Deep Learning on Mobile Devices:

An In-Depth Overview

Andrey Ignatov

andrey@vision.ee.ethz.ch

ETH Zurich, Switzerland

Up to

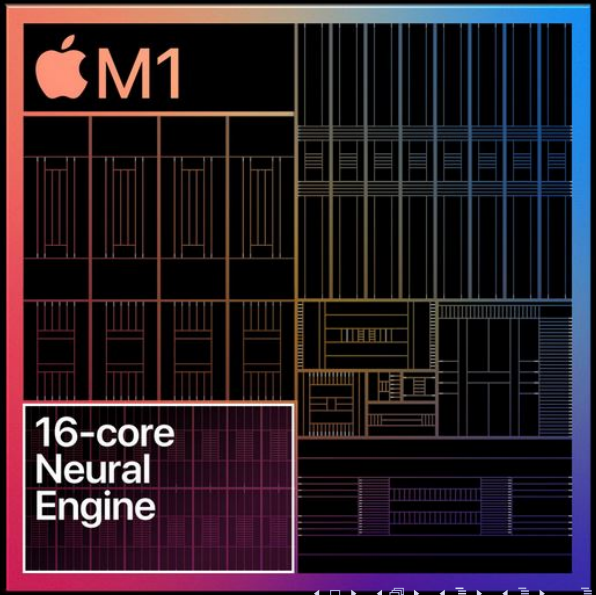
15x

faster machine learning performance[8]

Up to

11 trillion

operations per second



- What are the vendors talking about?
- What is the real performance of their machine learning hardware?
- How to deploy NN models on their AI accelerators?
- How to optimize neural networks for edge devices?

TALK SCHEDULE

Part 1: Deep Learning on Mobile Devices – Introduction

1. SOFTWARE: Mobile deep learning libraries, tools and software stacks

2. OPTIMIZATIONS: Key optimizations required for fast ML inference on the edge devices

3. HARDWARE: Specs and performance of the existing mobile processors with AI accelerators

4. HOWTO: Running any TensorFlow or PyTorch NN model on mobile devices in less than 5 minutes

MOBILE ML:

SOFTWARE


2020
2019
2018
2017

Going Back to the Past: Year 2015

Popular Deep Learning Libraries:

1. Theano
2. Caffe
3. Torch
4. TensorFlow: just launched

Going Back to the Past: Year 2015

Mobile / Android Support:

1. Theano: NO
2. Caffe: NO
3. Torch: NO
4. TensorFlow: Partial

Why Deep Learning on Mobile Devices?

- Photo & Video Denoising / Deblurring
- Photo & Video Super-Resolution
- HDR Image Processing
- Camera Scene Detection
- Image Categorization
- Face Recognition
- Augmented Reality
- Natural Language Translation
- Voice Recognition
- Voice Assistants
- Activity Recognition
- Gesture Recognition
- Music Tracking
- Adaptive Power Management

Mobile Deep Learning SDKs

- **2016, Qualcomm:** Snapdragon Neural Processing Engine (SNPE) SDK
  Frameworks: TensorFlow, Caffe, Caffe2, ONNX
  Acceleration: Snapdragon Hexagon DSPs + Adreno GPUs
- **2017, Huawei:** HiAI SDK
  Frameworks: TensorFlow, Caffe
  Acceleration: Kirin NPUs
- **2018, MediaTek:** NeuroPilot SDK
  Frameworks: TensorFlow, TFLite, Caffe, Caffe2, MXNet, NNabla
  Acceleration: MediaTek APUs + GPUs
- **2019, Samsung:** Samsung Neural SDK
  Frameworks: Tensorflow, Caffe
  Acceleration: Exynos NPUs + GPUs
- **2019, Unisoc:** UNIAI SDK
  Frameworks: Tensorflow, Caffe, ONNX
  Acceleration: Unisoc / Imagination NPUs

Mobile Deep Learning SDKs: HowTo

1. Train your deep learning model with TensorFlow, Caffe, Torch, etc.

2. Convert it to a special format defined by vendor's SDK

3. Run the model with acceleration on mobile platforms supported by the SDK.

Problem: Each framework supports only hardware from the corresponding vendor!

Mobile Deep Learning Libraries

- **2015:** TensorFlow Mobile

Full TensorFlow port for Arm devices

Runs the standard TensorFlow .pb models

Supports all Android devices!

Acceleration: CPU only ☹☹

Mobile Deep Learning Libraries

- **2017:** TensorFlow Lite

Replaced TensorFlow Mobile

Is not a TensorFlow port, designed from scratch

Operators / kernels are optimized for mobile inference

Supports a subset of TensorFlow ops and layers

Supports all Android devices!

Acceleration: Android NNAPI & TFLite Delegates

TFLite & On-Device Inference

Exporting your TensorFlow model to TFLite format:

```
import tensorflow as tf

converter = tf.lite.TFLiteConverter.from_saved_model("saved/model/dir")

tflite_model = converter.convert()
open("model.tflite", "wb").write(tflite_model)
```

On-device inference:

```
Interpreter.Options tfLiteOptions = new Interpreter.Options();

Interpreter tfLite = new Interpreter(loadModelFile(...), tfLiteOptions);

inputTensor =   // define the input tensor
outputTensor =  // define the output tensor

tfLite.run(inputTensor, outputTensor);
```

Mobile Deep Learning Libraries

- **2017:** TensorFlow Lite

Replaced TensorFlow Mobile

Is not a TensorFlow port, designed from scratch

Operators / kernels are optimized for mobile inference

Supports a subset of TensorFlow ops and layers

Uses its own .tflite model format, conversion required

Supports all Android devices!

Acceleration: Android NNAPI & TFLite Delegates

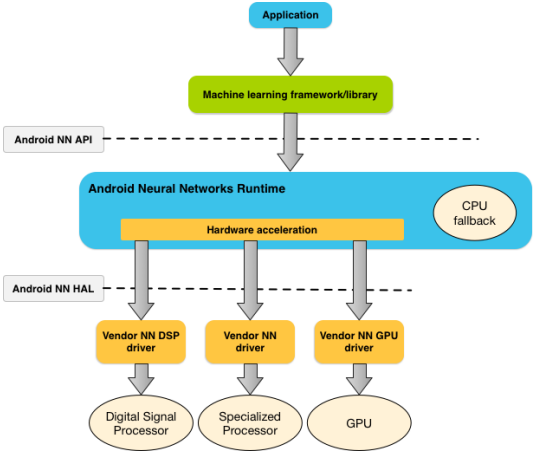
IDEA:

1. Move all low-level NPU and DSP NN drivers inside the device firmware

2. Create a unified API for these drivers

3. Mobile ML libraries interact with NPUs, DSPs and GPUs through this API

Android Neural Network API (NNAPI): Android 8.1+



Intermediate layer between the higher-level DL frameworks and the device's hardware acceleration drivers

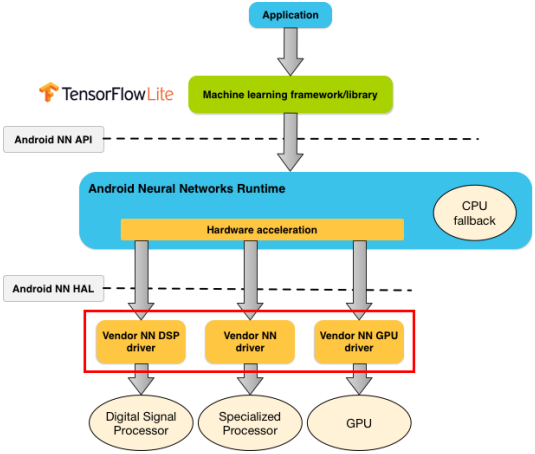
Running the computations on CPU:

```
Interpreter.Options tfLiteOptions = new Interpreter.Options();

Interpreter tfLite = new Interpreter(loadModelFile(...), tfLiteOptions);
```

Running the computations with NNAPI:

```
Interpreter.Options tfLiteOptions = new Interpreter.Options();

Delegate delegate = new NnApiDelegate();
tfLiteOptions.addDelegate(delegate);

Interpreter tfLite = new Interpreter(loadModelFile(...), tfLiteOptions);
```

Android Neural Network API: Drivers



Vendor NN drivers are missing / outdated / contain bugs — issues with acceleration!

NNAPI Drivers: Fragmentation

- Typical NNAPI drivers development **lifecycle:**

0 month after chipset released: partial functioning, critical issues might exist
6 month after chipset released: functional, critical issues are fixed, some performance issues still exist
12 month after chipset released: fully functional, satisfactory performance
24 month after chipset released: fully functional, improved performance and Android+1 features support
36 month after chipset released: deprecation

NNAPI Drivers: Fragmentation

- Typical NNAPI drivers development **lifecycle:**

  0 month after chipset released: partial functioning, critical issues might exist
  6 month after chipset released: functional, critical issues are fixed, some performance issues still exist
  12 month after chipset released: fully functional, satisfactory performance
  24 month after chipset released: fully functional, improved performance and Android+1 features support
  36 month after chipset released: deprecation

- Until Android 12: NNAPI drivers are not updatable without full firmware update

  → SoC vendor *(Qualcomm, MediaTek, Unisoc)* develops new NNAPI drivers version
  → New NNAPI drivers are sent to smartphone vendor *(Sony, Nokia, Asus, Xiaomi, OPPO, etc.)*
  → Smartphone vendor tests them and integrates into new phone's firmware
  → The updated Android firmware is sent to the end user

NNAPI Drivers: Fragmentation

- Typical NNAPI drivers development **lifecycle:**

  0 month after chipset released: partial functioning, critical issues might exist
  6 month after chipset released: functional, critical issues are fixed, some performance issues still exist
  12 month after chipset released: fully functional, satisfactory performance
  24 month after chipset released: fully functional, improved performance and Android+1 features support
  36 month after chipset released: deprecation

- Until Android 12: NNAPI drivers are not updatable without full firmware update

  → SoC vendor *(Qualcomm, MediaTek, Unisoc)* develops new NNAPI drivers version
  → New NNAPI drivers are sent to smartphone vendor *(Sony, Nokia, Asus, Xiaomi, OPPO, etc.)*
  → Smartphone vendor tests them and integrates into new phone's firmware
  → The updated Android firmware is sent to the end user

