



Tina V853 NPU Yolact 模型部署实战

版本号: 1.0

发布日期: 2021.07.21

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.07.21	PDC	NPU Yolact 模型部署实战

目 录

1 前言	1
1.1 读者对象	1
1.2 约定	1
1.2.1 符号约定	1
2 正文	2
2.1 NPU 开发简介	2
2.2 开发流程	2
2.3 获取 YOLACT 原始模型	3
2.4 模型部署工作目录结构	3
2.5 导入模型	3
2.6 创建 YML 文件	3
2.7 量化	4
2.8 预推理	5
2.9 导出代码和 NBG 文件	5
2.10 模型算力测量	6
2.11 后处理验证	7
2.12 结束	9

插图

2-1 npu_1.png	2
2-2 npu_workspace	3
2-3 npu_import	3
2-4 npu_yml	4
2-5 npu_scale	4
2-6 npu_quantilize	4
2-7 npu_inf	5
2-8 npu_export	6
2-9 npu_nbg	6
2-10 npu_measure	6
2-11 npu_gerit	7
2-12 npu_post	7
2-13 npu_postprocess	7
2-14 npu_tensor	7
2-15 npu_exepost	8
2-16 npu_yolact	8
2-17 npu_yolact_ok	9

1 前言

1.1 读者对象

本文档（本指南）主要适用于以下人员：

- 技术支持工程师
- 软件开发工程师
- AI 应用案客户

1.2 约定

1.2.1 符号约定

本文中可能出现的符号如下：



警告

警告



技巧

1. 技巧
2. 小常识



说明

说明

2 正文

2.1 NPU 开发简介

- 支持 int8/uint8/int16 量化精度，运算性能可达 1TOPS.
- 相较于 GPU 作为 AI 运算单元的大型芯片方案，功耗不到 GPU 所需要的 1%.
- 可直接导入 Caffe, TensorFlow, Onnx, TFLite, Keras, Darknet, pyTorch 等模型格式.
- 提供 AI 开发工具：支持模型快速转换、支持开发板端侧转换 API，支持 TensorFlow, TF Lite, Caffe, ONNX, Darknet, pyTorch 等模型.
- 提供 AI 应用开发接口：提供 NPU 跨平台 API.

2.2 开发流程

NPU 开发完整的流程如下图所示：

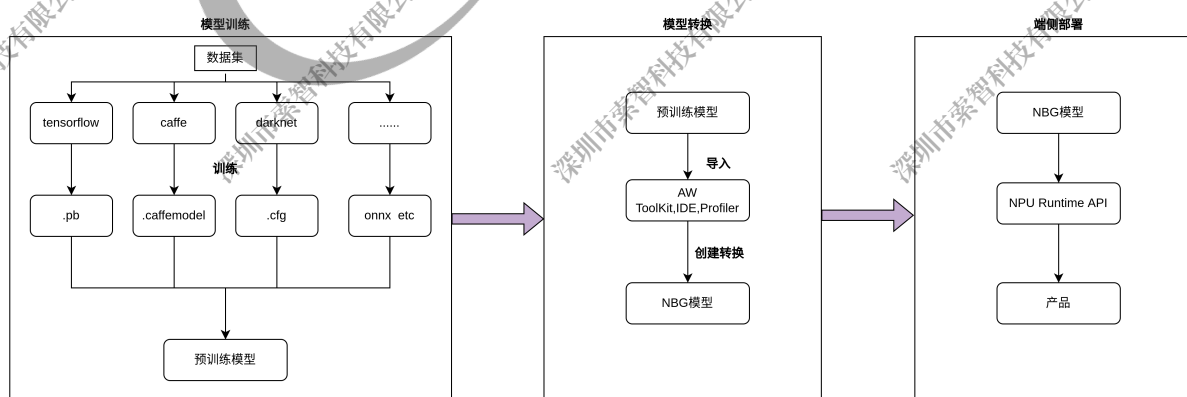


图 2-1: npu_1.png

2.3 获取 YOLACT 原始模型

YOLACT 模型获取方式有很多种，可以基于项目 <https://github.com/dbolya/yolact> 产生 onnx 格式的 yolact 模型，本文档假设你已经产生了 yolact.onnx 模型

