设备端深度学习落地实践

何亮亮

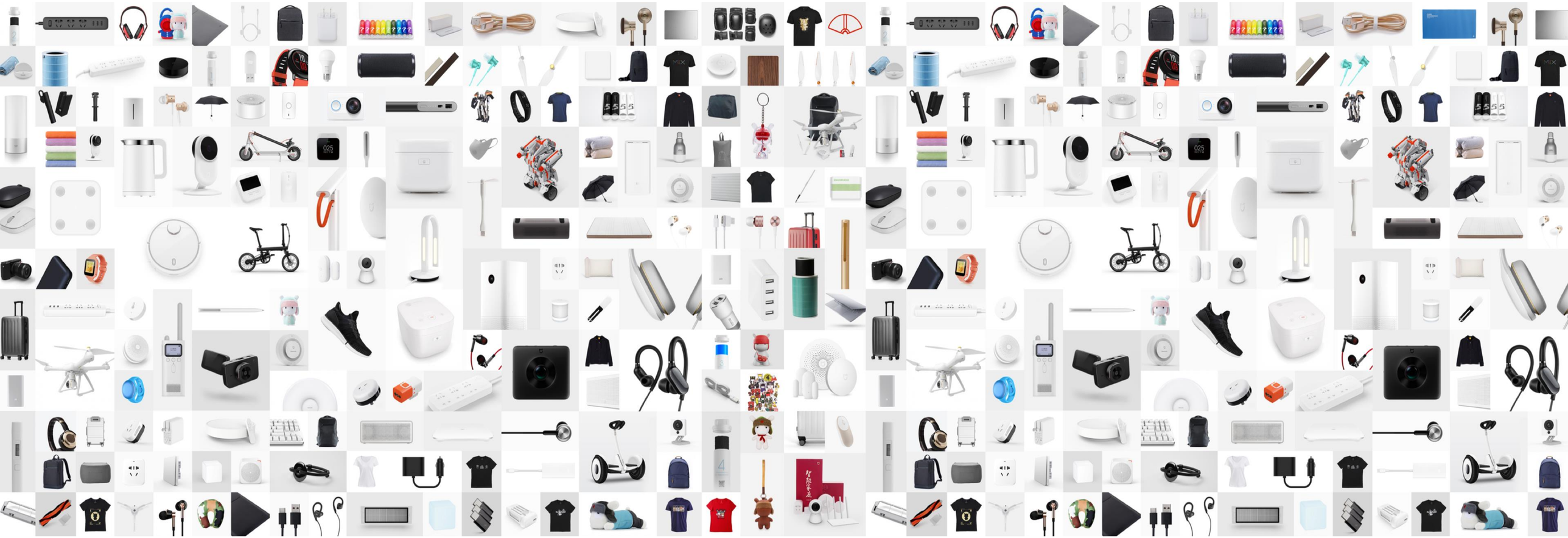
小米人工智能与云平台

目录

- 设备端深度学习推理现状
- MACE的设计与架构
- MACE的使用简介
- MACE优化技术
- MACE的应用案例

设备端深度学习推理现状

1亿+设备



丰富的落地场景

云 -> 端

体验

翻译

拍照

成本

图像风格化

超分辨率

隐私

语音

监控

开发部署流程

Xiaomi
Cloud ML

模型文件

MACE

训练平台

设备端部署

碎片化现状

TensorFlow Lite

Caffe

通用处理器CPU

高通SNPE

MTK NeuroPilot

通用异构计算设备GPU, DSP

各种专有框架

专有加速器NPU

QUALCOMM

MEDIATEK

Amlogic

SPREADTRUM 展讯通信

Rockchip 瑞芯微电子

Movidius an intel company

CEVA

VeriSilicon

HISILICON

Cambricon

支持异构计算的移动端
深度学习计算引擎

MACE设计目标

多框架支持

异构硬件

推理速度

内存占用

用户体验

丰富算子支持

多芯片平台

启动速度

功耗

模型安全

开源框架对比

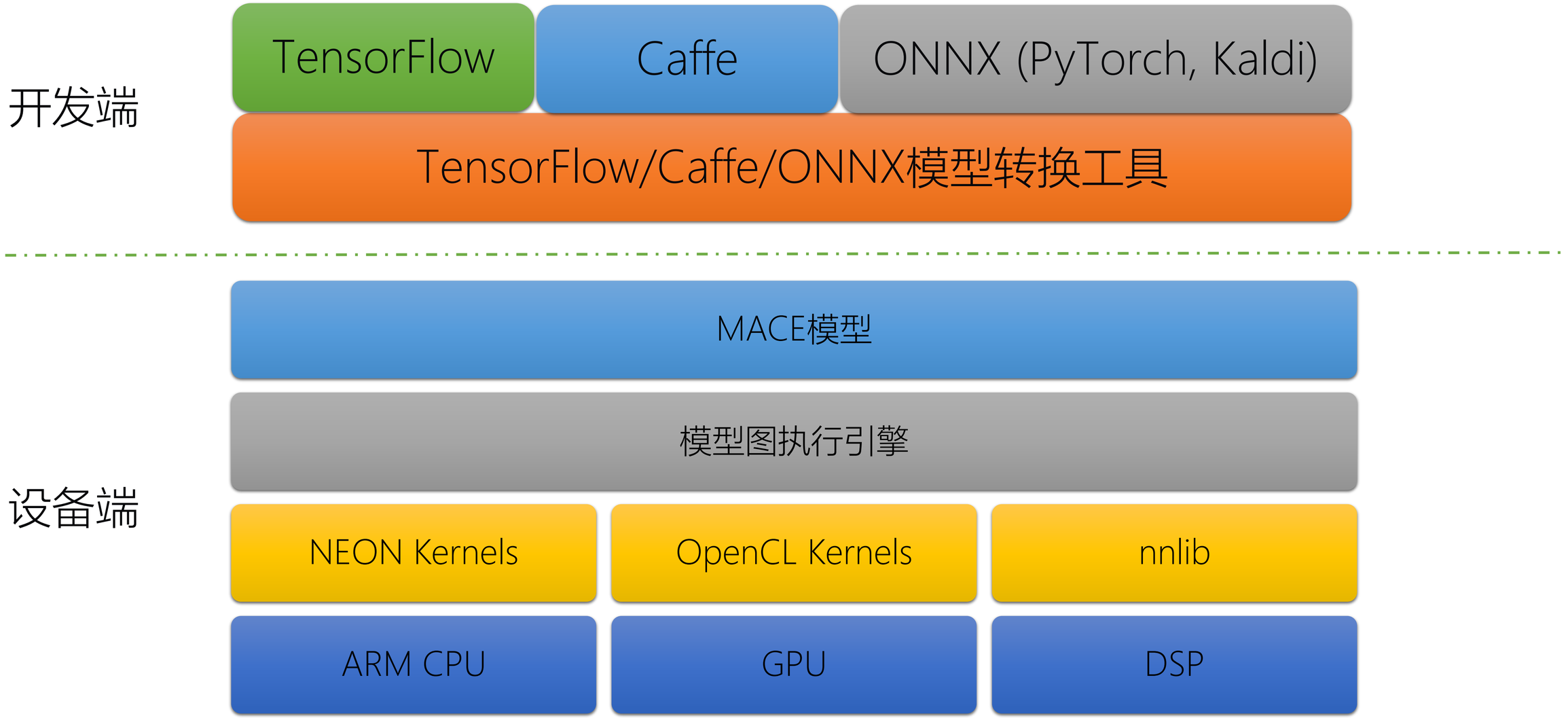
	TensorFlow Lite	Caffe	MACE
CPU速度	整型快	较慢	快
GPU速度	不支持	不支持	支持，快
DSP速度	不支持	不支持	支持HVX，快
模型加密	一般	一般	优

专有框架对比

	SNPE	MACE
是否开源	否	是
二次开发	难	易
CPU速度	慢	快
GPU支持	仅支持Adreno	支持多数移动GPU
GPU初始化速度	慢	快
GPU内存占用优化	差	优
GPU卡顿优化	易卡顿	不易卡顿
模型加密	较差	优

框架设计

MACE架构


开发端
TensorFlow
Caffe
ONNX (PyTorch, Kaldi)
TensorFlow/Caffe/ONNX模型转换工具
设备端
MACE模型
模型图执行引擎
NEON Kernels
OpenCL Kernels
nnlib
ARM CPU
GPU
DSP

MACE的使用

训练模型并导出

配置模型部署描述文件

```yaml
library_name: mobilenet-v1
target_abis: [armeabi-v7a, arm64-v8a]
model_graph_format: file
model_data_format: file
models:
  mobilenet_v1:
    platform: tensorflow
    model_file_path: https://cnbj1.fds.api.xiaomi.com/mace/miai-models/mobilenet-v1/mobilenet-v1-1.0.pb
    model_sha256_checksum: 71b10f540ece33c49a7b51f5d4095fc9bd78ce46ebf0300487b2ee23d71294e6