- We have now **50+** mobile chipset models with NNAPI drivers

NNAPI Drivers: Fragmentation

- Typical NNAPI drivers development **lifecycle:**

  0 month after chipset released: partial functioning, critical issues might exist
  6 month after chipset released: functional, critical issues are fixed, some performance issues still exist
  12 month after chipset released: fully functional, satisfactory performance
  24 month after chipset released: fully functional, improved performance and Android+1 features support
  36 month after chipset released: deprecation

- Until Android 12: NNAPI drivers are not updatable without full firmware update

  → SoC vendor *(Qualcomm, MediaTek, Unisoc)* develops new NNAPI drivers version
  → New NNAPI drivers are sent to smartphone vendor *(Sony, Nokia, Asus, Xiaomi, OPPO, etc.)*
  → Smartphone vendor tests them and integrates into new phone's firmware
  → The updated Android firmware is sent to the end user

- Since Android 12, NNAPI drivers can be updated separately via Google Play Services

Android NNAPI: Ops Coverage

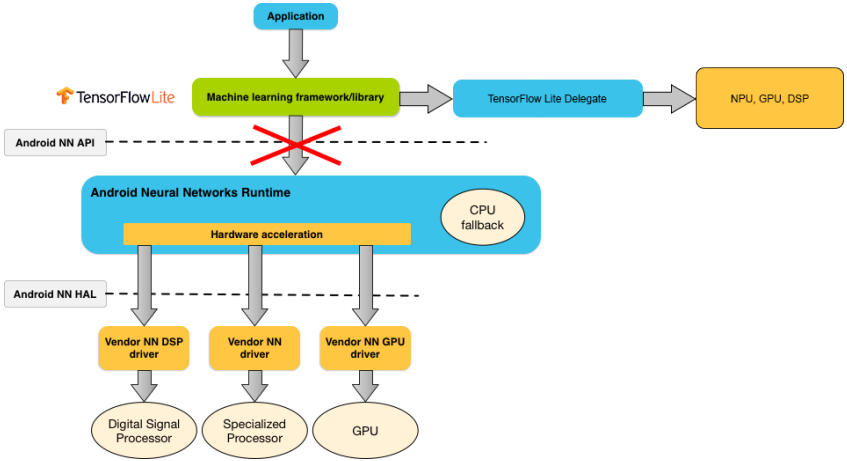
- TensorFlow Lite ← supports a subset of normal TensorFlow ops
  NNAPI ← supports a subset of TensorFlow Lite ops
  NN HAL ← supports a subset of NNAPI ops

Android NNAPI: Ops Coverage

- TensorFlow Lite ← supports a subset of normal TensorFlow ops
  NNAPI ← supports a subset of TensorFlow Lite ops
  NN HAL ← supports a subset of NNAPI ops

- Android 8.1: **NNAPI 1.0** *(do not use!)*
  28 TensorFlow ops **supported**
- Android 9: **NNAPI 1.1** *(use with caution)*
  37 TensorFlow ops **supported**
- Android 10: **NNAPI 1.2**
  94 TensorFlow ops **supported**
- Android 11: **NNAPI 1.3**
  101 TensorFlow ops **supported**

TensorFlow Lite Delegates: Replacing NNAPI



TensorFlow Lite Delegates: Replacing NNAPI

Using TFLite delegates instead of NNAPI:

```
Interpreter.Options tfLiteOptions = new Interpreter.Options();

Delegate delegate = new YourTFLiteDelegate();
tfLiteOptions.addDelegate(delegate);

Interpreter tfLite = new Interpreter(loadModelFile(...), tfLiteOptions);
```

Advantages:

- Independent of Android OS system
- Overcomes all Android NNAPI limitations
- Better performance and optimization
- Support for legacy devices

TensorFlow Lite GPU Delegate: https://www.tensorflow.org/lite/performance/gpu

General information:

+ Inference acceleration on any GPU supporting ***OpenCL*** or ***OpenGL 3.0+***

  ~ *Can run even on the Snapdragon S4 Pro released 9 years ago*

+ Provides great acceleration for the majority of architectures
+ Can run huge models *(consuming up to several GB of RAM)*
— Issues with the accuracy on some devices for some models *(almost random outputs)*
— Not optimized for PowerVR graphics

Ops and layers coverage:

- TensorFlow → developed by Google
- TensorFlow Lite → developed by Google
- TensorFlow GPU delegate → developed by Google
- General OpenCL/GL kernel implementation → the largest ops coverage

Qualcomm Hexagon TFLite Delegate:

https://www.tensorflow.org/lite/performance/hexagon_delegate

General information:

- Supports all Qualcomm SoCs with NN-compatible Hexagon DSPs:
  - *Snapdragon 820 and above*
  - *Snapdragon 710 and above*
  - *Snapdragon 660 and above*
  - *Snapdragon 460 and above*
- Better ops support and coverage compared to NNAPI drivers
- Better runtime results compared to NNAPI (especially for older SoCs):

  Snapdragon 845, NNAPI, Inception-V3: 39.1ms
  Snapdragon 845, Hexagon NN, Inception-V3: 31.1ms (-20%)

  Snapdragon 855, NNAPI, Inception-V3: 14.7ms
  Snapdragon 855, Hexagon NN, Inception-V3: 11.0ms (-25%)

- Some vendors disable ☹ external access to the Hexagon DSP (*e.g.*, *Google*, *OnePlus*)

Samsung Eden, MediaTek Neuron, Apple CoreML TFLite Delegates

Samsung Eden:

- Supports high-end and mid-range Samsung SoCs:
  → *Exynos 2100, 1080, 990, 980, 9825, 9820, 9630*
- GPU inference, optimized kernels for *Mali graphics*
- Non-public

MediaTek Neuron:

- Supports high-end and mid-range Dimensity SoCs:
  → *Dimensity 1200, 1100, 1000(+,L), 1000, 900, 820, 800*
- Requires Android 11+
- NPU inference, comparable or faster runtime than with NNAPI
- Non-public

Apple CoreML:

- Supports Apple A12 and newer SoCs, requires iOS 12+
- Inference on Apple's Neural Engine
- Issues with quantized models acceleration

Overall summary:

- TensorFlow Lite 2 years ago ☺

Pros and Cons:

- Good ops coverage
- Flawless CPU inference
- Acceleration only through NNAPI
- Terrible NNAPI support ☹

MOBILE ML:
OPTIMIZATIONS

Standard Architecture Optimizations

- No large convolutional filters *(e.g., 5×5, 7×7)*
- No large feature maps *(e.g., 128, 256, 512)*
- Try to move all processing to lower scales / resolutions
- Make sure that all model layers are supported by NNAPI:

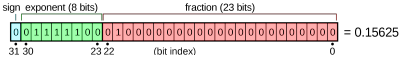
  `https://developer.android.com/ndk/reference/group/neural-networks`

- Learning by doing: more information in MAI Challenge Papers & Top Methods

Model and Inference Types

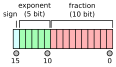
- **FP32:** single-precision floating point

  accuracy: 7-8 digits after decimal point



- **FP16:** half-precision floating point

  accuracy: 3 digits after decimal point



- **INT8:** 8-bit integer

  signed: from −128 to 127
  unsigned: from 0 to 255

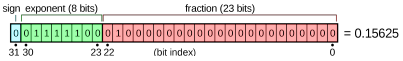
- **INT16:** 16-bit integer

  signed: from −32768 to 32767
  unsigned: from 0 to 65535

FP32 Inference Type

- **FP32:** single-precision floating point

  accuracy: 7-8 digits after decimal point



- When training the network with *TensorFlow*, *PyTorch*, *etc.* — you get FP32 model

  The accuracy of FP16 format is not sufficient for gradients update if no other tricks are used

- The accuracy of FP32 *inference* mode is generally excessive for the majority of tasks:

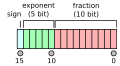
  E.g., classification, recognition, image processing, etc.

- NPUs do not support FP32 inference!

FP16 Inference Type

- **FP16:** half-precision floating point

  accuracy: 3 digits after decimal point



- The accuracy of FP32 inference mode is generally excessive for the majority of tasks

  E.g., classification, recognition, image processing, etc.

- FP16 inference is **faster** and consumes **less RAM**

- FP32 → FP16 conversion happens automatically in all mobile libraries:

  TFLite: setAllowFp16(true) option

  TFLite GPU delegate: setPrecisionLossAllowed(true) option

INT8 Inference Type / Quantized inference

- **INT8:** 8-bit integer

  signed: from −128 to 127
  unsigned: from 0 to 255

- Mapping the original floating-point values to INT8 interval:

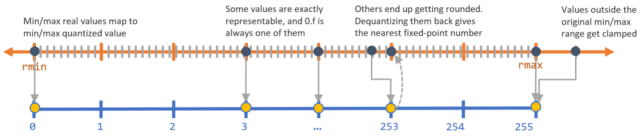


Image Source: https://bit.ly/3gzJ0z1

INT8 Inference Type / Quantized inference

- **INT8:** 8-bit integer

  signed: from −128 to 127
  unsigned: from 0 to 255

- Sample distribution of model weights before and after quantization:

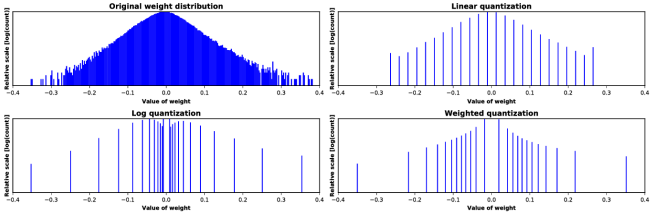


Image Source: *Eunhyeok Park et al.*, Weighted-Entropy-Based Quantization for Deep Neural Networks

INT8 Inference Type / Quantized inference

- **INT8:** 8-bit integer

  signed: from −128 to 127
  unsigned: from 0 to 255

Advantages:

- Many mobile NPUs support integer inference only (e.g., *Exynos NPUs* or *Qualcomm Hexagon DSPs*)
- The model becomes 4x and 2x smaller compared to FP32 and FP16 formats, respectively
- Faster inference on almost all hardware
- Smaller RAM consumption
- Higher power efficiency

INT8 Inference Type / Quantized inference

- **INT8:** 8-bit integer

  signed: from –128 to 127
  unsigned: from 0 to 255

Issues: Accuracy ☹

- 32/16 → 8 weight bit-width reduction affects the precision of the computations:

+ Works well for many image classification tasks *(threshold-based problems)*
± Works so-so for some simple image processing tasks *(e.g., image super-resolution)*
– Does not work for complex vision problems *(e.g., ISP image processing)* and many NLP tasks

INT8 Quantization: HowTo

- **Dynamic range quantization:** good accuracy but useless for acceleration:

  "At inference, weights are converted from 8-bits of precision to floating point and computed using floating-point kernels."

  https://www.tensorflow.org/lite/performance/post_training_quantization#dynamic_range_quantization

- **Full integer quantization:** good accuracy but often useless for acceleration:

  Inputs / outputs and some ops are not quantized

  https://www.tensorflow.org/lite/performance/post_training_quantization#full_integer_quantization

- **Integer only:** lowest accuracy but the model can run on all INT8 NPUs:

  "you can enforce full integer quantization for all ops including the input and output, by using the following steps"

  https://www.tensorflow.org/lite/performance/post_training_quantization#integer_only

---

- **Quantization aware training:** requires model re-training but usually the best accuracy:

  https://www.tensorflow.org/model_optimization/guide/quantization/training

INT16 Inference Type

- **INT16:** 16-bit integer

  signed: from -32768 to 32767
  unsigned: from 0 to 65535

- Good alternative for FP16 inference: 16-bit weights, same model size while integer-only
- No major accuracy problems, suitable for NLP and image processing tasks

- Not supported by many NPUs
- **16-bit activations with 8-bit weights:** the only option available in TensorFlow Lite:

  "activations are quantized based on their range to 16-bits, weights are quantized in 8-bit integer and bias is quantized into 64-bit integer"

  https://www.tensorflow.org/lite/performance/post_training_quantization#integer_only_16-bit_activations_with_8-bit_weights_experimental

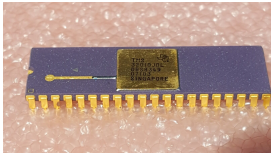
MOBILE ML:

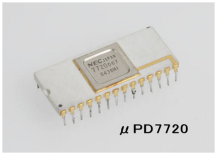
AI HARDWARE


2020
2019
2018
2017

Going back to early 1980s...

- Early 80s, computer processors are very slow and not suitable for multimedia tasks...
  - → low clock rates, no special vector instructions / extensions
  - → *e.g., Intel 8086/8088 (5-10 MHz), Zilog Z8000 (5.5 MHz), Motorola 68000 (8 MHz)*
- Additional digital signal processors (DSPs) for multimedia apps:
  - First presented in the early 1980s: *NEC PD7720*, *AT&T DSP1*, *TI TMS32010*
  - Very efficient at matrix and vector operations
  - Harvard architecture, VLIW and SIMD instruction sets, hardware block for multiply-accumulate (MAC) operations
  - *Used for computer graphics, sound and video decoding, as accelerators for photo editing software*





Going back to late 1980s...

- Backpropagation applied to handwritten zip code recognition

  Yann LeCun, Bernhard Boser, et al.

  Neural computation 1, no. 4 (1989)

  "AT&T DSP-32C general purpose DSP with a peak performance of 12.5 million multiply add operations per second on 32 bit floating point numbers (25 MFLOPS)"

5.2 DSP Implementation. During the recognition process, almost all the computation time is spent performing multiply accumulate operations, a task that digital signal processors (DSP) are specifically designed for. We used an off-the-shelf board that contains 256 kbytes of local memory and an AT&T DSP-32C general purpose DSP with a peak performance of 12.5 million multiply add operations per second on 32 bit floating point numbers (25 MFLOPS). The DSP operates as a coprocessor; the host is a personal computer (PC), which also contains a video acquisition board connected to a camera.

The personal computer digitizes an image and binarizes it using an adaptive thresholding technique. The thresholded image is then scanned and each connected component (or segment) is isolated. Components that are too small or too large are discarded; remaining components are sent to the DSP for normalization and recognition. The PC gives a variable sized pixel map representation of a single digit to the DSP, which performs the normalization and the classification.

The overall throughput of the digit recognizer including image acquisition is 10 to 12 classifications per second and is limited mainly by the normalization step. On normalized digits, the DSP performs more than 30 classifications per second.

6 Conclusion

Going back to late 90s...

- Digital signal processors (DSPs):
  - + Very efficient at matrix and vector operations
  - + Very power efficient
  - – Very specialized, designed for some particular task or even app

---

- Late 90s — GPUs killed DSPs on the consumer computer market since:
  - – Are not specialized, can be used for various tasks
  - + Powerful enough
  - + Power efficiency — not the main feature for desktops

DSPs: Conquering The Mobile Market

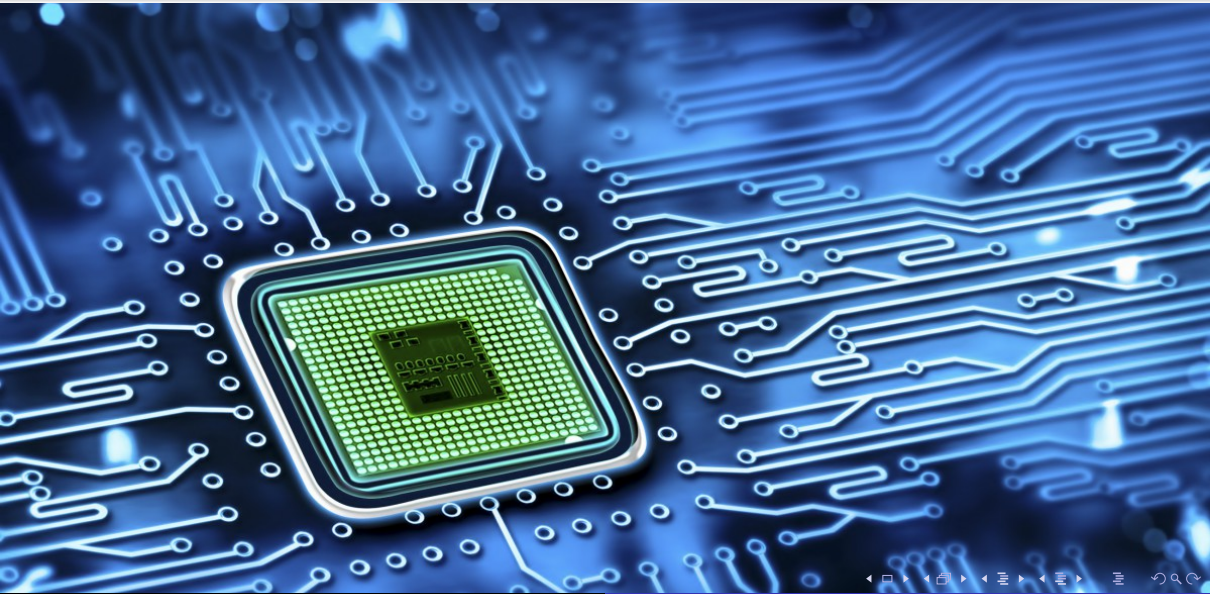
- Digital signal processors (DSPs):
  - + Very efficient at matrix and vector operations
  - + Very power efficient
  - → At the beginning, were used for signal processing only

DSPs: Conquering The Mobile Market

- Phones – 2000s: → DSPs started being used for music and image encoding / decoding:

- Phones – 2005s: → More complex image, video and sound processing tasks:

What are NPUs?



Neural Network Inference

- Running neural networks:
  - Mainly matrix and vector operations
  - Huge power consumption on general-purpose hardware
  - A restricted set of used ops

Mobile Deep Learning Hardware

- **2016, Qualcomm:** Snapdragon 820
  Accelerator: Hexagon V6 68x DSP
- **2017, Huawei:** Kirin 970
  Accelerator: NPU / Cambricon
- **2017, Samsung:** Exynos 8895
  Accelerator: Vision Processing Unit
- **2017, Google:** Pixel 2
  Accelerator: Pixel Visual Core
- **2018, MediaTek:** Helio P60
  Accelerator: AI Processing Unit
- **2019, Unisoc:** Tiger T710
  Accelerator: NPU / Imagination Technologies

Qualcomm Chipsets: NN Inference Acceleration

- Hexagon DSPs:
  - ∗ INT8 NN inference support
  - ◦ Accessible through the *Hexagon TFLite delegate* and *NNAPI*
  - ◦ Can be found in:

    Snapdragon 820 and above
    Snapdragon 710 and above
    Snapdragon 660 and above
    Snapdragon 460 and above

- Adreno GPUs:
  - ∗ FP16 NN inference support
  - ◦ Accessible through the *TFLite GPU delegate* and *NNAPI*
  - ◦ Compatible models: *Adreno 304* and above

MediaTek Chipsets: NN Inference Acceleration

- APU 3.0:
  - INT8 & FP16 NN inference support
  - Accessible through the *Neuron TFLite delegate* and *NNAPI*
  - Can be found in:

    Dimensity 1200, 1100, 1000(+/L)
    Dimensity 900, 820, 800

- APU 2.0:
  - INT8 & FP16 NN inference support
  - Accessible through *NNAPI*
  - Can be found in: *Helio P95, P90*

- APU 1.0:
  - INT8 NN inference support
  - Accessible through *NNAPI*
  - Can be found in: *Helio G95, G90T, P70, P60*

Huawei Chipsets: NN Inference Acceleration

- NPU, Da Vinci:
  - ⋆ INT8 & FP16 NN inference support
  - ◦ Accessible through *NNAPI*
  - ◦ Can be found in:

    3-Core, Refreshed: Kirin 9000
    3-Core: Kirin 990 5G
    2-Core: Kirin 990, 985
    1-Core: Kirin 820, 810