2.4 模型部署工作目录结构

yolact-sim.onnx 有 120M，还是蛮大的。

```
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220210$ tree
.
├── data
│   ├── dog.jpg
│   ├── dataset.txt
│   └── yolact-sim.onnx
└── 1 directory, 3 files
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220210$ cat dataset.txt
./data/dog.jpg
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220210$
```

图 2-2: npu_workspace

2.5 导入模型

```
pegasus import onnx --model yolact-sim.onnx --output-model yolact-sim.json --output-data yolact-sim.data
```

导入模型的目的是将开放模型转换为符合 VIP 模型网络描述文件 (.json) 和权重文件 (.data)

```
data dataset.txt yolact-sim.data yolact-sim.json yolact-sim.onnx
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220210$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    yolact-sim.data
    yolact-sim.json
nothing added to commit but untracked files present (use "git add" to track)
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220210$
```

图 2-3: npu import

2.6 创建 YML 文件

YML 文件对网络的输入和输出进行描述，比如输入图像的形状，归一化系数 (均值，零点)，图像格式，输出 tensor 的输出格式，后处理方式等等，命令如下：

```
pegasus generate inputmeta --model yolact-sim.json --input-meta-output yolact-sim-inputmeta.yml
```

```
pegasus generate postprocess-file --model yolact-sim.json --postprocess-file-output yolact-sim-postprocess-file.yml
```

```
(vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$ git status .
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    yolact-sim-inputmeta.yml
    yolact-sim-postprocess-file.yml

nothing added to commit but untracked files present (use "git add" to track)
(vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
(vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
(vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
(vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
```

图 2-4: npu_yml

修改 input meta 文件中的的 scale 参数为 0.0039(1/256)，和 yolov3 的改法完全一致。

```
1 # yolact-sim-inputmeta.yml 2: 1 2
2 [[yolact-sim-inputmeta.yml]](2: 1 2)
3
4 # This file disallow TAB!!!
5 # "category" allowed values: "image, frequency, undefined"
6 # "database" allowed types: "TEXT, NPY, H5PS, SQLITE, LMDB, GENERATOR"
7 # "tensor_name" only support in h5ps database
8 # "preproc_type" allowed types: "IMAGE_RGB, IMAGE_ROBUST_PLANAR, IMAGE_I420, IMAGE_NV12, IMAGE_YUV444, IMAGE_GRAY, IMAGE_BGRA, TENSOR"
9
10 input_meta:
11   databases:
12     path: dataset.txt
13     type: TEXT
14   path:
15     id: input_1_210
16     category: image
17     dtype: float32
18     sparse: false
19     tensor_name:
20     layout: mchw
21     shape:
22       1
23       3
24       550
25       550
26   fitting: scale
27   preprocess:
28     reverse_channel: true
29     mean:
30       0
31       0
32       0
33     scale: 0.0039
34     add_preproc_node: false
35     preproc_type: IMAGE_RGB
36     preproc_perm:
37       0
38       1
39       2
40       3
41   redirect_to_output: false
```

图 2-5: npu_scale

2.7 量化

量化命令:

```
pegasus quantize --model yolact-sim.json --model-data yolact-sim.data --batch-size 1 --device CPU
--with-input-meta yolact-sim-inputmeta.yml --rebuild --model-quantize yolact-sim.quantize --
quantizer asymmetric_affine --qtype uint8
```

