SAMSUNG SDS

Foresee

Techtonic 2021

Partner

Disrupt

딥러닝 경량화와 최적화 기술을 통한 Embedded AI 구현

Embedded Deep Learning

손창용 전문연구원

삼성전자 종합기술원

AGENDA

1) Embedded 딥러닝 S/W 기술 동향

2) 딥러닝 경량화 연구

3) Inference S/W 최적화 구현

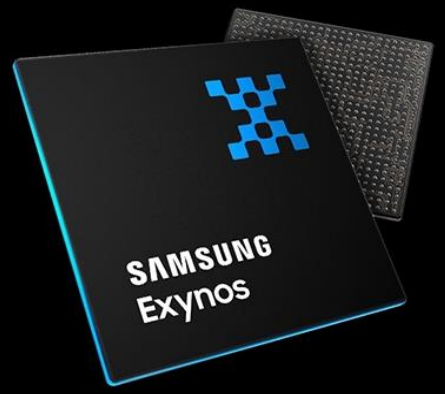
4) AI Software 프레임워크 개발

① Prospects for Embedded Deep Learning (1/4)

AI on the Edge

- AI Computing is spreading from server to edge for wider application

On-Server AI | On-Device AI | On-Sensor AI | On-Things AI



Cloud Services

- 수십 Giga Byte 메모리
- 수 Tera 컴퓨팅

Mobile AP

- 수십 Mega Byte 메모리
- 수 Giga 컴퓨팅

Image Sensor

- 수백 Kilo Byte 메모리
- 수백 Mega 컴퓨팅

Internet of Things

- 수십 Kilo Byte 메모리
- 수십 Mega 컴퓨팅

① Prospects for Embedded Deep Learning (2/4)

Why On-Device AI?

- Cost-saving on cloud services
  : Screen unlock about 100 times a day

- Privacy protection and locality
  : Store & process biometric data within a device
  : Language support depends on each user

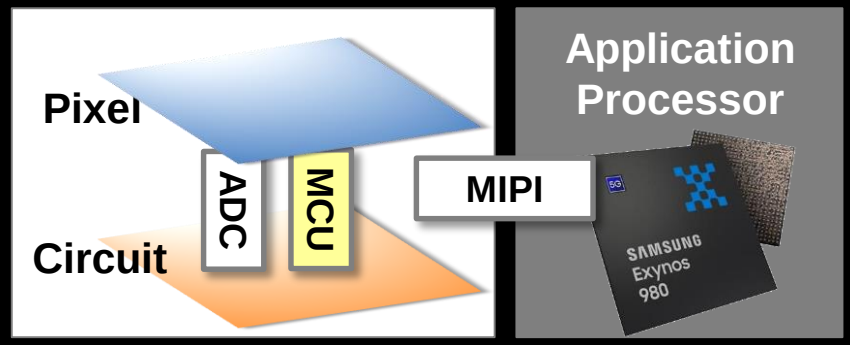
- Rapid response
  : Faster than human perception (about 150 msec)

[On-Device AI의 장점]

① Prospects for Embedded Deep Learning (3/4)

Why On-Sensor AI?

[Traditional Image Sensor]



- Reduced power consumption
  : Sensor + AP (150mW~1.5W)
    → Sensor only (10~50mW)

- Extremely low latency
  : On-sensor processing enables lower latency than AP processing

- Higher security and privacy
  : Instantaneous processing on sensor reduces security risks and privacy concerns

- Advantageous raw image processing
  : Leverage inherent sensor structure and access to the raw image without ISP

① Prospects for Embedded Deep Learning (4/4)

