORCH_API_INCLUDE_EXTENSION_H -DTORCH_EXTENSION_NAME=video_reader -D_GLIBCXX_USE_CXX11_ABI=1
In file included from /home/pi/pytorch/vision/torchvision/csrc/cpu/decoder/memory_buffer.h:3,
                  from /home/pi/pytorch/vision/torchvision/csrc/cpu/video_reader/VideoReader.cpp:6:
/home/pi/pytorch/vision/torchvision/csrc/cpu/decoder/defs.h:18:10: fatal error: libswscale/swscale.h: No such file or directory
#include "libswscale/swscale.h"
compilation terminated.
error: command 'arm-linux-gnueabihf-gcc' failed with exit status 1
pi@raspberrypi:~/pytorch/vision $
```

**Solution:** use step 3. build pytorch from the source.

### 3. Build PyTorch from the source (WITHOUT Raspberry Pi 4)

Ref:

+ <https://nmilosev.svbtle.com/compiling-arm-stuff-without-an-arm-board-build-pytorch-for-the-raspberry-pi>

+ Cross compiling for RaspberryPi: <https://blog.kitware.com/cross-compiling-for-raspberry-pi/>

+ QEMU documentation: <https://www.qemu.org/docs/master/user/main.html>

4. Install opencv

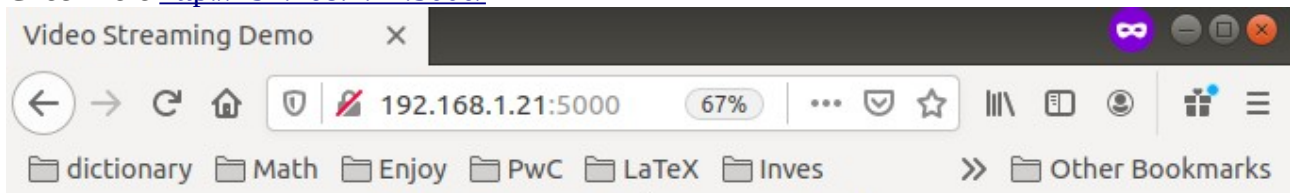
\$ pip3 install opencv-python

```
pi@raspberrypi:~/DetectorPi $ python3
Python 3.7.3 (default, Jul 25 2020, 13:03:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> print(cv2.__version__)
4.5.1
>>> 
```

Run in Pi using command below

```
pi@raspberrypi:~/VisionPi/test $ python3 test_opencv.py --source
rtsp://Anonymous:Anonymous@192.168.1.11:554
```

Check here <http://192.168.1.21:5000/>



### Video Streaming Demonstration



**Problem:** During my testing, JIT tracing didn't work for 32 bit. As JITing your model is something that comes to mind with ARM, I thought it was better to work with the (working) 64 bit version.

Ref: <http://mathinf.com/pytorch/arm64/> and <https://github.com/pytorch/pytorch/issues/27040>

## 2. Section 2

### 0. Tools

#### 0.1. Hardware

- + **RaspberryPi 4 Model B** Rev 1 (RPi), RAM: **4 GB** and Processor 4-core @ **1.5 GHz**
- + microSD Card **64 GB**
- + 5M USB Retractable Clip 120 Degrees WebCam Web Wide-angle Camera Laptop U7 Mini or Raspi Camera

#### 0.2. Tested Software

- + Ubuntu 20.10 [**aarch64**] **64 bit**, install on RPi
- + PyTorch [**aarch64**]: torch 1.6.0 and torchvision 0.7.0
- + Python min. ver. 3.6

```
mhy@pi:~$ neofetch

      .-/+00SSSS00+/- .
      `:+SSSSSSSSSSSSSSSS+:`
      -+SSSSSSSSSSSSSSSSyySSSS+-
      .oSSSSSSSSSSSSSSSSdMMMMNySSSSo.
      /SSSSSSSSSSShdmmNNmyNMMMMhSSSSS/
      +SSSSSSSSShmydMMMMMMMMNdddySSSSSSS+
      /SSSSSSSSShNMMMyhhyyyhNMMMMNhSSSSSSS/
      .SSSSSSSSdMMMNhSSSSSSSSShNMMMdSSSSSSS.
      +SSSSShhyNMMNySSSSSSSSSSSyNMMMySSSSSSS+
      oSSyNMMMNyMMhSSSSSSSSSSSSShmmhSSSSSSSo
      oSSyNMMMNyMMhSSSSSSSSSSSSShmmhSSSSSSSo
      +SSSSShhyNMMNySSSSSSSSSSSyNMMMySSSSSSS+
      .SSSSSSSSdMMMNhSSSSSSSSShNMMMdSSSSSSS.
      /SSSSSSSShNMMMyhhyyyhNMMMMNhSSSSSSS/
      +SSSSSSSSdmydMMMMMMMMNdddySSSSSSS+
      /SSSSSSSSShdmmNNmyNMMMMhSSSSS/
      .oSSSSSSSSSSSSSSSSdMMMMNySSSSo.
      -+SSSSSSSSSSSSSSSSyySSSS+-
      `:+SSSSSSSSSSSSSSSS+:`
      .-/+00SSSS00+/- .