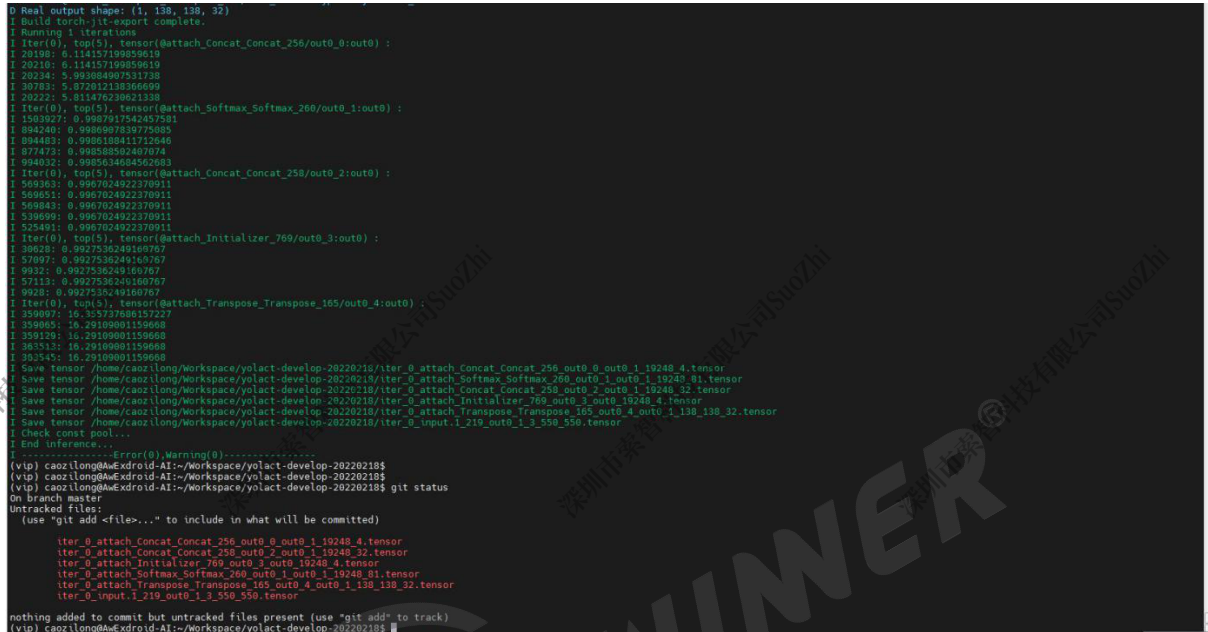
命令执行后，创建了量化表文件

```
1 End quantization...
2 Dump net quantize tensor table to yolact-sim.quantize
3 Save net to yolact-sim.data
4 -----Error(s)/Warning(s)-----
5 (vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
6 (vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$ git status .
7 On branch master
8 Untracked files:
9   (use "git add <file>..." to include in what will be committed)
10
11     yolact-sim.quantize
12
13 nothing added to commit but untracked files present (use "git add" to track)
14 (vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
15 (vip) caozilong@wexdroid-AI:~/Workspace/yolact-develop-20220210$
```