- NPU, Cambricon:
  - ⋆ FP16 NN inference support
  - ◦ Accessible through *NNAPI*
  - ◦ Can be found in:

    2-Core: Kirin 980
    1-Core: Kirin 970

Samsung Chipsets: NN Inference Acceleration

- NPU:
  - ∗ INT8 NN inference support
  - ◦ Accessible through *NNAPI*
  - ◦ Can be found in:

    Exynos 2100
    Exynos 990 (some models)

- GPU:
  - ∗ INT8 & FP16 NN inference support
  - ◦ Accessible through the *Eden TFLite delegate* and *NNAPI*
  - ◦ Supported models (Eden):

    Exynos 2100, 990
    Exynos 9825, 9820, 980

Unisoc & Google Chipsets: NN Inference Acceleration

- Unisoc Tiger T770:
  - NPU: INT8 NN inference support
  - Accessible through *NNAPI*
- Unisoc Tiger T710:
  - NPU: INT8 & FP16 NN inference support
  - Accessible through *NNAPI*

- Google Pixel 4 / Edge TPU:
  - INT8 NN inference support
  - Accessible through *NNAPI*

MOBILE ML:
BENCHMARKING

AI Benchmark (ETHZ)

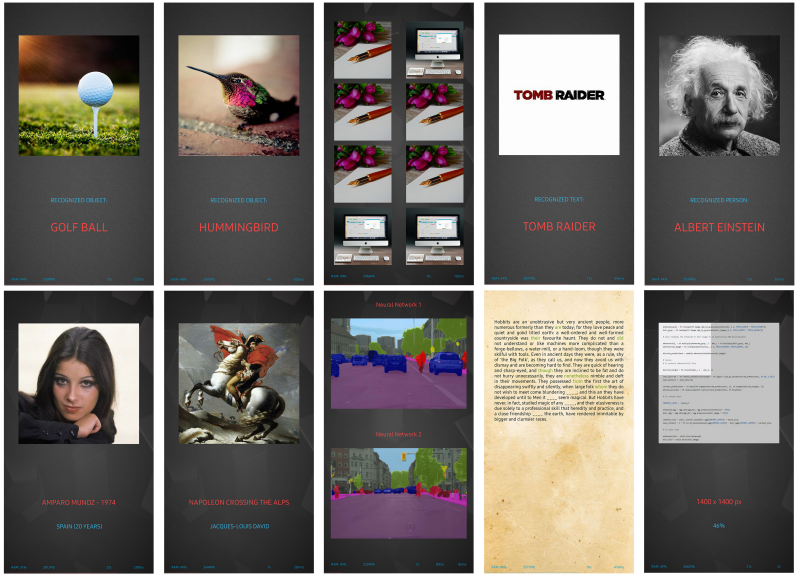
- Android application for measuring smartphones' AI performance
- First public version: May 2018 *(17 public releases, last: 4.0.4)*
- TensorFlow Lite + Android NNAPI | TensorFlow Lite Delegates
- 14 Sections / 46 Tests:

- Section 1. Classification, MobileNet-V2 (INT8 + FP16 + Batch Size=4)
- Section 2. Classification, Inception-V3 (INT8 + FP16)
- Section 3. Face Recognition, MobileNet-V3 (INT8 + FP16)
- Section 4. Classification, MobileNet-V2 x 8 (INT8 + FP16 | Parallel)
- Section 5. OCR, CRNN (ResNet-18 + Bi-LSTM) (FP16 + FP32)
- Section 6. Deblurring, PyNET (INT8 + FP16)
- Section 7. Super-Resolution, VGG19 (INT8 + FP16)
- Section 8. Super-Resolution, SRGAN (INT8 + FP16)
- Section 9. Bokeh Rendering, U-Net (INT8 + FP16)
- Section 10. Semantic Segmentation, DeepLab-V3+ (INT8 + FP16)
- Section 11. Semantic Segmentation, DeepLab-V3+ (INT8 + FP16 In Parallel)
- Section 12. Image Enhancement, DPED ResNet (INT8 + FP16)
- Section 13. Text Completion, LSTM (FP16)
- Section 14. Memory limits, SRCNN (FP16 | Various Resolutions)

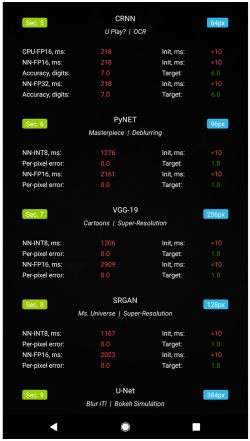
AI Benchmark (ETHZ): Measurements

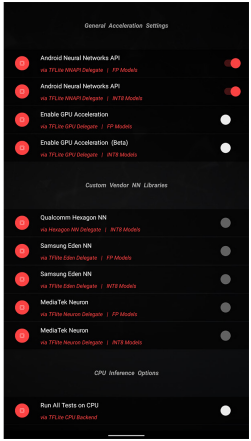
- FLOAT-16 performance (NPU, APU, GPU)
- INT-8 performance (NPU, APU, GPU, DSP)
- CPU single- and multi-thread performance (FP32, INT8)
- Single / throughput inference times
- Memory / RAM performance
- Initialization time
- Accuracy

AI Benchmark (ETHZ): Visualization



AI Benchmark (ETHZ): Visualization





AI Benchmark Results: FP16 Inference

SoC Model	Accelerator	MobileNet V2, ms	Inception V3, ms	MobileNet V3, ms	PyNET, ms	VGG-19, ms	SRGAN, ms	U-Net, ms	Deeplab, V3+, ms	DPED, ms	Relative Perf, %
HiSilicon Kirin 9000	NPU (3 Cores, Da Vinci series)	3.3	8.9	8.8	15	18	24	9.4	18	5	100
HiSilicon Kirin 990 5G	NPU (3 cores, Da Vinci series)	3.9	12	11	25	42	79	14	50	7.5	57
Exynos 2100	GPU (Mali-G78 MP14)	4.2	38	7.8	63	57	58	20	28	14	44.6
HiSilicon Kirin 990	NPU (2 cores, Da Vinci series)	4.4	17	12	39	68	98	17	55	13	43
HiSilicon Kirin 985 5G	NPU (2 cores, Da Vinci series)	5.3	20	17	40	69	107	20	62	13	38
Snapdragon 888	GPU (Adreno 660)	6.7	29	13	66	60	70	25	39	18	36.5
MediaTek Dimensity 1200	APU 3.0 (6 cores)	3.7	28	11	74	99	97	26	30	24	35.7
MediaTek Dimensity 1100	APU 3.0 (6 cores)	3.9	30	12	83	110	106	28	32	27	32.7
MediaTek Dimensity 1000+	APU 3.0 (6 cores)	4.2	31	12	83	111	105	27	34	27	32.3
MediaTek Dimensity 1000L	APU 3.0 (6 cores)	4.8	32	14	84	113	110	29	36	28	30.3
Exynos 990	GPU (Mali-G77 MP11)	12	53	18	94	85	83	31	44	25	26
Exynos 9825 Octa	GPU (Mali-G76 MP12)	11	53	18	119	123	111	40	46	32	22.4
Snapdragon 865 Plus	GPU (Adreno 650)	6.1	55	11	127	180	396	38	39	39	20.8
Exynos 1080	GPU (Mali-G78 MP10)	19	62	24	111	103	105	51	49	24	20.7
MediaTek Dimensity 820	APU 3.0 (4 cores)	5.9	50	19	124	186	177	45	51	44	20.4
Exynos 9820 Octa	GPU (Mali-G76 MP12)	15	59	22	137	136	128	53	49	34	19.1
MediaTek Dimensity 800	APU 3.0 (4 cores)	5.8	57	19	144	221	194	61	53	52	18.2
Snapdragon 865	GPU (Adreno 650)	9.6	65	15	145	204	430	43	45	47	17.2
Snapdragon 855 Plus	GPU (Adreno 640)	8.4	66	14	159	230	384	47	46	52	17
Snapdragon 855	GPU (Adreno 640)	9.3	75	16	185	263	443	53	49	58	15
HiSilicon Kirin 820	NPU (Da Vinci series)	6.3	24	18	3390	76	132	1279	72	17	12.9
Snapdragon 845	GPU (Adreno 630)	11	93	21	243	358	648	73	71	79	11.2
HiSilicon Kirin 810	NPU (Da Vinci series)	9.2	34	24	3330	116	161	1251	97	26	10
Snapdragon 835	GPU (Adreno 540)	15	116	26	298	459	836	84	92	100	8.9
Exynos 980	GPU (Mali-G76 MP5)	25	148	42	287	337	268	112	107	94	8.8
Snapdragon 765G	GPU (Adreno 620)	15	135	30	379	564	684	112	110	122	7.8
Snapdragon 732G	GPU (Adreno 618)	14	144	31	395	601	738	118	122	130	7.4
Snapdragon 820	GPU (Adreno 530)	22	142	30	364	569	886	104	101	129	7.3
Snapdragon 720G	GPU (Adreno 618)	13	151	31	418	639	719	126	130	137	7.2
Snapdragon 730G	GPU (Adreno 618)	14	160	33	446	679	782	132	134	146	6.8
Snapdragon 730	GPU (Adreno 618)	15	160	33	448	680	781	133	134	146	6.7
Snapdragon 712	GPU (Adreno 616)	15	181	36	508	779	943	151	153	167	6
Snapdragon 750G	GPU (Adreno 619)	18	136	47	559	549	746	547	263	120	5.3
Snapdragon 810	GPU (Adreno 430)	39	191	52	434	760	1409	144	143	172	5.1
Snapdragon 710	GPU (Adreno 616)	18	216	43	609	931	1086	179	179	199	5.1
MediaTek Helio G95	GPU (Mali-G76 MP4)	21	154	56	949	761	676	620	251	153	4.4
Snapdragon 690	GPU (Adreno 619L)	24	175	56	772	729	971	770	344	160	4
MediaTek Helio P60	GPU (Mali-G72 MP3)	18	264	45	2603	537	958	648	277	116	4
MediaTek Helio G90T	GPU (Mali-G76 MP4)	27	189	69	1050	842	807	687	299	179	3.8
Snapdragon 660	GPU (Adreno 512)	19	307	52	925	1483	1393	249	241	317	3.7
Snapdragon 675	GPU (Adreno 612)	22	325	57	992	1655	1260	278	253	344	3.5
MediaTek Helio P90	APU 2.0	52	159	148	505	253	1945	1656	1209	58	3.2
Exynos 7904	GPU (Mali-G71 MP2)	46	397	84	1371	815	1222	249	430	184	3.2
Snapdragon 801	GPU (Adreno 330)	37	397	94	1030	1622	1764	293	356	335	2.8