Comparison On-Device vs On-Sensor

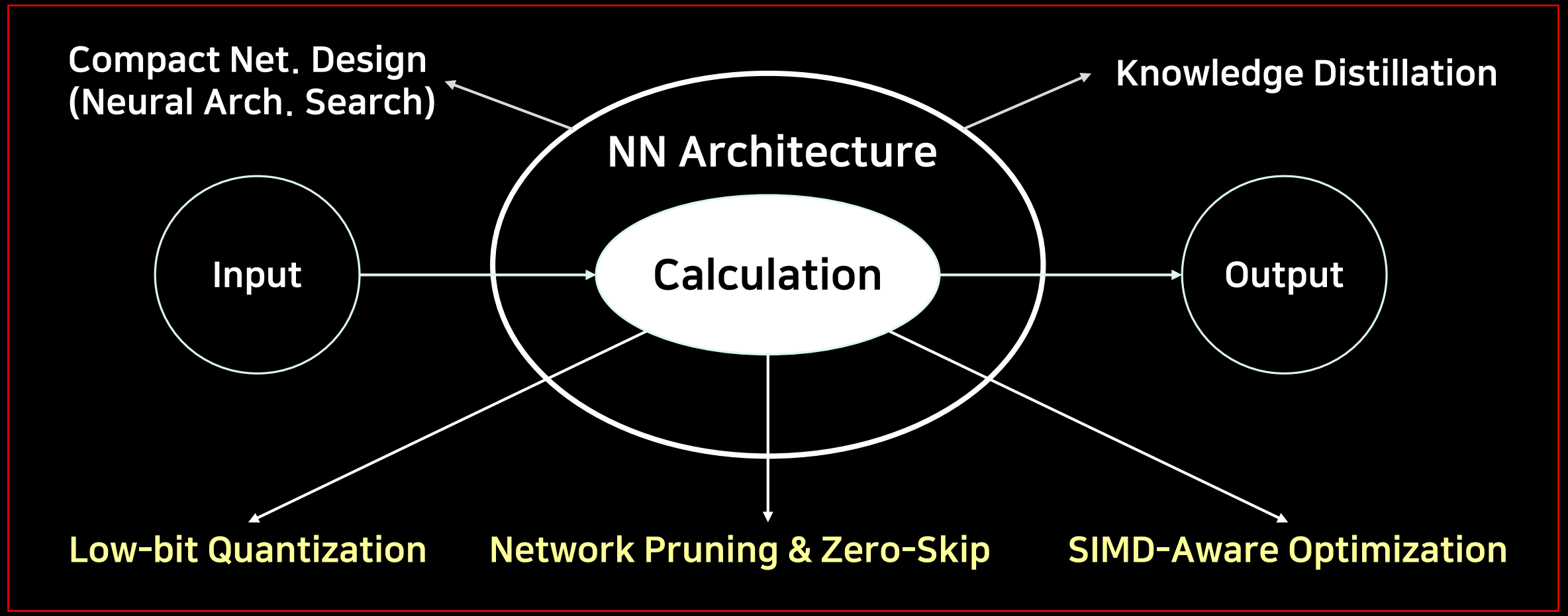
- MCUs have less memory and computing compared to mobile APs

	On-Device	On-Sensor
Processor	Galaxy S20 AP	Cortex-M7 MCU
Runtime Memory	8 GB	< 512 KB
Storage	128 GB	< 2 MB
Power	~ 8 Watt	0.3 Watt
Neural Net	MobileNetV2 (8-bit Quantization)	MCUNet*
Peak Memory	1.7 MB	< 256 KB
Model (Weights)	3.4 MB	< 2 MB
MACs	300 M	< 60 M

"MCUNet: Tiny Deep Learning on IoT Devices", NeurIPS 2020

① Technology Trends of Embedded Deep Learning

Lightweight & Optimization for Embedded AI Software



① Need for Hardware-Aware Neural Network

Compact network design considering actual energy consumption

- SqueezeNet: AlexNet-level Accuracy with 50x fewer parameters

  → But Energy required for inference operation ?

AlexNet 대비 SqueezeNet 성능	
Accuracy (Top-1 ImageNet)	0.3% ↑
# of Weights	51.8x ↓
# of Layers	2.3x ↑
Normalized Energy	1.3x ↑

- A regular 3x3 convolution has more compute (MACs) than an depthwise-separable convolution

  → But It executes faster on Edge TPU due to ~3x more effective hardware utilization