图 2-6: npu_quantize

2.8 预推理

```
pegasus inference --model yolact-sim.json --model-data yolact-sim.data --batch-size 1 --dtype  
quantized --model-quantize yolact-sim.quantilize --device CPU --with-input-meta yolact-sim-  
inputmeta.yml --postprocess-file yolact-sim-postprocess-file.yml
```



```
0 Real output shape: (1, 138, 138, 32)
1 Build torch-jit-export complete.
2 Running 1 iterations
3 Iter(0), top(5), tensor(@attach_Concat_Concat_256/out0_0:out0) :
4 20108: 6.114157198859619
5 20210: 6.114157198859619
6 20234: 5.99884907531738
7 30783: 5.072012138366699
8 20222: 5.81495239621138
9 Iter(0), top(5), tensor(@attach_Softmax_Softmax_260/out0_1:out0) :
10 1503827: 0.9987917542457581
11 894240: 0.9986987835775085
12 894483: 0.9986188411712646
13 877473: 0.998588502487074
14 894032: 0.998434645452683
15 Iter(0), top(5), tensor(@attach_Concat_Concat_258/out0_2:out0) :
16 568363: 0.9967024922370911
17 568651: 0.9967024922370911
18 568843: 0.9967024922370911
19 539699: 0.9967024922370911
20 528481: 0.9967024922370911
21 Iter(0), top(5), tensor(@attach_Initializer_769/out0_3:out0) :
22 39628: 0.9927536245160767
23 57097: 0.9927536245160767
24 9932: 0.9927536245160767
25 57113: 0.9927536245160767
26 9928: 0.9927536245160767
27 Iter(0), top(5), tensor(@attach_Transpose_Transpose_165/out0_4:out0) :
28 359897: 16.353737686157227
29 359866: 16.29109001159668
30 359120: 16.29109001159668
31 383513: 16.29109001159668
32 38345: 16.29109001159668
33 Save tensor /home/caozilong/Workspace/yolact-develop-20220210/iter_0_attach_Concat_Concat_256_out0_0_out0_1_19248_4.tensor
34 Save tensor /home/caozilong/Workspace/yolact-develop-20220210/iter_0_attach_Softmax_Softmax_260_out0_1_out0_1_19248_81.tensor
35 Save tensor /home/caozilong/Workspace/yolact-develop-20220210/iter_0_attach_Concat_Concat_258_out0_2_out0_1_19248_32.tensor
36 Save tensor /home/caozilong/Workspace/yolact-develop-20220210/iter_0_attach_Initializer_769_out0_3_out0_1_19248_4.tensor
37 Save tensor /home/caozilong/Workspace/yolact-develop-20220210/iter_0_attach_Transpose_Transpose_165_out0_4_out0_1_138_138_32.tensor
38 Save tensor /home/caozilong/Workspace/yolact-develop-20220210/iter_0_input_1_192_out0_1_9_550_550.tensor
39 Check const pool...
40 End inference...
41 (vip) caozilong@Aixendroid-AI:~/Workspace/yolact-develop-20220210$
42 (vip) caozilong@Aixendroid-AI:~/Workspace/yolact-develop-20220210$
43 (vip) caozilong@Aixendroid-AI:~/Workspace/yolact-develop-20220210$ git status
44 On branch master
45 Untracked files:
46   (use "git add <file>..." to include in what will be committed)
47
48   iter_0_attach_Concat_Concat_256_out0_0_out0_1_19248_4.tensor
49   iter_0_attach_Concat_Concat_258_out0_2_out0_1_19248_32.tensor
50   iter_0_attach_Initializer_769_out0_3_out0_1_19248_4.tensor
51   iter_0_attach_Softmax_Softmax_260_out0_1_out0_1_19248_81.tensor
52   iter_0_attach_Transpose_Transpose_165_out0_4_out0_1_138_138_32.tensor
53   iter_0_input_1_192_out0_1_9_550_550.tensor
54
55 nothing added to commit but untracked files present (use "git add" to track)
56 (vip) caozilong@Aixendroid-AI:~/Workspace/yolact-develop-20220210$
```

图 2-7: npu_inf

2.9 导出代码和 NBG 文件

```
pegasus export ovxlib --model yolact-sim.json --model-data yolact-sim.data --dtype quantized --  
model-quantize yolact-sim.quantilize --batch-size 1 --save-fused-graph --target-ide-project '  
linux64' --with-input-meta yolact-sim-inputmeta.yml --postprocess-file yolact-sim-postprocess-file  
.yml --output-path ovxlib/yolact/yolact --pack-nbg-unify --optimize "VIP9000PICO_PID0XEE" --viv-  
sdk ${VIV_SDK}
```

```
pegasus export ovxlib --model yolact-sim.json --model-data yolact-sim.data --dtype quantized --  
model-quantize yolact-sim.quantilize --batch-size 1 --save-fused-graph --target-ide-project '  
linux64' --with-input-meta yolact-sim-inputmeta.yml --postprocess-file yolact-sim-postprocess-file  
.yml --output-path ovxlib/yolact/yolact --pack-nbg-viplite --optimize "VIP9000PICO_PID0XEE" --viv-  
sdk ${VIV_SDK}
```

```
vdrtproxy -l/home/caozilong/VeriSilicon/VivanteIDE5.5.0/cmdtools/vsimulator/./common/lib/ -lvdrtproxy /home/caozilong/VeriSilicon/VivanteIDE5.5.0/cmdtools/vsimulator/./common/lib/libpeg.a -o gen_nbg

Create Neural Network: 204ms or 204579us
Verify:
Verify Graph: 60371ms or 60371600us
Start run graph 1 times.
Run the 1 time: 6.00ms or 6004.00us
vxProcessGraph execution time:
Total 6.00ms or 6936.00us
Average 6.04ms or 6936.00us
--- Top5 ---
0: 0.000000
1: 0.000000
2: 0.000000
3: 0.000000
4: 0.000000
1 Export ovxlib case with version 1.1.34
1 Dump nbg input meta to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/nbg_meta.json
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/vnn_post_process.c
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/vnn_pre_process.h
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/vnn_pre_process.c
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/vnn_pre_process.h
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/main.c
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/yolact_vcpro
1 Save vx network source file to /home/caozilong/Workspace/yolact-develop-20220218/ovxlib/yolact_nbg_viplite/makefile.linux
1 Dump net to ovxlib/yolact/yolact_fused.json
1 End exporting ovxlib case
1 -----Error(0),Warning(0)-----
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$ tree -L 2
.
├── data
│   ├── dog.jpg
│   └── dataset.txt
├── iter_0_attach_Concat_Concat_256_out0_0_out0_1_192x48_4.tensor
├── iter_0_attach_Concat_Concat_256_out0_2_out0_1_192x48_32.tensor
├── iter_0_attach_Initializer_769_out0_3_out0_192x48_4.tensor
├── iter_0_attach_Softmax_Softmax_260_out0_1_out0_1_192x48_32.tensor
├── iter_0_attach_Transpose_Transpose_165_out0_4_out0_1_138_138_32.tensor
├── iter_0_input_1_219_out0_1_3_550_550.tensor
├── ovxlib
│   ├── yolact
│   │   ├── yolact_nbg_unify
│   │   ├── yolact_nbg_unify_ovx
│   │   └── yolact_nbg_viplite
│   ├── yolact-sim.data
│   ├── yolact-sim-inputmeta.yml
│   ├── yolact-sim.json
│   ├── yolact-sim.ovmx
│   ├── yolact-sim-postprocess-file.yml
│   └── yolact-sim.quantize
└── 0 directories, 14 files
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
```