mhy@pi
-----
OS: Ubuntu 20.10 aarch64
Host: Raspberry Pi 4 Model B R
Kernel: 5.8.0-1011-raspi
Uptime: 3 mins
Packages: 1807 (dpkg), 6 (snap)
Shell: bash 5.0.17
Terminal: /dev/pts/0
CPU: BCM2835 (4) @ 1.500GHz
Memory: 279MiB / 3741MiB
```

```
$ sudo apt install python3-numpy python3-wheel python3-setuptools python3-future python3-yaml
python3-six python3-requests python3-pip python3-pillow
```

+ Download file that build with a 8GB Raspberry Pi 4 (and without needing swap) starting from the minimal Raspberry Pi OS (Buster), python 3.7.

torch-1.6.0a0+b31f58d-cp37-cp37m-linux\_aarch64.whl: [http://mathinf.com/pytorch/arm64/torch-1.6.0a0+b31f58d-cp37-cp37m-linux\\_aarch64.whl](http://mathinf.com/pytorch/arm64/torch-1.6.0a0+b31f58d-cp37-cp37m-linux_aarch64.whl)

torchvision-0.7.0a0+78ed10c-cp37-cp37m-linux\_aarch64.whl:  
[http://mathinf.com/pytorch/arm64/torchvision-0.7.0a0+78ed10c-cp37-cp37m-linux\\_aarch64.whl](http://mathinf.com/pytorch/arm64/torchvision-0.7.0a0+78ed10c-cp37-cp37m-linux_aarch64.whl)

+ Source: <https://github.com/ljk53/pytorch-rpi>



**torch-1.6.0a0+b31f58d-cp38-cp38-linux\_aarch64.whl** [https://github.com/ljk53/pytorch-rpi/blob/master/torch-1.6.0a0%2Bb31f58d-cp38-cp38-linux\\_aarch64.whl](https://github.com/ljk53/pytorch-rpi/blob/master/torch-1.6.0a0%2Bb31f58d-cp38-cp38-linux_aarch64.whl)  
**torchvision 0.7.0:**

\$ scp torch-1.6.0a0+b31f58d-cp38-cp38-linux\_aarch64.whl  
[mhy@192.168.1.21](mailto:mhy@192.168.1.21):/home/mhy/Downloads

mhy@pi:~/Downloads\$ pip3 install torch-1.6.0a0+b31f58d-cp38-cp38-linux\_aarch64.whl

+ Install packages

\$ sudo apt install build-essential make cmake git python3-pip libatlas-base-dev

\$ sudo apt install libssl-dev

\$ sudo apt install libopenblas-dev libblas-dev m4 python3-yaml

\$ sudo apt install libomp-dev

+ Make WASP to 2048 MB

\$ free -h

```
mhy@pi:~$ free -h
```

	total	used	free	shared	buff/cache	available
Mem:	3,7Gi	273Mi	2,3Gi	4,0Mi	1,1Gi	3,3Gi
Swap:	1,0Gi	0B	1,0Gi			

\$ sudo swapoff -a

\$ sudo dd if=/dev/zero of=/swapfile bs=1M count=2048

```
mhy@pi:~$ sudo dd if=/dev/zero of=/swapfile bs=1M count=2048
2048+0 records in
2048+0 records out
2147483648 bytes (2,1 GB, 2,0 GiB) copied, 126,679 s, 17,0 MB/s
```

\$ sudo mkswap /swapfile

\$ sudo swapon /swapfile

\$ free -h

```
mhy@pi:~$ free -h
```

	total	used	free	shared	buff/cache	available
Mem:	3,7Gi	275Mi	216Mi	4,0Mi	3,2Gi	3,3Gi
Swap:	2,0Gi	0B	2,0Gi			

+ Test PyTorch:

```
mhy@pi:~$ python3
Python 3.8.6 (default, Sep 25 2020, 09:36:53)
[GCC 10.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> exit()
```

```

mhy@pi:~/DetectorPi/test$ python3 test_torch.py
tensor(2.)
tensor(1.)
tensor(1.)
w: Parameter containing:
tensor([[ 0.1052,  0.4078, -0.4026],
        [ 0.4273,  0.3837, -0.0545]], requires_grad=True)
b: Parameter containing:
tensor([ 0.5224, -0.3139], requires_grad=True)
loss: 0.8868435025215149
dL/dw: tensor([[ -0.3426,  0.1850, -0.0676],
               [ -0.1126,  0.5372,  0.1734]])
dL/db: tensor([0.6858, 0.1581])
loss after 1 step optimization: 0.8770723342895508
mhy@pi:~/DetectorPi/test$

```

+ Install packages

\$ pip3 install Flask

\$ pip3 install waitress

+ Install opencv

\$ pip3 install opencv-python

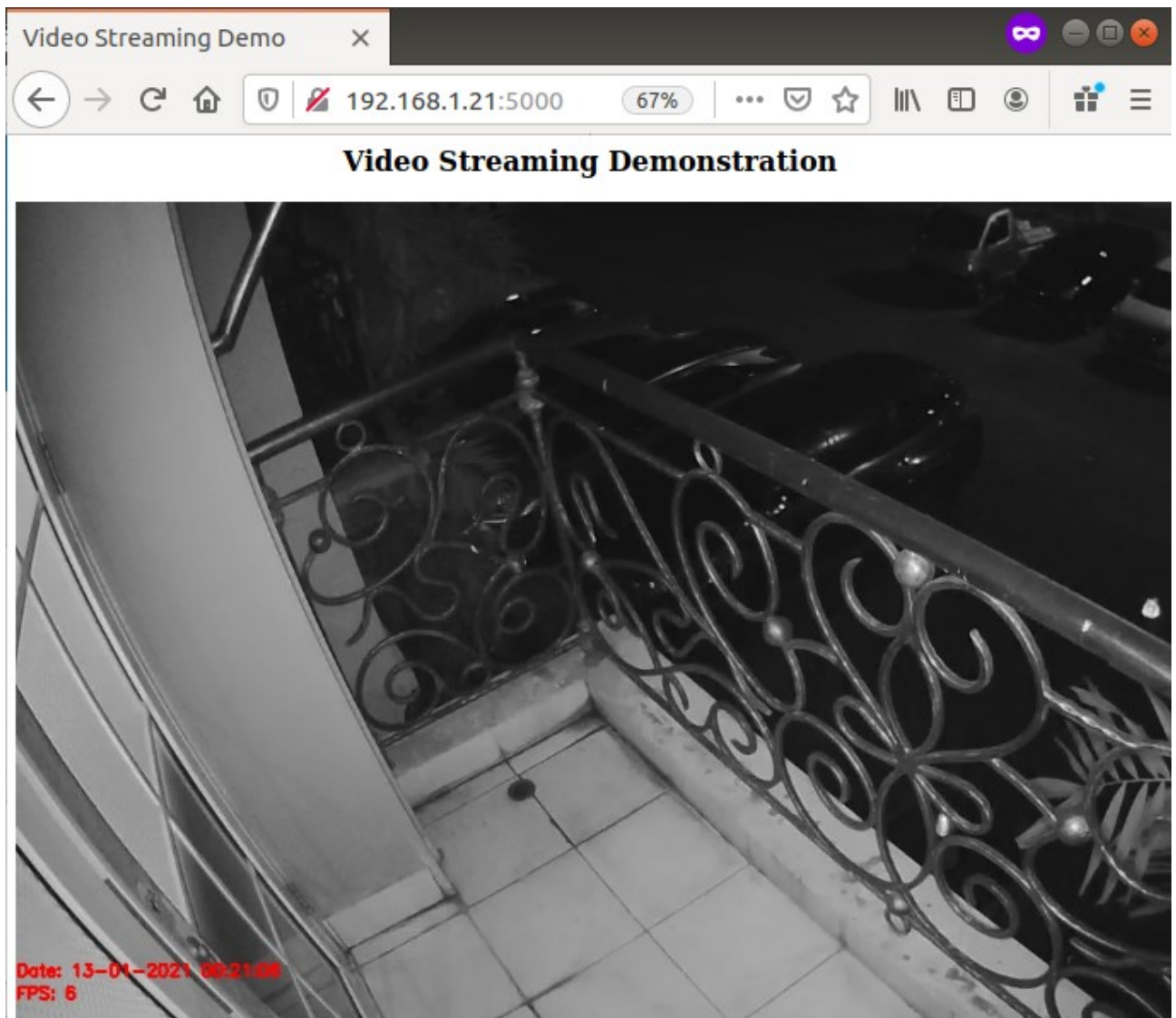
```