    subgraphs:
      - input_tensors:
          - input
        input_shapes:
          - 1,224,224,3
        output_tensors:
          - MobilenetV1/Predictions/Reshape_1
        output_shapes:
          - 1,1001
        validation_inputs_data:
          - https://cnbj1.fds.api.xiaomi.com/mace/inputs/dog.npy
    runtime: cpu+gpu
    limit_opencl_kernel_time: 0
    nnlib_graph_mode: 0
    obfuscate: 0
    winograd: 0
```

基础功能

转换模型

```
python tools/converter.py convert \
        --config=path/to/your/mobilenet-v1.yml
```

编译

```
bash tools/build-standalone-lib.sh
```

详细文档：https://mace.readthedocs.io

```
builds
├── include
│   └── mace
│       └── public
│           └── mace.h
├── lib
│   ├── arm64-v8a
│   │   └── cpu_gpu
│   │       ├── libmace.a
│   │       └── libmace.so
│   ├── armeabi-v7a
│   │   ├── cpu_gpu
│   │   │   ├── libmace.a
│   │   │   └── libmace.so
│   │   └── cpu_gpu_dsp
│   │       ├── libhexagon_controller.so
│   │       ├── libmace.a
│   │       └── libmace.so
│   └── linux-x86-64
│       ├── libmace.a
│       └── libmace.so
└── mobilenet-v1
    ├── model
    │   ├── mobilenet_v1.data
    │   └── mobilenet_v1.pb
    └── _tmp
        └── arm64-v8a
            └── mace_run_static
```

性能优化

优化原则

- 模型层面优化与精简
- 优化的实现算法 (e.g. winograd)
- 有效的实现方式 (e.g. cache aware, packing)
- 体系结构与指令级优化 (e.g. register, cache size , 指令)