图 2-8: npu_export

生成的 NBG 文件, 可以部署到端侧:

```
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$ sha512sum ovxlib/yolact/*-nb
4e1876730b1883f4a5ec062b9f2b21da532614f91da80a245850b09a626fc3704d2b30193b99f41bc0ff49677c2e37e2c8970f028da4182a812d106a35b61 ovxlib/yolact_nbg_unify/network_binary.nb
4e1876730b1883f4a5ec062b9f2b21da532614f91da80a245850b09a626fc3704d2b30193b99f41bc0ff49677c2e37e2c8970f028da4182a812d106a35b61 ovxlib/yolact_nbg_unify_ovx/network_binary.nb
4e1876730b1883f4a5ec062b9f2b21da532614f91da80a245850b09a626fc3704d2b30193b99f41bc0ff49677c2e37e2c8970f028da4182a812d106a35b61 ovxlib/yolact_nbg_viplite/network_binary.nb
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
```

图 2-9: npu_nbg

2.10 模型算力测量

pegasus measure --model yolact-sim.json

```
Process Conv_Conv_161_22:
0 Acuity output shape(convolution): (1 138 138 256)
0 Tensor @Conv_Conv_161_22:out0 type: float32
0 Process ReLU_ReLU_162_30 ...
0 Acuity output shape(relu): (1 138 138 256)
0 Tensor @ReLU_ReLU_162_30:out0 type: float32
0 Process Conv_Conv_163_22 ...
0 Acuity output shape(convolution): (1 138 138 32)
0 Tensor @Conv_Conv_163_22:out0 type: float32
0 Process ReLU_ReLU_164_10 ...
0 Acuity output shape(relu): (1 138 138 32)
0 Tensor @ReLU_ReLU_164_10:out0 type: float32
0 Process attach_Transpose_Transpose_165/out0_4 ...
0 Acuity output shape(output): (1 138 138 32)
0 Tensor @attach_Transpose_Transpose_165/out0_4:out0 type: float32
1 Build torch-rt-export complete.
1 Dump analysis to /home/caozilong/Workspace/yolact-develop-20220218/analysis.json
1 sum_macc:59.280 sum_param:34.98M sum_activation:227.32M
1 End measuring...
1 -----Error(0),Warning(0)-----
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
(vip) caozilong@wExdroid-AI:~/Workspace/yolact-develop-20220218$
```

图 2-10: npu_measure

对比 yolov3 的 32.99G MACC 算力和 yolov3-tiny 的 2.79G macc 的算力需求, YOLACT 明显对算力的需求大很多. 题外话, 1TOPS=1000GOPS, 对于 yolov3-tiny 来说, 我们以 3G 的算力需求来计算, 就是 0.003T, 在 500M 频率下, NPU 的理论算力是 0.5T, 所以, YOLOV3 的帧率为: 0.5T/0.003T=166 帧. 这个值和仿真值是比较接近的.