AI Benchmark Results: INT8 Inference

SoC Model	Accelerator	MobileNet V2, ms	Inception V3, ms	MobileNet V3, ms	PyNET, ms	VGG-19, ms	SRGAN, ms	U-Net, ms	Deeplab, V3+, ms	DPED, ms	Relative Perf, %
Snapdragon 888	DSP (Hexagon 780)	1.1	2.6	1.8	5.2	8.2	10	3.2	9	1.9	100
Exynos 2100	NPU	2.2	4.1	5	8.4	7.4	36	6	9.3	4	55.9
MediaTek Dimensity 1200	APU 3.0 (6 cores)	1.8	8.3	4.4	22	26	36	9.3	11	8	36.3
HiSilicon Kirin 9000	NPU (3 Cores, Da Vinci series)	2.6	7.2	8.5	13	11	29	11	67	4.1	34
MediaTek Dimensity 1100	APU 3.0 (6 cores)	2	9.1	4.6	24	29	39	10	11	8.7	33.6
MediaTek Dimensity 1000+	APU 3.0 (6 cores)	2.2	9.3	4.8	24	29	39	10	13	8.8	32.4
MediaTek Dimensity 1000L	APU 3.0 (6 cores)	2.6	10	5.5	25	29	43	11	14	9.5	29.8
Snapdragon 865 Plus	DSP (Hexagon 698)	3.1	10	7.1	26	37	65	16	39	11	22.1
Snapdragon 855 Plus	DSP (Hexagon 690)	3.2	10	7.5	26	38	65	17	37	11	21.8
MediaTek Dimensity 820	APU 3.0	2.9	15	6.3	35	47	60	16	18	14	21.8
Snapdragon 855	DSP (Hexagon 690)	3.6	11	7.8	27	39	66	17	42	11	20.7
Snapdragon 865	DSP (Hexagon 698)	3.4	11	7.4	26	38	65	17	56	11	20.5
MediaTek Dimensity 800	APU 3.0 (4 cores)	2.8	16	5.9	40	56	61	21	17	16	20.3
HiSilicon Kirin 990 5G	NPU (3 cores, Da Vinci series)	4.3	13	16	35	27	93	24	92	9.7	16.1
HiSilicon Kirin 990	NPU (2 cores, Da Vinci series)	5.7	19	19	38	48	103	25	104	13	12.8
Snapdragon 765G	DSP (Hexagon 696)	4.5	18	11	49	76	97	30	84	20	12.5
SDM855 + Pixel Neural Core	Hexagon DSP + Google TPU	5.5	11	6.6	32	2168	44	151	30	13	10.6
Snapdragon 675	DSP (Hexagon 685)	5.6	24	14	67	99	120	37	102	26	9.8
Snapdragon 845	DSP (Hexagon 685)	6.1	33	15	95	108	130	46	101	27	8.5
Exynos 990	GPU (Mali-G77 MP11)	12	43	18	112	91	98	48	41	29	8.5
Snapdragon 712	DSP (Hexagon 685)	5.8	33	16	102	112	139	50	81	29	8.4
Snapdragon 732G	DSP (Hexagon 688)	5.8	34	16	97	107	138	48	109	27	8.3
Snapdragon 730G	DSP (Hexagon 688)	6	34	16	98	105	138	48	111	28	8.2
Snapdragon 710	DSP (Hexagon 685)	6.4	34	16	97	105	135	48	120	27	8.2
Snapdragon 720G	DSP (Hexagon 692)	7.3	32	19	87	135	157	49	82	34	7.8
Exynos 9825 Octa	GPU (Mali-G76 MP12)	12	45	18	117	136	123	49	41	35	7.7
Snapdragon 750G	DSP (Hexagon 694)	9.2	36	19	77	120	103	193	97	31	6.9
Exynos 9820 Octa	GPU (Mali-G76 MP12)	15	52	22	125	146	132	53	43	38	6.9
Exynos 1080	GPU (Mali-G78 MP10)	18	53	32	126	100	109	55	45	35	6.8
Snapdragon 690	DSP (Hexagon 692)	9.5	39	20	86	138	118	223	89	35	6.3
Snapdragon 665	DSP (Hexagon 686)	9.2	43	23	119	190	201	65	128	47	5.8
Snapdragon 460	DSP (Hexagon 683)	8.7	42	23	119	189	202	67	143	47	5.8
Snapdragon 662	DSP (Hexagon 683)	8.9	43	23	119	190	198	68	146	47	5.7
HiSilicon Kirin 985	NPU (2 cores, Da Vinci series)	7.2	21	20	1374	51	132	1132	115	14	5.1
Snapdragon 835	DSP (Hexagon 682)	8.6	60	27	164	259	255	85	155	60	4.6
Snapdragon 820	DSP (Hexagon 680)	8.4	59	25	159	272	269	95	152	61	4.6
Snapdragon 660	DSP (Hexagon 680)	9.3	63	27	168	289	287	88	141	65	4.4
HiSilicon Kirin 820 5G	NPU (Da Vinci series)	9.2	26	26	1401	67	167	1215	140	18	4.2
MediaTek Helio P90	APU 2.0	11	31	21	606	63	272	1296	198	20	4.1
Exynos 980	GPU (Mali-G76 MP5)	22	103	42	276	327	290	115	93	79	3.4
HiSilicon Kirin 810	NPU (Da Vinci series)	11	37	30	1381	98	273	1223	338	24	3.1
MediaTek Helio G95	APU 1.0	16	62	37	430	234	224	1017	144	56	3
MediaTek Helio G90T	APU 1.0	17	63	37	437	235	227	1048	154	57	2.9
MediaTek Helio P70	APU 1.0	31	141	86	929	622	539	1661	339	151	1.3
MediaTek Helio P60	APU 1.0	30	140	85	947	621	537	1735	344	150	1.3

Performance Ranking

https://ai-benchmark.com/ranking

View Detailed Results

Model	CPU	RAM	Year	Android	Updated	Lib	CPU-Q Score	CPU-F Score	INT8 NNAPI 1.1	INT8 NNAPI 1.2	INT8 Accuracy	FP16 NNAPI 1.1	FP16 NNAPI 1.2	FP16 Accuracy	INT8 Parallel	FP16 Parallel	LSTM	Memory	AI Score, K
Huawei Mate 40 Pro	Kirin 9000	8GB	2020	10	11.20	nn	4346	8584	24398	14280	62.4	37064	73781	81.9	1814	4204	461	2100	216.6
Huawei Mate 40 Pro+	Kirin 9000 5G	12GB	2020	10	11.20	nn	4338	8379	24446	14174	62.4	37289	73145	81.9	1839	4101	451	2100	215.3
Oppo Find X3 Pro	Snapdragon 888	12GB	2021	11	4.21	nn	5502	9632	43220	47295	92.1	12673	26491	61.8	4575	2103	708	2700	205.4
LG V70	Snapdragon 888	8GB	2021	11	4.21	nn	4948	8714	43790	47090	92.1	12762	26502	61.8	4714	1973	736	2700	203.1 [1]
Samsung Galaxy S21 Ultra	Exynos 2100	12GB	2021	11	4.21	ne	4839	9632	32314	27394	93	15982	37336	86.4	2237	2780	7324	2800	202.0 [5]
Samsung Galaxy S21 Ultra	Exynos 2100	12GB	2021	11	2.21	nn	4649	9495	25392	23549	93	12659	31980	86.4	1917	2035	6263	2800	170.4
Samsung Galaxy S21+	Exynos 2100	8GB	2021	11	2.21	nn	4603	9396	25224	23782	93	12647	31883	86.4	1912	2054	6263	2800	170.1
Samsung Galaxy S21	Exynos 2100	8GB	2021	11	2.21	nn	4549	9268	25457	23599	93	12620	31851	86.4	1953	2051	6283	2800	169.9
Realme GT Neo	MediaTek Dimensity 1200	8GB	2021	11	4.21	nn	5009	9655	19561	20646	97.7	14395	28314	83.6	2886	3748	1507	2800	156.1
vivo S9	MediaTek Dimensity 1100	8GB	2021	11	3.21	nn	4556	9092	18054	19371	97.7	13289	26332	83.6	2729	3570	1426	2800	145.2
Xiaomi Redmi K30 Ultra	MediaTek Dimensity 1000+	8GB	2020	10	11.20	nn	4155	8452	17040	18470	97.7	12868	25831	84.2	2297	3169	1476	2800	139.3
Xiaomi Mi 11	Snapdragon 888	12GB	2020	11	1.21	nn	4693	7774	31421	26025	92.1	7662	19451	62	3373	969	218	2800	138.9
Honor 30 Pro+	Kirin 990 5G	8GB	2020	10	2.21	nn	3985	8140	12259	6523	94.8	25560	37993	83	896	2665	493	2100	137.7
Huawei P40 Pro	Kirin 990 5G	8GB	2020	10	11.20	nn	3941	8100	12213	6487	94.8	25663	37896	83	900	2617	491	2100	137.5
Realme X7 Pro	Mediatek Dimensity 1000+	8GB	2020	10	11.20	nn	4152	8471	16319	17760	97.7	12565	25224	84.2	2240	3108	1351	2800	136.1