模型结构优化

- 模型精简

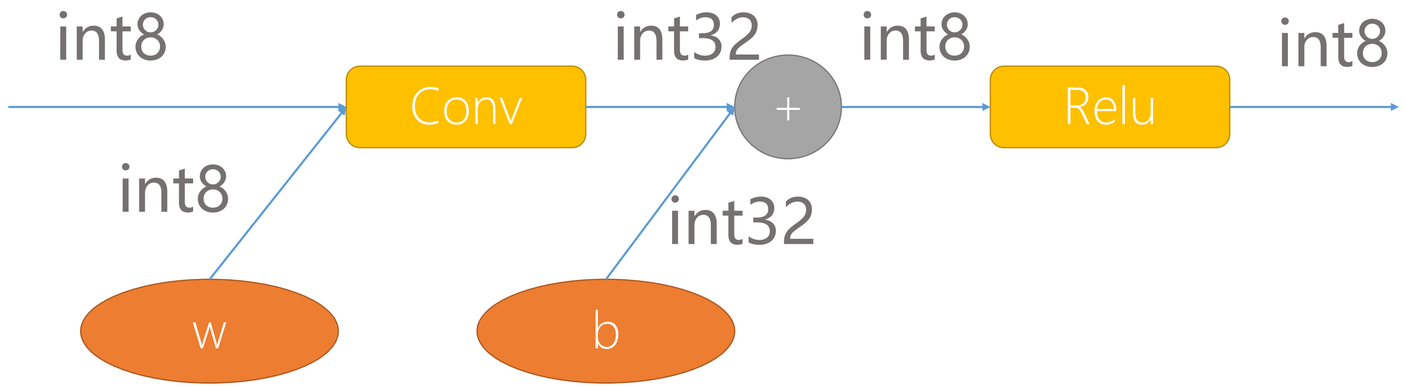
- BatchNorm 融合
  - $y = \frac{x - mean}{var} = scale * x + bias$
  - BN(Conv(x)) => Conv'(x)