2.11 后处理验证

yolact 的后处理 C 代码在如下位置：

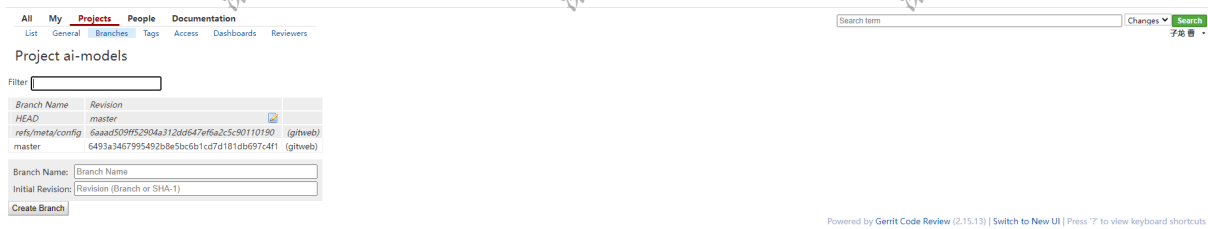


图 2-11: npu_gerit

编译后处理模型：

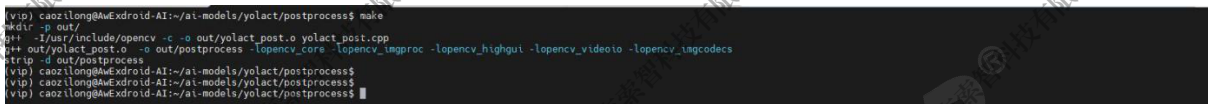


图 2-12: npu_post

out 目录生成了 yolact 的后处理程序：



图 2-13: npu_postprocess

yolact 模型有五个输出层,inference 阶段生成了 5 个 output tensor.

iter_0_attach_Concat_Concat_256_out0_0_out0_1_19248_4.tensor	2022/2/18 16:23	TENSOR 文件	1.444 KB
iter_0_attach_Concat_Concat_256_out0_2_out0_1_19248_32.tensor	2022/2/18 16:23	TENSOR 文件	11.570 KB
iter_0_attach_initializer_789_out0_3_out0_1_19248_4.tensor	2022/2/18 16:23	TENSOR 文件	880 KB
iter_0_attach_Softmax_Softmax_260_out0_1_out0_1_19248_81.tensor	2022/2/18 16:23	TENSOR 文件	33.932 KB
iter_0_attach_Transpose_Transpose_165_out0_4_out0_1_138_138_32.tensor	2022/2/18 16:23	TENSOR 文件	9.622 KB
iter_0_input_1_219_out0_1_3_550_550.tensor	2022/2/18 16:23	TENSOR 文件	16.965 KB

图 2-14: npu_tensor

tensor之间的对应关系,0是location, 1是confidence, 2是mask, 3暂时无用, 4是maskmaps.

output0_4_19248_1.dat <----> iter_0_attach_Concat_Concat_256_out0_0_out0_1_19248_4.tensor

output1_81_19248_1.dat<----> iter_0_attach_Softmax_Softmax_260_out0_1_out0_1_19248_81.tensor

```
output2_32_19248_1.dat<--->iter_0_attach_Concat_Concat_258_out0_2_out0_1_19248_32.tensor  
output3_4_19248.dat <--->iter_0_attach_Initializer_769_out0_3_out0_19248_4.tensor  
output4_32_138_138_1.dat<--->iter_0_attach_Transpose_Transpose_165_out0_4_out0_1_138_138_32.tensor
```

执行后处理

```
(vip) caozilong@AwExdroid-AI:~/ai-models/yolact/postprocess$ ./out/postprocess dog.jpg  
time : 0.035 Sec  
2 = 0.96443 at 99.48 111.53 478.22 x 329.83  
17 = 0.76052 at 128.47 212.00 196.70 x 339.42  
17 = 0.70652 at 130.96 150.66 341.01 x 358.92  
8 = 0.52484 at 472.83 76.51 207.34 x 92.95  
3 = 0.52384 at 472.83 75.79 207.34 x 94.07  
(vip) caozilong@AwExdroid-AI:~/ai-models/yolact/postprocess$
```