Performance Ranking

https://ai-benchmark.com/ranking_processors

View Detailed Results

Processor	CPU Cores	AI Accelerator	Year	Lib	CPU-Q Score	CPU-F Score	INT8 NNAPI 1.1	INT8 NNAPI 1.2	INT8 Accuracy	FP16 NNAPI 1.1	FP16 NNAPI 1.2	FP16 Accuracy	INT8 Parallel	FP16 Parallel	LSTM	AI Score, K
Snapdragon 888	1x2.84 GHz & 3x2.42 GHz & 4x1.80 GHz Kryo 680	DSP (Hex. 780) + GPU (Adreno 660)	2020	nn	5401	9891	53925	54216	92.1	14096	28718	61.8	5032	2107	808	156.5
HiSilicon Kirin 9000	1x3.13GHz + 3x2.54GHz A77 & 4x2.04GHz A55	NPU (3 Cores, Da Vinci series)	2020	nn	4346	8584	24398	14280	62.4	37064	73781	81.9	1814	4204	461	148.0
Exynos 2100	1x2.9 GHz X1 & 3x2.80 GHz A78 & 4x2.2 GHz A55	NPU + GPU (Mali-G78 MP14)	2021	ne	4839	9632	32314	27394	93	15982	37336	86.4	2237	2780	7324	133.2
MediaTek Dimensity 1200	1x3.0 GHz A78 & 3x2.6 GHz A78 & 4x2.0 GHz A55	APU 3.0 (6 cores)	2021	nn	5009	9655	19561	20646	97.7	14395	28314	83.6	2886	3748	1507	102.9
MediaTek Dimensity 1100	4x2.6 GHz Cortex-A78 & 4x2.0 GHz Cortex-A55	APU 3.0 (6 cores)	2021	nn	4556	9092	18054	19371	97.7	13289	26332	83.6	2729	3570	1426	95.8
HiSilicon Kirin 990 5G	2x2.86 + 2x2.36 GHz A76 & 4x1.95 GHz A55	NPU (3 cores, Da Vinci series)	2019	nn	3973	8148	12248	6517	94.8	25242	37848	82.7	892	2533	485	93.6
MediaTek Dimensity 1000+	4x2.6 GHz Cortex A77 & 4x2.0 GHz Cortex A55	APU 3.0 (6 cores)	2019	nn	4159	8516	17124	18559	97.7	12970	25879	84.2	2280	3288	1513	92.3
MediaTek Dimensity 1000L	4x2.2 GHz Cortex A77 & 4x2.0 GHz Cortex A55	APU 3.0 (6 cores)	2020	nn	3613	7653	15495	16824	97.7	12130	24408	84.2	2002	2917	1296	84.4
HiSilicon Kirin 990	2x2.86 + 2x2.09 GHz A76 & 4x1.86 GHz A55	NPU (2 cores, Da Vinci series)	2019	nn	3823	7322	8662	5832	94.8	18126	30741	83	621	2322	485	74.6
HiSilicon Kirin 985 5G	1x2.58 + 3x2.40 GHz A76 & 4x1.84 GHz A55	NPU (2 cores, Da Vinci series)	2020	nn	3392	6890	8107	5334	94.8	16181	26293	82.5	527	1466	431	65.5
MediaTek Dimensity 820	4x2.6 GHz Cortex-A76 & 4x2.0 GHz Cortex-A55	APU 3.0	2020	nn	3622	7218	11718	12906	97.7	8347	16357	83.7	1742	2206	831	63.4
Snapdragon 865 Plus	1x3.1 + 3x2.42 + 4x1.8 GHz Kryo 585	DSP (Hex. 698) + GPU (Adreno 650)	2020	hg	3989	7190	13885	11193	90.3	8524	17021	54.8	1543	1627	2607	59.9
MediaTek Dimensity 800	4x2.0 GHz Cortex A76 & 4x2.0 GHz Cortex A55	APU 3.0 (4 cores)	2020	nn	2990	6594	10891	12081	97.7	7617	14711	83.7	1746	2308	1227	58.8
Snapdragon 855 Plus	1x2.96 + 3x2.42 + 4x1.80 GHz Kryo 485	DSP (Hex. 690) + GPU (Adreno 640)	2019	hg	3912	8304	13336	11019	90.3	6820	14379	54.8	1472	1489	2463	56.8
HiSilicon Kirin 990 5G	2x2.86 + 2x2.36 GHz A76 & 4x1.95 GHz A55	NPU (2 cores, Da Vinci series)	2019	nn	3820	7072	10752	1158	86.2	24896	6022	82.7	496	2264	472	56.0

SoS Model	MobileNet-V2								Inception-V3						MobileNet-V3 (Large-Minimalistic)						MobileNet-V2	
	CPU-Q	CPU-F	NN-INT8			NN-FP16			CPU-Q	CPU-F	NN-INT8		NN-FP16		CPU-Q	CPU-F	NN-INT8		NN-FP16		1 Model	2 Model
	ms	ms	ms	ms, bs=4	error, L1	ms	ms, bs=4	acc, digits	ms	ms	ms	error, L1	ms	acc, digits	ms	ms	ms	error, L1	ms	acc, digits	ms	ms
Snapdragon 888	8.4	16	1.1	0.5	23.5	6.7	5	4.8	46	135	2.6	1.4	29	5.6	30	56	1.8	61.3	13	4.7	1	1.2
HiSilicon Kirin 9000	13	25	2.6	1.9	22.4	3.3	2.2	4.8	47	132	7.2	2.4	8.9	5.6	35	65	8.5	332	8.8	4.9	2.7	3
Exynos 2100	10	17	2.2	4.1	25.4	4.2	4.6	4.7	44	119	4.1	1	38	5.7	31	61	5	49.1	7.8	4.7	1.9	2.8
MediaTek Dimensity 1200	9.7	17	1.8	1.2	36.6	3.7	2.9	4.8	42	131	8.3	0.9	28	6.1	35	58	4.4	46	11	4.9	2	2.1
MediaTek Dimensity 1100	11	19	2	1.3	36.6	3.9	3	4.8	46	134	9.1	0.9	30	6.1	40	66	4.6	46	12	4.9	2.1	2.3
HiSilicon Kirin 990 5G	13	24	4.3	3.1	25.1	3.9	2.8	4.8	52	128	13	1.2	12	5.6	43	74	16	52.9	11	4.9	4.4	4.7
MediaTek Dimensity 1000+	12	19	2.2	1.3	36.6	4.2	3	4.9	52	155	9.3	0.9	31	6	44	68	4.8	46	12	5	2.4	3
MediaTek Dimensity 1000L	14	22	2.6	1.5	36.6	4.8	3.5	4.9	59	162	10	0.9	32	6	51	79	5.5	46	14	5	2.8	3.3
HiSilicon Kirin 990	14	26	5.7	5.2	25.1	4.4	3.7	4.8	55	142	19	1.2	17	5.6	43	75	19	52.9	12	4.9	5.8	6.4
HiSilicon Kirin 985 5G	14	26	6	5.4	25.1	5.3	4.6	4.8	62	147	21	1.2	20	5.6	48	87	20	52.9	17	4.9	6.2	8.2
MediaTek Dimensity 820	13	24	2.9	1.6	36.6	5.9	4.7	4.8	58	141	15	0.9	50	6.1	49	87	6.3	46	19	4.9	3.1	3.5
Snapdragon 865 Plus	14	36	3.1	1.9	23.7	6.1	0	0	50	157	10	1.2	55	4.9	40	68	7.1	49.6	11	4.7	2.9	3.2
MediaTek Dimensity 800	17	27	2.8	1.4	36.6	5.8	4.4	4.8	69	153	16	0.9	57	6.1	63	97	5.9	46	19	4.9	3.1	3.4
Snapdragon 855 Plus	13	22	3.2	2	23.7	8.4	0	0	56	123	10	1.2	66	4.9	42	68	7.5	49.6	14	4.7	3.2	3.6
HiSilicon Kirin 990 5G	13	24	5.6	9.4	25.1	4.2	7.1	4.7	54	129	14	1.2	13	5.5	43	75	17	52.9	13	4.9	5.7	6.3
Snapdragon 865	13	22	3.4	2	23.7	9.6	0	0	60	162	11	1.2	65	4.9	43	78	7.4	49.6	15	4.7	3	3.3
Exynos 990	17	29	12	6.2	6.2	12	8	4.7	69	128	43	0	53	5.7	40	79	18	0	18	4.7	16	14
Snapdragon 855	14	23	3.6	2.1	23.7	9.3	0	0	57	137	11	1.2	75	4.9	47	76	7.8	49.6	16	4.7	3.4	3.6
Exynos 1080	14	31	18	8.1	6.2	19	12	4.7	57	128	53	0	62	5.7	43	74	32	0	24	4.7	12	23
Exynos 9825 Octa	23	59	12	7	6.2	11	7.4	4.7	116	163	45	0	53	5.7	43	114	18	0	18	4.7	10	14
Exynos 9820 Octa	33	74	15	8.7	6.2	15	10	4.7	114	167	52	0	59	5.7	44	121	22	0	22	4.7	13	17
HiSilicon Kirin 985	20	26	7.2	17	25.1	5.3	12	4.7	69	149	21	1.2	19	5.5	64	85	20	52.9	16	4.9	7.5	9.9
SDM855 + Pixel Neural Core	28	61	5.5	2.1	25.6	17	11	4.6	65	127	11	1.6	65	4.7	54	115	6.6	46.1	29	4.7	2.8	4.9
HiSilicon Kirin 820 5G	20	30	9.2	20	25.1	6.3	14	4.7	72	159	26	1.2	24	5.5	65	91	26	52.9	18	4.9	9.3	11
Snapdragon 765G	33	78	4.5	3.1	23.7	15	0	0	115	224	18	1.2	135	4.9	60	93	11	49.6	30	4.7	4.6	5.1
Snapdragon 845	47	91	6.1	4.1	23.7	11	0	0	151	222	33	1.2	93	4.9	132	164	15	49.6	21	4.7	5.8	6.3
HiSilicon Kirin 980	19	36	15	0	34	15	0	0	69	162	88	1.7	85	5.5	51	103	29	58.7	24	4.7	14	11
Snapdragon 720G	15	28	7.3	5.1	23.7	13	0	0	101	215	32	1.2	151	4.9	55	87	19	49.6	31	4.7	7.4	8.1
HiSilicon Kirin 810	33	68	11	25	25.1	9.2	20	4.7	127	244	37	1.2	34	5.5	75	120	30	52.9	24	4.9	10	12
Snapdragon 732G	34	73	5.8	4.3	23.7	14	0	0	139	241	34	1.2	144	4.9	76	109	16	49.6	31	4.7	5.9	7
Exynos 980	18	36	22	14	6.2	25	18	4.7	98	228	103	0	148	5.7	52	89	42	0	42	4.7	29	22
Snapdragon 730G	34	73	6	4.3	23.7	14	0	0	138	244	34	1.2	160	4.9	76	111	16	49.6	33	4.7	5.9	6.9
Snapdragon 730	34	75	6.1	4.3	23.7	15	0	0	139	243	34	1.2	160	4.9	76	111	16	49.6	33	4.7	5.8	6.9
Snapdragon 712	43	49	5.8	4.2	23.7	15	0	0	301	342	33	1.2	181	4.9	158	166	16	49.6	36	4.7	6.1	6.7
Snapdragon 675	34	73	5.6	3.8	23.7	22	0	0	141	253	24	1.2	325	4.9	77	114	14	49.6	57	4.7	5.5	6.3
Snapdragon 750G	14	65	9.2	5.2	23.9	18	13	4.7	105	248	36	2.8	136	5.5	51	89	19	82.3	47	4.8	8.9	10
MediaTek Dimensity 720	18	28	13	8	0	22	14	4.6	139	273	55	0	119	4.6	64	97	29	0	50	4.7	13	15

V5.0

5 min

Running any TensorFlow or PyTorch Model on Your Android Smartphone

Think That's Impossible?