- 激活函数融合
  - 减少Kernel数量
  - 减少访存次数

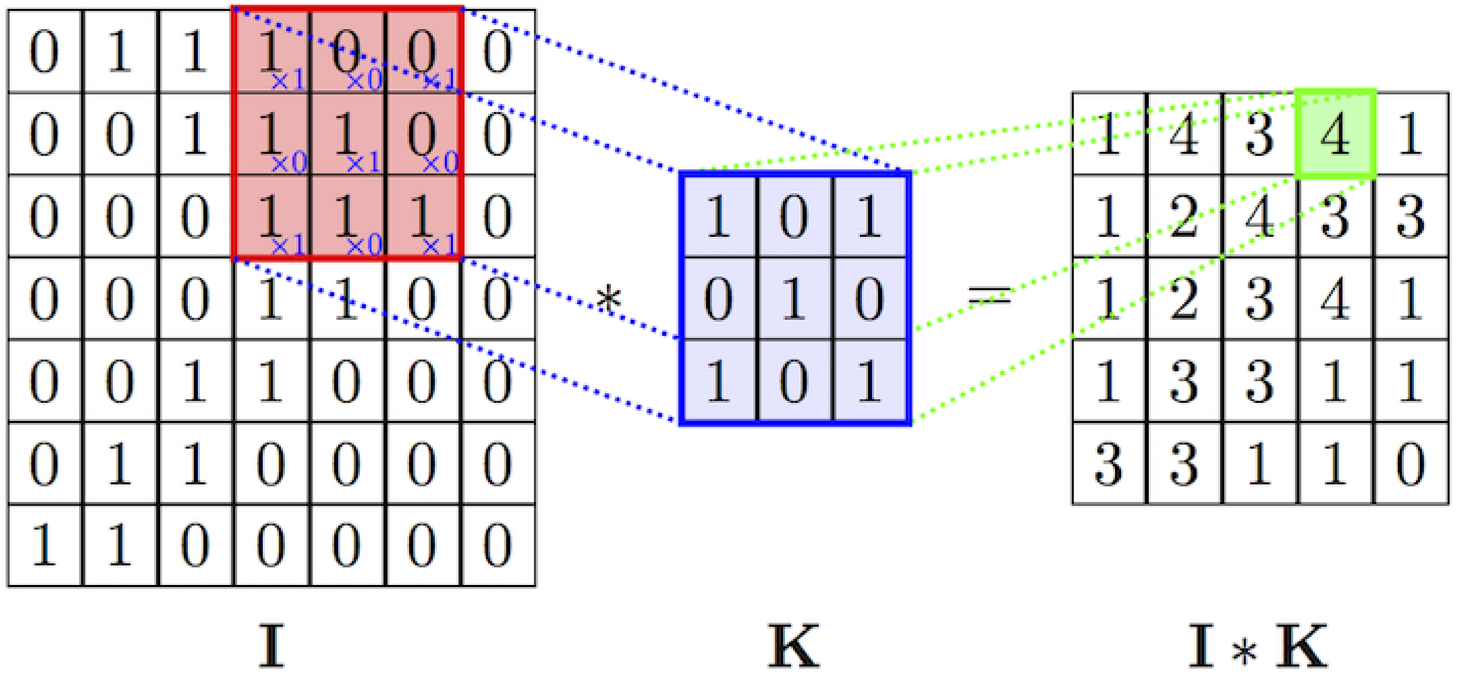
模型量化

$$R = S \times (Q - Z)$$

R：浮点值
Q：量化值
S：scale
Z：zero point



卷积算法层优化

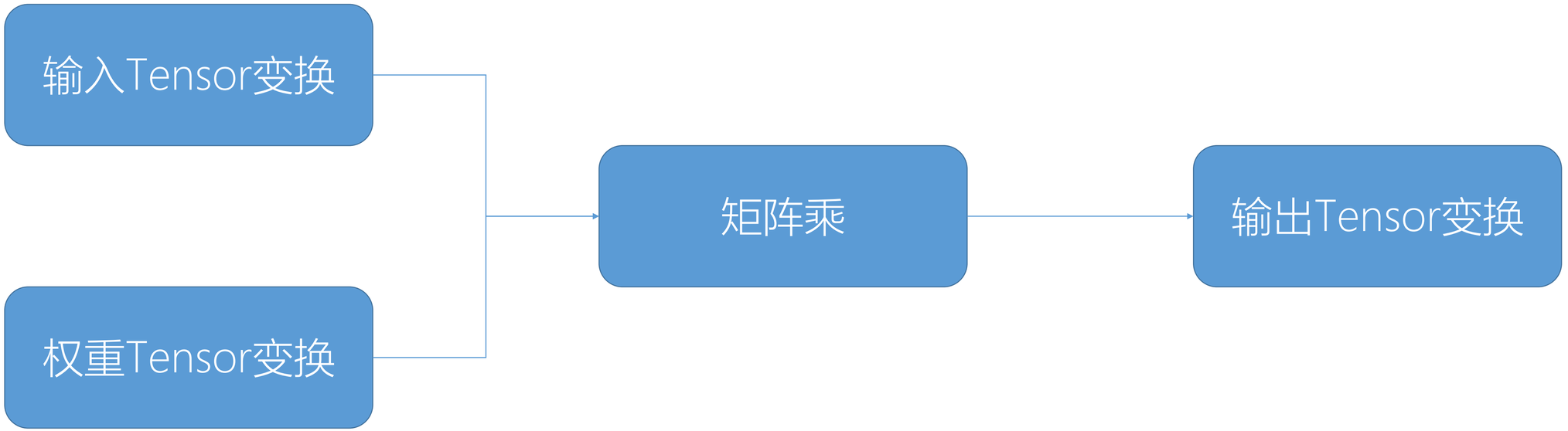


卷积实现方式

- 直接计算
- im2col + GEMM
- Winograd Convolution

Winograd算法思路

- 通用硬件乘法计算效率远低于加法
- 用廉价的计算取代昂贵的计算



1维Winograd卷积

$$Y = A^T\left[(Gg) \odot (B^T d)\right]$$

$$F(2,3) = \begin{bmatrix} d_0 & d_1 & d_2 \\ d_1 & d_2 & d_3 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} m_1 + m_2 + m_3 \\ m_2 - m_3 - m_4 \end{bmatrix}$$

$$m_1 = (d_0 - d_2)g_0 \qquad m_2 = (d_1 + d_2)\frac{g_0 + g_1 + g_2}{2}$$
$$m_4 = (d_1 - d_3)g_2 \qquad m_3 = (d_2 - d_1)\frac{g_0 - g_1 + g_2}{2}$$

标准卷积乘法数量：6

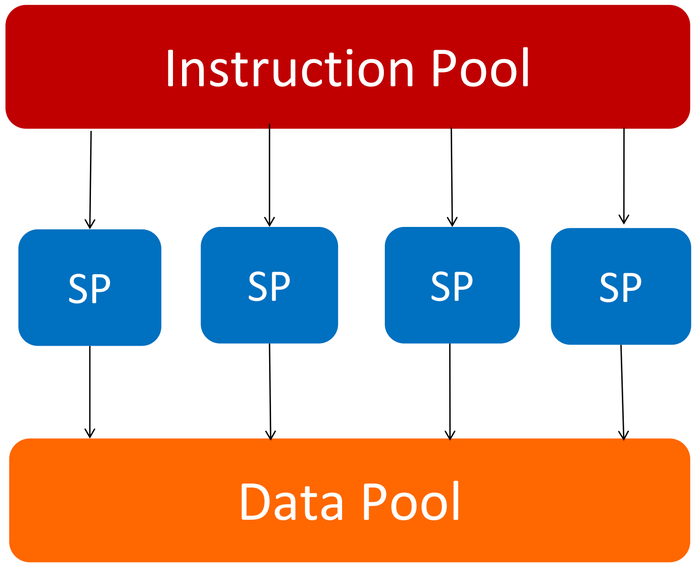
变换后乘法数量：4

2维Winograd卷积

Type	Tile Size	Memory Expansion	Speedup
f(2x2, 3x3)	4	Filter: (16 - 9) * Oc * Ic Input: ~ 4 * Ic * Oh * Ow Matmul:~ 4 * Oc * Oh * Ow	9/4.0 = 2.25
f(4x4, 3x3)	6	Filter: (36 - 9) * Oc * Ic Input ~ (36 / 16) * Ic * Oh * Ow Matmul~ (36 / 16) * Oc * Oh * Ow	9/2.25 = 4
f(6x6, 3x3)	8	Filter: (64 - 9) * Oc * Ic Input ~ (64 / 36) * Ic * Oh * Ow Matmul~ (64 / 36) * Oc * Oh * Ow	9/1.78 = 5.05