mhy@pi:~$ python3
Python 3.8.6 (default, Sep 25 2020, 09:36:53)
[GCC 10.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> print(cv2.__version__)
4.5.1
>>>

```

mhy@pi:~/VisionPi/test\$ python3 test\_opencv.py --source

rtsp://Anonymous:[Anonymous@192.168.1.11](mailto:Anonymous@192.168.1.11):554



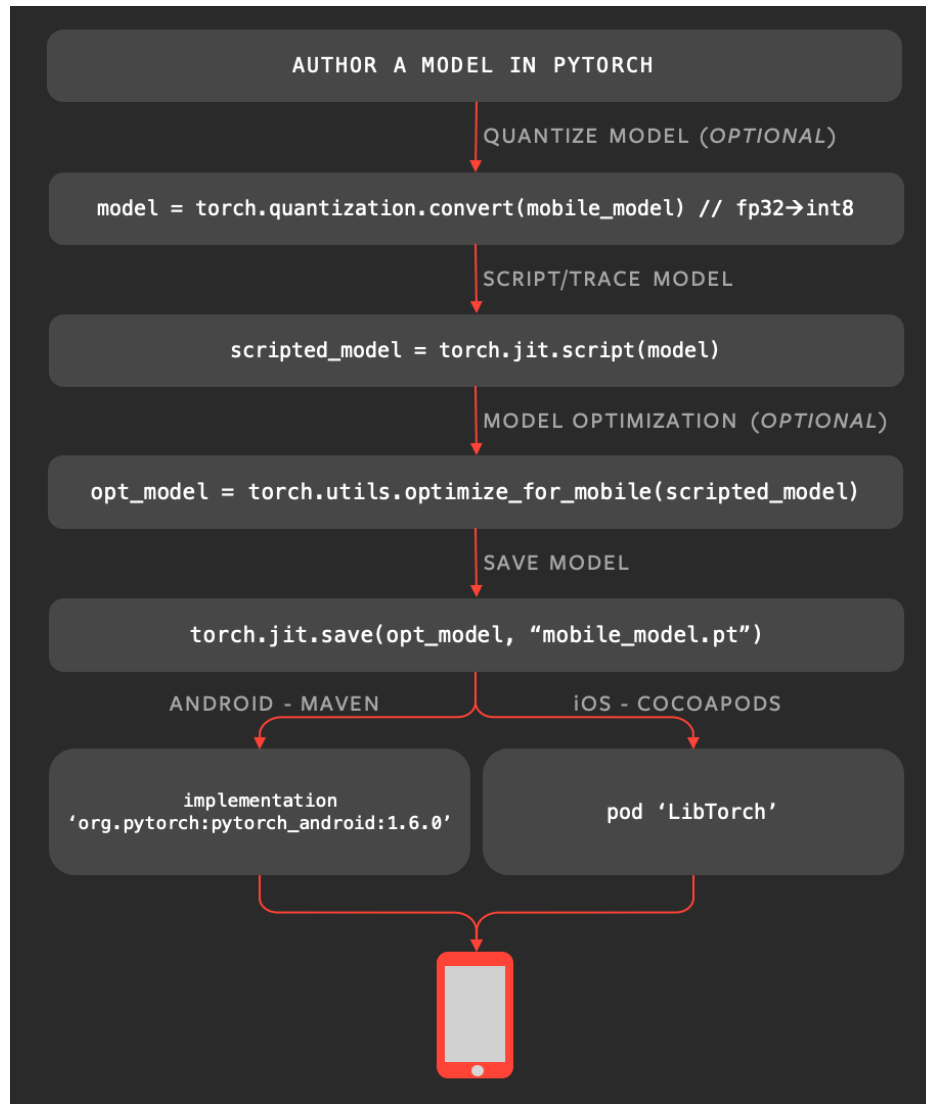


# PyTorch Mobile

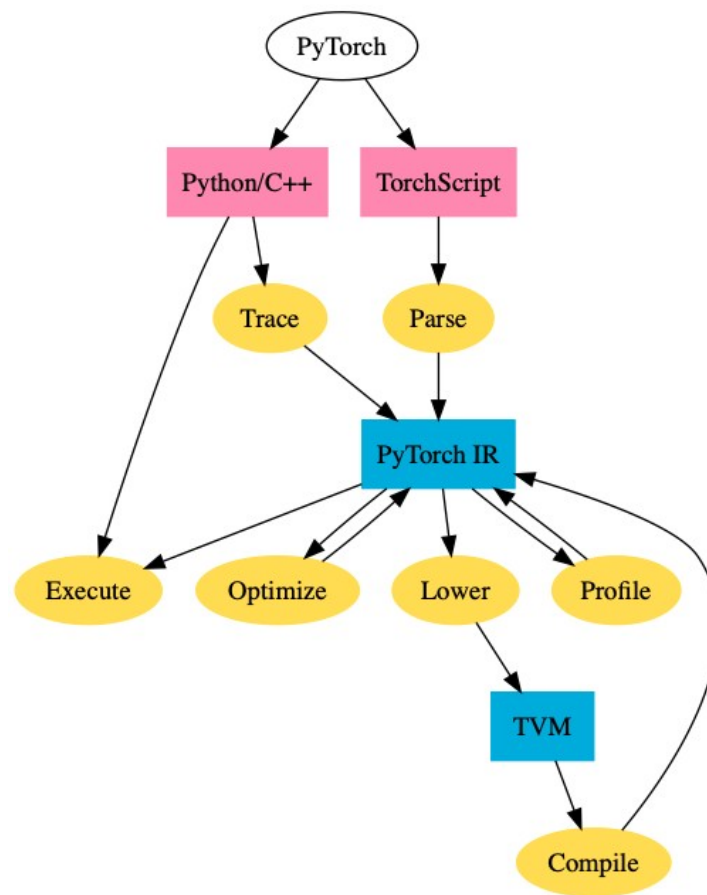
## 0. Referensi

+ Official website: <https://pytorch.org/mobile/home/>

Masih versi beta tetapi juga sudah digunakan untuk tahap produksi secara luas.



URL: <https://github.com/pytorch/tvm>



- + <https://github.com/pytorch/android-demo-app>
- + <https://github.com/cedrickchee/pytorch-mobile-kit>

# PyTorch Embedded System

## 0. Referensi

+ FedML: A Research Library and Benchmark for Federated Machine Learning: <https://github.com/FedML-AI>

+ FedML-IoT: Federated Learning on IoT Devices (supported by FedML framework) : <https://github.com/FedML-AI/FedML-IoT>

+ FastDepth: Fast Monocular Depth Estimation on Embedded Systems: <https://github.com/tau-adl/FastDepth>

## 1. Compiling PyTorch

+ <https://nmilosev.svbtle.com/compiling-arm-stuff-without-an-arm-board-build-pytorch-for-the-raspberry-pi>

# NanoDet

Last update: Sep 5, 2021

Ref: <https://github.com/RangiLyu/nanodet>

## >> Train Custom Dataset

Ref: <https://github.com/RangiLyu/nanodet#how-to-train>