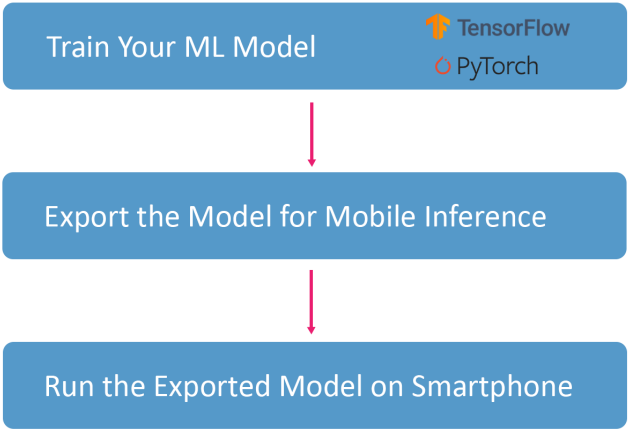
1. Pick your favorite deep learning model, *e.g.*:

   Inception
   MobileNet
   ResNet
   R-CNN
   Deeplab
   UNet
   Pix2Pix

2. Pick your own / favorite Android smartphone, *e.g.*:

   Samsung Galaxy
   Sony Xperia
   Huawei Mate
   OnePlus One
   Asus Zenfone
   Xiaomi Redmi
   Oppo Reno

Running AI Models on Mobile Devices: Overview



Export Your Model: TensorFlow (Keras)

```python
# Define and load your pre-trained Keras model

model = ...

# Export and save the model for mobile deployment

converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()

open("model.tflite", "wb").write(tflite_model)
```

Full example (UNet): https://github.com/aiff22/MAI-2021-Workshop/blob/main/keras_to_tflite.py

Export Your Model: TensorFlow (Plain)

```
# Define and load your pre-trained Keras model

session = ...

input_tensor = ...
output_tensor = ...

# Export and save the model for mobile deployment

converter = tf.compat.v1.lite.TFLiteConverter
            .from_session(session, [input_tensor], [output_tensor])

tflite_model = converter.convert()
open("model.tflite", "wb").write(tflite_model)
```

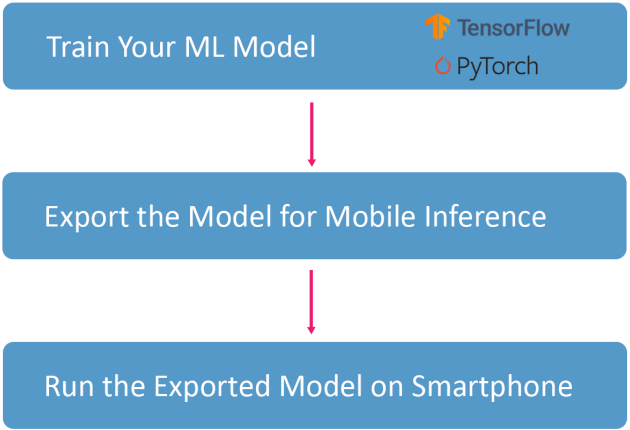
Full example (UNet): https://github.com/aiff22/MAI-2021-Workshop/blob/main/tensorflow_to_tflite.py

Export Your Model: PyTorch

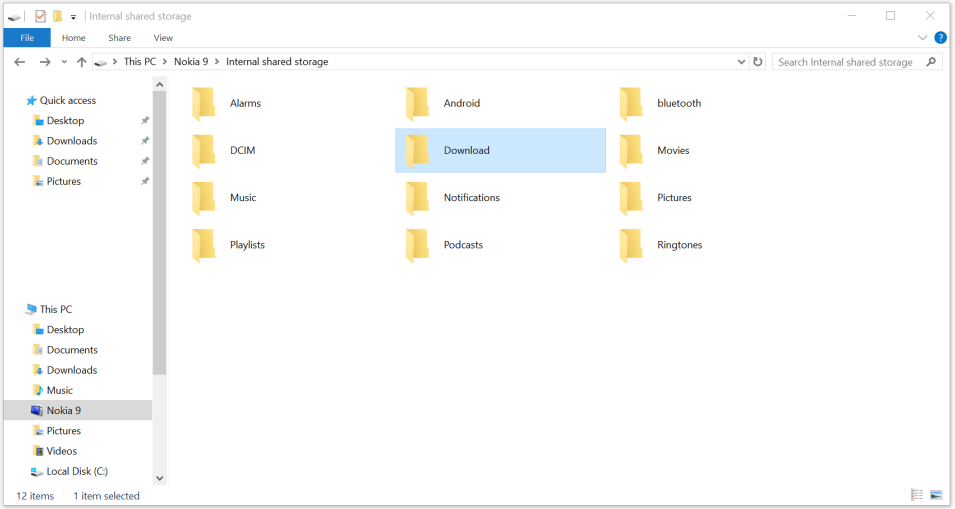
```python
# Define and load your pre-trained PyTorch model

model = ...

# Export model to ONNX

sample_input = torch.randn($batch_size, $depth, $height, $width)
torch.onnx.export(model, sample_input, "model.onnx", export_params=True,
                  input_names=['input'], output_names=['output'])

# Load ONNX model and convert it to TF format

onnx_model = onnx.load("model.onnx")
output = prepare(onnx_model)
output.export_graph("tf_model/")

# Export and save the model for mobile deployment

converter = tf.lite.TFLiteConverter.from_saved_model("tf_model")
tflite_model = converter.convert()

open("model.tflite", "wb").write(tflite_model)
```

Full example (UNet): https://github.com/aiff22/MAI-2021-Workshop/blob/main/pytorch_to_tflite.py

Running AI Models on Mobile Devices: Overview



Copy model.tflite File to Your Phone Storage



Install AI Benchmark Application: https://ai-benchmark.com/download

AI Benchmark Mobile

Version: 4.0.4

Is your smartphone capable of running the latest Deep Neural Networks to perform these AI-based tasks? Does it have a dedicated AI Chip? Is it fast enough? Run AI Benchmark to professionally evaluate its AI Performance!

Direct APK Download

Download from Google Play

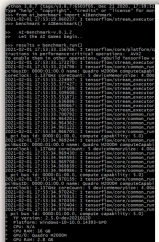
AI Benchmark Mobile Lite

Version: 4.0.4 Lite

Designed for legacy devices running Android 4.x or having less than 1Gb of RAM. Contains a reduced number of tests, supports CPU-based inference only. The scores are calibrated to be comparable with the results of the full version.

Direct APK Download

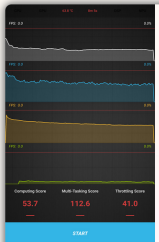
Download from Google Play



AI Benchmark Alpha

Version: 0.1.2

AI Benchmark Alpha is an open source python library for evaluating AI performance of various hardware platforms, including CPUs, GPUs and TPUs. The benchmark is relying on TensorFlow machine learning library, and is providing a precise and lightweight solution for assessing inference and training speed for key ML models.

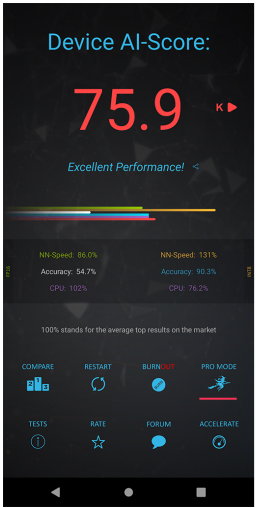


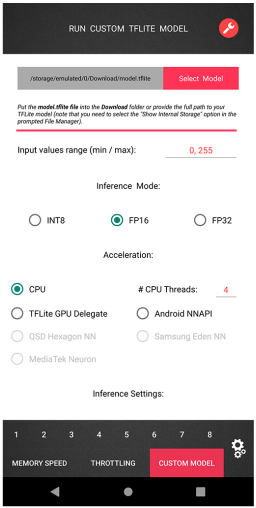
Burnout Benchmark

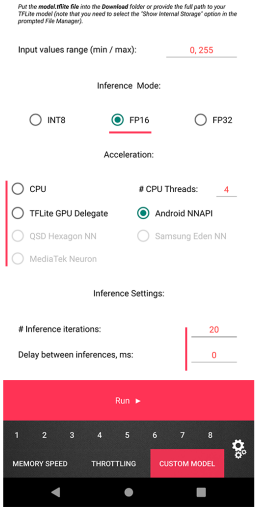
Version: 1.0.1

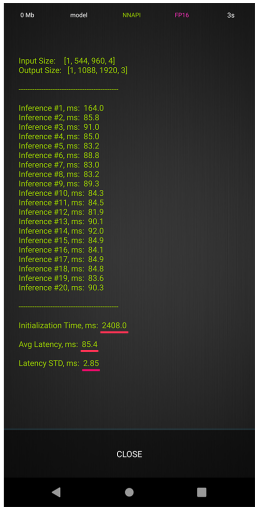
Burnout Benchmark is a professional tool for an extensive performance evaluation of all chipset components. It pushes the device to its limits by loading CPU, GPU, NPU and DSP by 100% separately or simultaneously to assess the real-world device performance under heavy load and check its thermal throttling behavior over time.