OpenCL优化

SIMT



访存方式

- Image/Buffer
- Memory Coalescing
- 寄存器
- Local memory

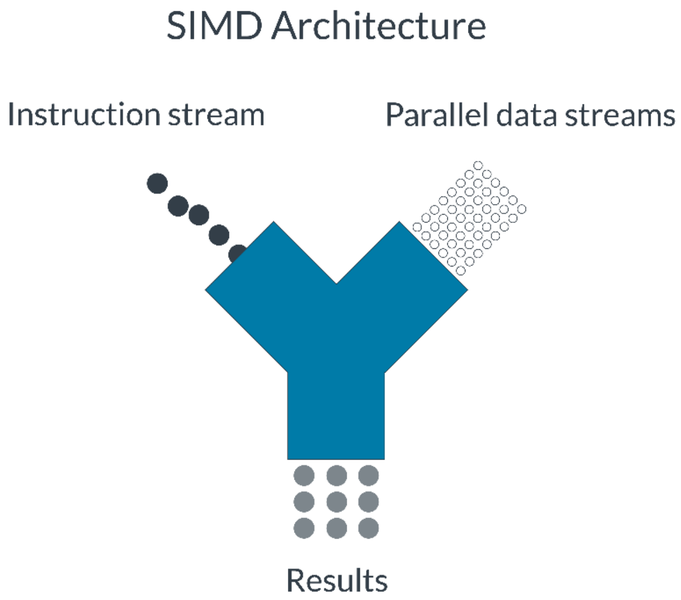
并行方式

- Workgroup大小
- Local workgroup 大小

代码层面

- 避免分支
- 减少乘除法
- mad指令
- Int24指令

NEON优化



分块

向量化

并行

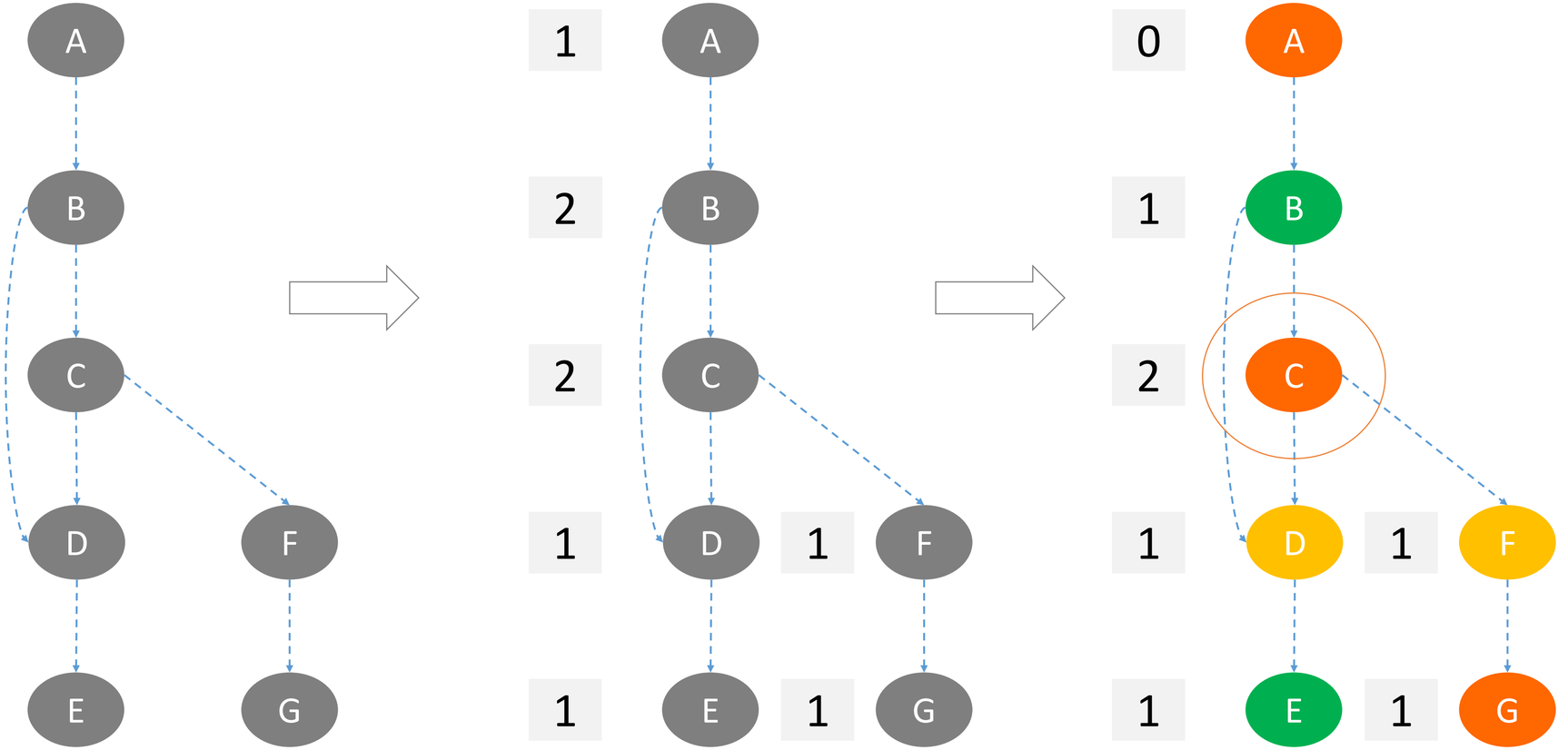
循环展开

循环顺序调整

指令优化

内存优化

内存优化(复用)