+1. Prepare dataset

If your dataset annotations are pascal voc xml format, refer to **config/nanodet\_custom\_xml\_dataset.yml**.

+2. Prepare config file

Detail explanation: [https://github.com/RangiLyu/nanodet/blob/main/docs/config\\_file\\_detail.md](https://github.com/RangiLyu/nanodet/blob/main/docs/config_file_detail.md)

- Copy and modify an **example yml config file** in config/ folder.
  - Change **save\_path** to where you want to save model.
  - Change **num\_classes** in models->arch->head.
  - Change **image path** and **annotation path** in both data->train and data->val.
  - Set **gpu ids**, **num workers** and **batch size** in device to fit your device.
- gpu\_ids: CUDA device id. For multi-gpu training, set [0, 1, 2...].  
workers\_per\_gpu: how many dataloader processes for each gpu.  
batchsize\_per\_gpu: amount of images in one batch for each gpu.

Issue1: <https://stackoverflow.com/questions/53998282/how-does-the-number-of-workers-parameter-in-pytorch-dataloader-actually-work>

So when **num\_workers=2** you have at most 2 workers simultaneously putting data into **RAM**.  
So setting **workers to number of CPU cores** is a good rule of thumb, nothing more.

- Set **total\_epochs**, **lr** and **lr\_schedule** according to your dataset and batchsize.

+3. Start training

NanoDet is now using **pytorch lightning** for training.

## >> Architecture