Run the Model on Your Smartphone with Various Acceleration Options









Think That's Impossible?

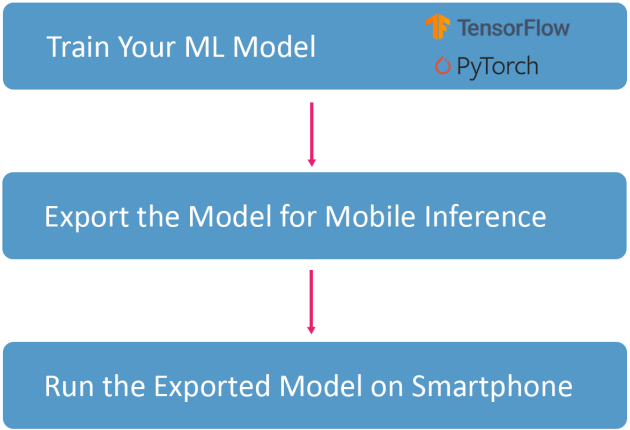
1. Pick your favorite deep learning model, *e.g.*:

   Inception
   MobileNet
   ResNet
   R-CNN
   Deeplab
   UNet
   Pix2Pix

2. Pick your own / favorite Android smartphone, *e.g.*:

   Samsung Galaxy
   Sony Xperia
   Huawei Mate
   OnePlus One
   Asus Zenfone
   Xiaomi Redmi
   Oppo Reno

Running AI Models on Mobile Devices: Overview



WRAP UP

AI Benchmark: Research Papers

- AI Benchmark: All About Deep Learning on Smartphones in 2019

  Andrey Ignatov (ETH Zurich), Radu Timofte (ETH Zurich), Andrei Kulik (Google Research), Seungsoo Yang (Samsung, Inc), Ke Wang (Huawei, Inc.), Felix Baum (Qualcomm, Inc.), Max Wu (MediaTek, Inc.), Lirong Xu (Unisoc, Inc.), Luc Van Gool (ETH Zurich)

- AI Benchmark: Running Deep Neural Networks on Android Smartphones

  Andrey Ignatov (ETH Zurich), Radu Timofte (ETH Zurich), William Chou (Qualcomm, Inc.), Ke Wang (Huawei, Inc.), Max Wu (MediaTek, Inc.), Tim Hartley (Arm, Inc.), Luc Van Gool (ETH)

- AI Benchmark 2021: TBA

Mobile AI Workshop 2021

Designing the Next-Generation AI Models

for Edge and Mobile Devices...

20 June, 7:00 Pacific Time

CVPR 2021

https://ai-benchmark.com/workshops/mai/2021/

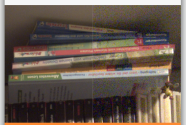
Mobile AI 2021 Workshop – Competitions

Learned Smartphone ISP

Evaluation Platform:
MediaTek Dimensity APU

Powered by: MEDIATEK

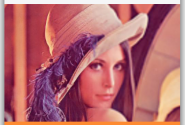
Image Denoising



Evaluation Platform:
Exynos Mali GPU

Powered by: SAMSUNG

Image Super-Resolution



Evaluation Platform:
Synaptics Dolphin NPU

Powered by: Synaptics

Video Super-Resolution

Evaluation Platform:
Snapdragon Adreno GPU

Powered by: oppo

Depth Estimation

Evaluation Platform:
Raspberry Pi 4

Powered by: Raspberry Pi

Camera Scene Detection

Evaluation Platform:
Apple Bionic

Powered by: CVL

Mobile AI 2021 Workshop – Vendor Presentations

- MediaTek:

  Edge AI Technology – from Development to Deployment

- Imagination:

  Imagination Technologies Approach to Overcame the Challenges of Deploying AI in Mobile

- Samsung:

  Samsung Exynos Mobile NPUs and SDK: Hardware Design, Performance, Models Deployment and Efficient Inference

- Google:

  Android Neural Networks API - What's New and Best Practices

- Synaptics:

  AI on the Edge @Synaptics : HW and SW Products and Development

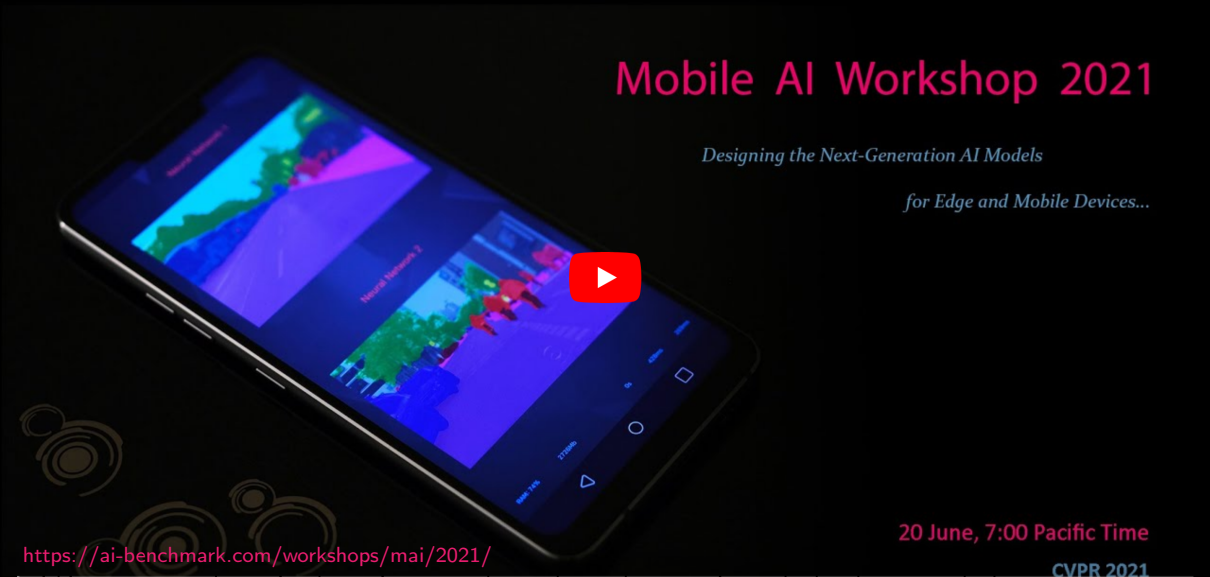
- Huawei:

  AI Deployment from Hardware to Software – Challenges and Opportunities

- Qualcomm:

  Hate it or Love it, Your SW Stack Defines Application Performance and Reach: An Overview or Our HW and SW

Mobile AI Workshop 2021: Deep Learning on Mobile Devices, Live Stream



0:00 / 9:53:30 • Streaming started ›

Scroll for details

Mobile AI Workshop 2022

Designing the Next-Generation AI Models for Mobile Devices...

June 2022 Virtual, US

https://ai-benchmark.org/

Over the past years, mobile AI-based applications are becoming more and more ubiquitous. Various deep learning models can now be found on any mobile device, starting from smartphones running portrait segmentation, image enhancement, face recognition and natural language processing models, to smart-TV boards coming with sophisticated image super-resolution algorithms. The performance of mobile NPUs and DSPs is also increasing dramatically, making it possible to run complex deep learning models and to achieve fast runtime in the majority of tasks.

While many research works targeted at efficient deep learning models have been proposed recently, the evaluation of the obtained solutions is usually happening on desktop CPUs and GPUs, making it nearly impossible to estimate the actual inference time and memory consumption on real mobile hardware. To address this problem, we introduce the first Mobile AI Workshop, where all deep learning solutions are developed for and evaluated on mobile devices.

Due to the performance of the last-generation mobile AI hardware, the topics considered in this workshop will go beyond the simple classification tasks, and will include such challenging problems as *image denoising, HDR photography, accurate depth*

AI Benchmark Forum

https://ai-benchmark.net/

Official Announcements

	Threads	Messages	
News, Releases and More	6	6	CVPR Mobile AI Workshop @ Live ... Tuesday at 12:41 PM · Andrey Ignatov

AI Benchmark Mobile

Benchmarking Android, iOS, Sailfish OS and other mobile and IoT devices

	Threads	Messages	
General AI Benchmark Questions	3	10	much higher latencies with NNAPI ... Tuesday at 11:10 AM · Andrey Ignatov
AI Benchmark Results	0	0	None
Crashes & Issues	4	19	Higher latencies with NNAPI on Sn... Apr 22, 2021 · anamika
Other Topics	0	0	None

AI Benchmark Alpha

Benchmarking Desktops, Servers, Laptops and other general-purpose ML hardware

	Threads	Messages	
General AI Benchmark Questions	2	7	Why no Nvidia NVIDIA RTX A6000 ... Mar 28, 2021 · gbolcer
AI Benchmark Results	0	0	None
Choosing Hardware for Deep Learning	0	0	None

Members online

No members online now.

Total: 4 (members: 0, guests: 4)

Latest posts

- Live @ MAI Workshop - Leave your questions to the speakers in this thread!
  Latest: Andrey Ignatov · Yesterday at 2:12 PM
  Mobile AI Workshop 2021
- CVPR Mobile AI Workshop @ Live on YouTube
  Latest: Andrey Ignatov · Tuesday at 12:41 PM
  News, Releases and More
- High Dynamic Range Challenge
  Latest: Andrey Ignatov · Tuesday at 11:15 AM
  Mobile AI Workshop 2021
- much higher latencies with NNAPI on Snapdragon 888
  Latest: Andrey Ignatov · Tuesday at 11:10 AM
  General AI Benchmark Questions
- NNAPI usage on Exynos 2100 devices
  Latest: Andrey Ignatov · May 27,

Thank you for your attention!

Project Website

Forum

Google Play

Desktop Version