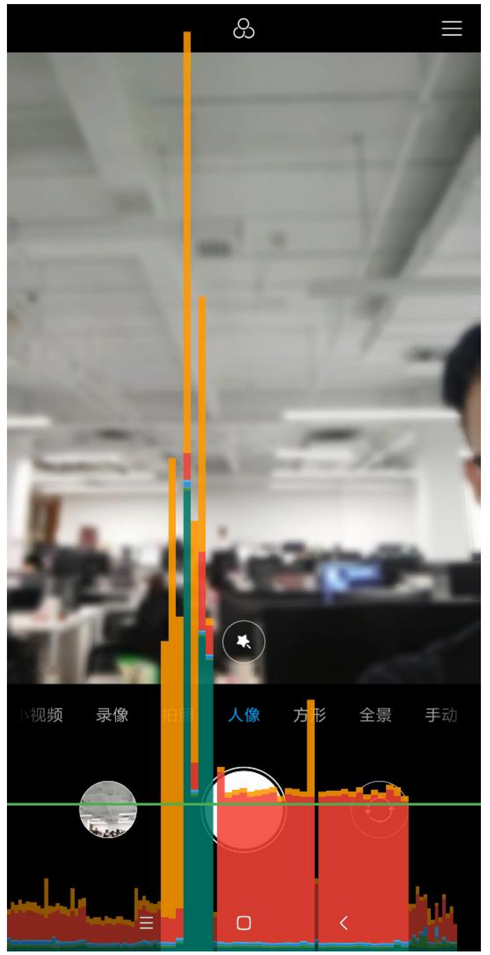


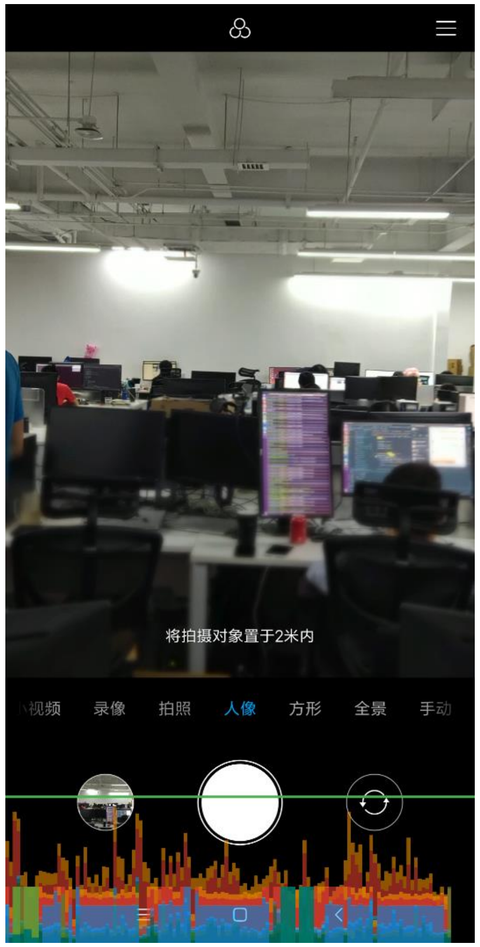
依赖分析　　引用计数　　基于染色的内存优化

用户体验优化

GPU卡顿优化

- 通过Tuning获取kernel计算时间
- 根据kernel计算时间拆分成< 1 ms的计算单元
- 设置OpenCL计算低优先级





模型安全

模型安全

$ strings apk/lib/armeabi-v7a/libalgo.so | grep conv

```
AnimationAssetBuilder::convert asset error!
Convolution conv1 16 4 4 4 4 0 0 1 1 2 9 4 19 2 5 data conv1
DepthwiseSeparableConvolution block1_conv_a 16 3 3 1 1 1 1 1 1 2 7 4 12 2 4 conv1 block1_conv_a
Convolution block1_conv_b 16 1 1 1 1 0 0 1 0 2 9 4 13 2 4 block1_conv_a block1_conv_b
Eltwise block1_eltsum conv1 block1_conv_b block1_eltsum 2 5 1
DepthwiseSeparableConvolution block2_conv_a 16 3 3 1 1 1 1 1 1 2 7 4 12 2 5 block1_eltsum block2_conv_a
Convolution block2_conv_b 16 1 1 1 1 0 0 1 0 2 9 4 14 2 4 block2_conv_a block2_conv_b
Eltwise block2_eltsum block1_eltsum block2_conv_b block2_eltsum 2 4 1
DepthwiseSeparableConvolution block3_conv_a 16 3 3 1 1 1 1 1 1 2 9 4 13 2 5 block2_eltsum block3_conv_a
Convolution block3_conv_b 16 1 1 1 1 0 0 1 0 2 9 4 14 2 5 block3_conv_a block3_conv_b
Eltwise block3_eltsum block2_eltsum block3_conv_b block3_eltsum 2 4 1
DepthwiseSeparableConvolution block4_conv_a 16 3 3 1 1 1 1 1 1 2 11 4 15 2 5 block3_eltsum block4_conv_a
Convolution block4_conv_b 16 1 1 1 1 0 0 1 0 2 9 4 14 2 4 block4_conv_a block4_conv_b
```

```
Convolution     conv1           1 1 data conv1 28 3 1 1 0 1 756
PReLU           prelu1          1 1 conv1 conv1_prelu1 28
Pooling         pool1           1 1 conv1_prelu1 pool1 0 3 2 0 0
Convolution     conv2           1 1 pool1 conv2 48 3 1 1 0 1 12096
PReLU           prelu2          1 1 conv2 conv2_prelu2 48
Pooling         pool2           1 1 conv2_prelu2 pool2 0 3 2 0 0
Convolution     conv3           1 1 pool2 conv3 64 2 1 1 0 1 12288
PReLU           prelu3          1 1 conv3 conv3_prelu3 64
InnerProduct    conv4           1 1 conv3_prelu3 conv4 128 1 73728
PReLU           prelu4          1 1 conv4 conv4_prelu4 128
Dropout         loss1/featdrop  1 1 conv4_prelu4 conv4_loss1/featdrop
InnerProduct    conv5           1 1 conv4_loss1/featdrop conv5 2 1 256
Softmax         prob            1 1 conv5 prob
Convolution     conv1/7x7_s2    1 1 data conv1/7x7_s2 32 7 1 2 3 1 1568
ReLU            conv1/relu_7x7  1 1 conv1/7x7_s2 conv1/7x7_s2/bn 0.000000
```

```
kModel/Predict/Stage_3/expand1/expand1/Conv/unit_1/bottleneck_v1/conv1/BatchNorm/batchnorm/sub/_497__cf__497
kModel/Predict/Stage_3/expand1/expand1/Conv/unit_1/bottleneck_v1/conv2/BatchNorm/batchnorm/sub/_494__cf__494
kModel/Predict/Stage_3/expand1/expand1/Conv/unit_1/bottleneck_v1/conv3/BatchNorm/batchnorm/sub/_491__cf__491
cModel/Predict/Stage_3/shrink1/Conv/unit_1/bottleneck_v1/conv3/BatchNorm/batchnorm/sub/_485__cf__485
cModel/Predict/Stage_3/bridge1/Conv/unit_1/bottleneck_v1/conv1/BatchNorm/batchnorm/sub/_482__cf__482
cModel/Predict/Stage_3/bridge1/Conv/unit_1/bottleneck_v1/conv2/BatchNorm/batchnorm/sub/_479__cf__479
cModel/Predict/Stage_3/bridge1/Conv/unit_1/bottleneck_v1/conv3/BatchNorm/batchnorm/sub/_476__cf__476
kModel/Predict/Stage_2/expand1/expand1/Conv/unit_1/bottleneck_v1/conv2/BatchNorm/batchnorm/sub/_473__cf__473
kModel/Predict/Stage_2/expand1/expand1/Conv/unit_1/bottleneck_v1/conv3/BatchNorm/batchnorm/sub/_470__cf__470
cModel/Predict/Stage_3/shrink1/Conv/unit_1/bottleneck_v1/conv1/BatchNorm/batchnorm/sub/_467__cf__467
cModel/Predict/Stage_3/shrink1/Conv/unit_1/bottleneck_v1/conv2/BatchNorm/batchnorm/sub/_464__cf__464
cModel/Predict/Stage_2/bridge1/Conv/unit_1/bottleneck_v1/conv1/BatchNorm/batchnorm/sub/_461__cf__461
```