图 2-15: npu_exepost

后处理结果：



图 2-16: npu_yolact

图中识别除了两只 dog, 有两个狗的信息, 其他都能很好的输出结果, dog 70.7% 不应该出现的, 大概率是量化导致输出结果精度精度降低。

修改归一化参数后重新部署, 得到的结果如下, 可以看到, 之前的问题消失:

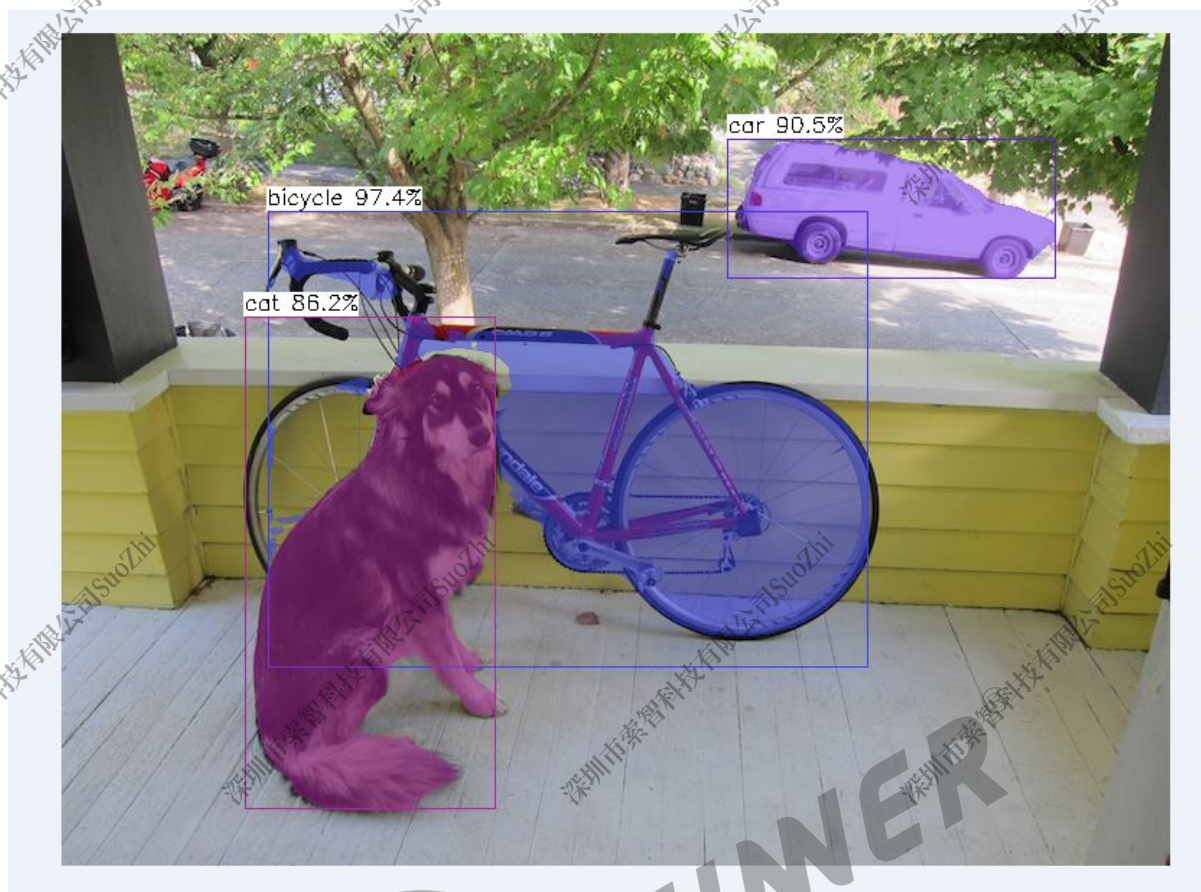


图 2-17: npu_yoloact_ok

至此，yoloact 模型导入完成。

2.12 结束

著作权声明

版权所有 © 2022 珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明

、 **全志科技** （不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本文档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。