模型安全

$ strings apk/lib/armeabi-v7a/libalgo.so | c++filt | clang-format

```
:NeuralNetworkModel::sigmoid(Eigen::Array<float, -1, -1, 0, -1, -1> &,
                             Eigen::Array<float, -1, -1, 0, -1, -1> &);
:NeuralNetworkModel::tanh(Eigen::Array<float, -1, -1, 0, -1, -1> &,
                          Eigen::Array<float, -1, -1, 0, -1, -1> &);
:NeuralNetworkModel::softmax(Eigen::Array<float, -1, -1, 0, -1, -1> &,
                             Eigen::Array<float, -1, -1, 0, -1, -1> &);
:NeuralNetworkModel::GRULayer(Eigen::Matrix<float, -1, -1, 0, -1, -1> &,
                              Eigen::Matrix<float, -1, -1, 0, -1, -1> &,
                              Eigen::Matrix<float, -1, -1, 0, -1, -1> &,
                              Eigen::Matrix<float, -1, 1, 0, -1, 1> &,
                              Eigen::Matrix<float, -1, -1, 0, -1, -1> &,
                              Eigen::Matrix<float, -1, 1, 0, -1, 1> &,
                              Eigen::Matrix<float, -1, -1, 0, -1, -1> &,
                              Eigen::Matrix<float, -1, -1, 0, -1, -1> &,
                              Eigen::Matrix<float, -1, -1, 0, -1, -1> &);
```

模型安全

- 常见的反向工程
  - 获取模型文件(DLC, pb)
  - Hook动态库调用，获取原始模型
  - 分析可执行文件得到模型结构信息

模型加密

- 模型结构 → C++代码 → 编译成二进制代码
- 模型结构相关字符串混淆
  - tower_0/MobilenetV2/expanded_conv_11/project/Conv2D(Conv2D) → dg93zxjf
- OpenCL kernel 混淆
- 使用静态库

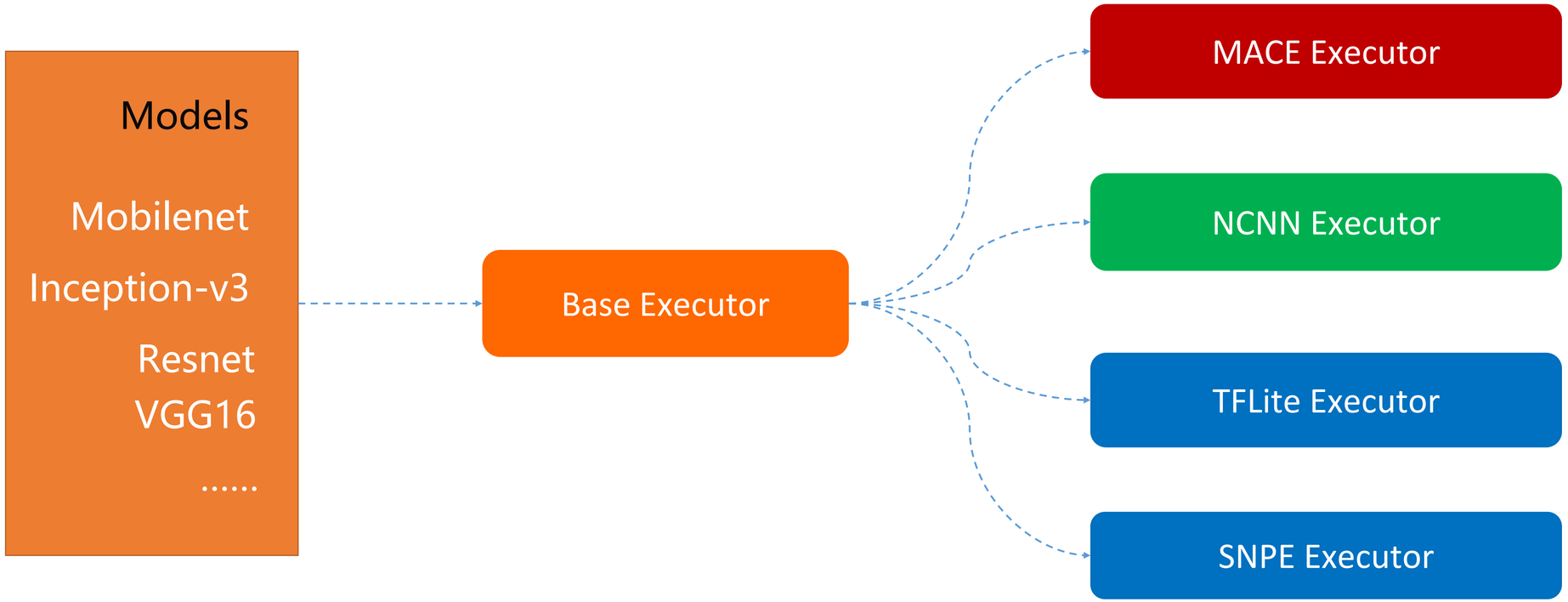
性能评测

基准测试

Mobile AI Bench

https://github.com/XiaoMi/mobile-ai-bench

Mobile AI Bench



https://github.com/XiaoMi/mobile-ai-bench

应用案例

应用场景

AI单摄

图片风格化

场景识别

图像超分辨率

翻译

语音

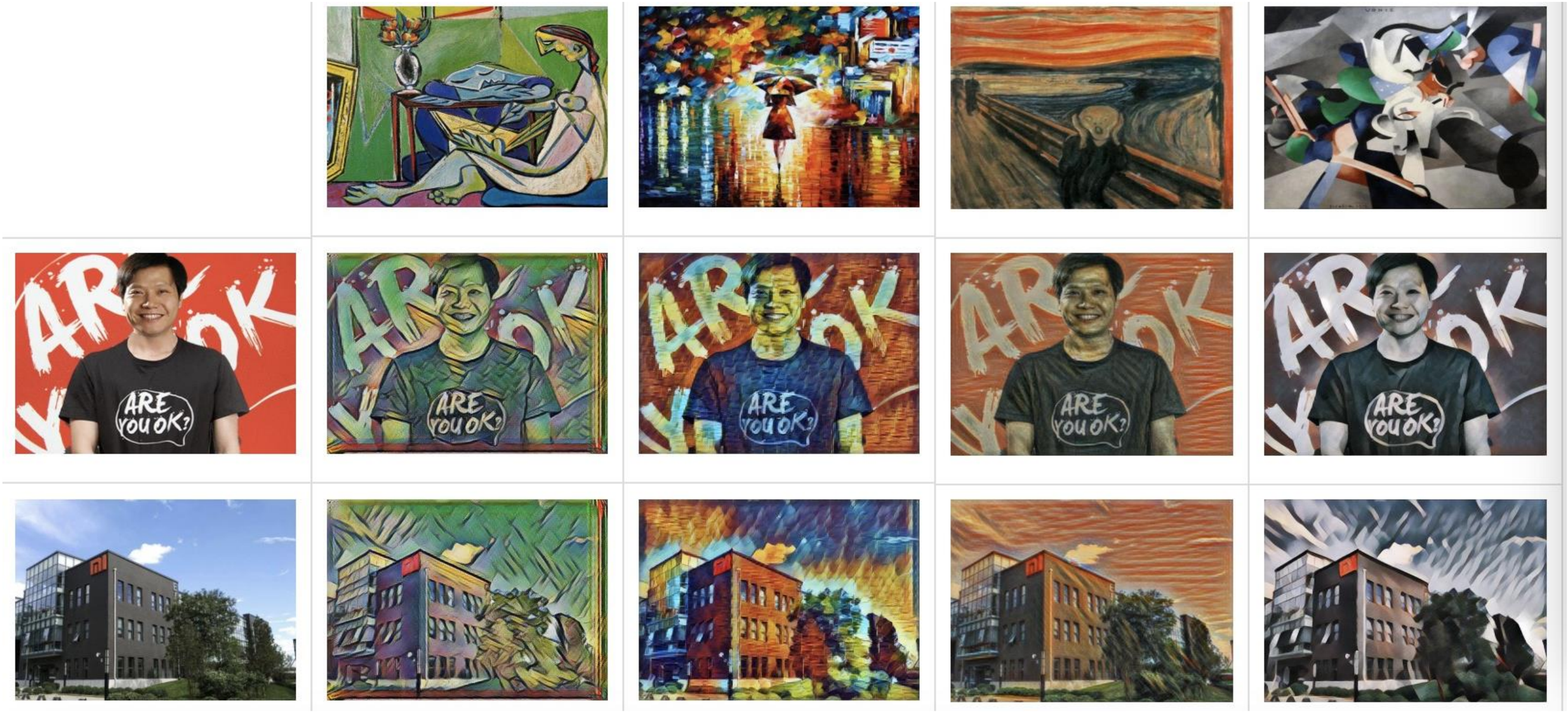

AI 单摄背景虚化
POWERED BY
MACE

POWERED BY MACE

图像超分辨率

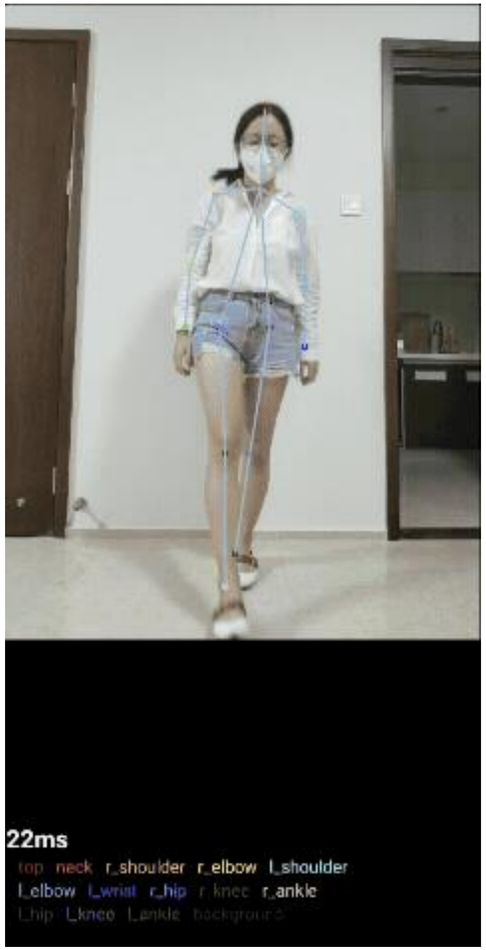
POWERED BY MACE

图片风格化



POWERED BY MACE

实时姿态估计



https://github.com/edvardHua/PoseEstimationForMobile

https://github.com/XiaoMi/mace-models/tree/master/convolutional-pose-machines

POWERED BY MACE

离线机器翻译

POWERED BY MACE

离线语音识别

KALDI

ONNX

项目地址

微信交流群

Thanks