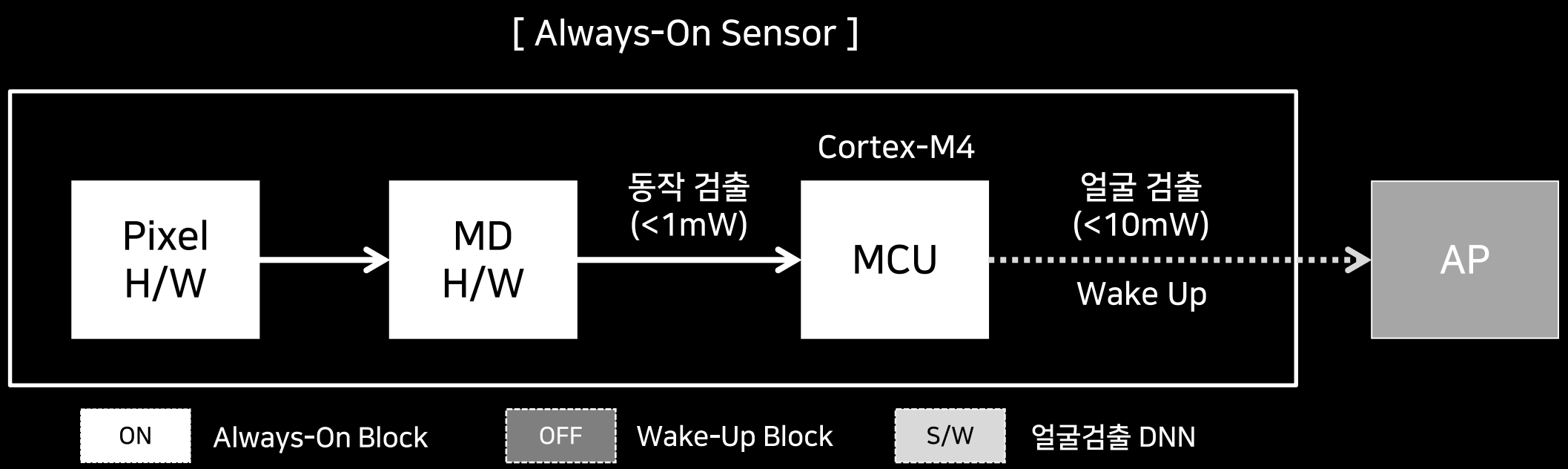
Neural Network	Accuracy (Top-1 ImageNet)	CPU (@Pixel 4)	Edge TPU (@Pixel 4)
MobileNetV2	73.7%	17.7ms	4.3ms
MobileNetV3 large	73.9%	10.0ms	3.7ms
MobileNetEdgeTPU	73.5%	13.8ms	3.1ms

② Compact Network Design for On-sensor AI (1/2)

저전력 Sensor향 AI 기술의 속도 / 메모리 요구 수준

- CIS 센서 內 사용 가능한 SRAM 100KB 이하 제약, MCU에서 실시간 동작하는 얼굴 검출 솔루션 요구

  → 단말기에서 구동하는 상용 얼굴 검출 솔루션 대비 SRAM 10배↓, Computing 20배↓ 절감 필요

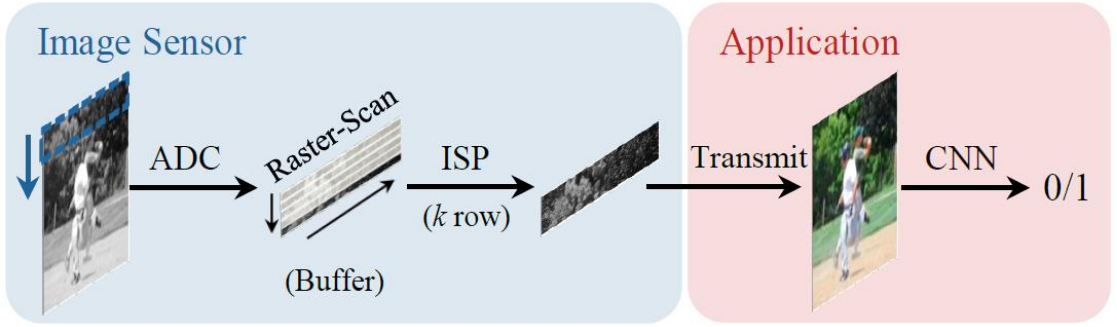


② Compact Network Design for On-sensor AI (2/2)

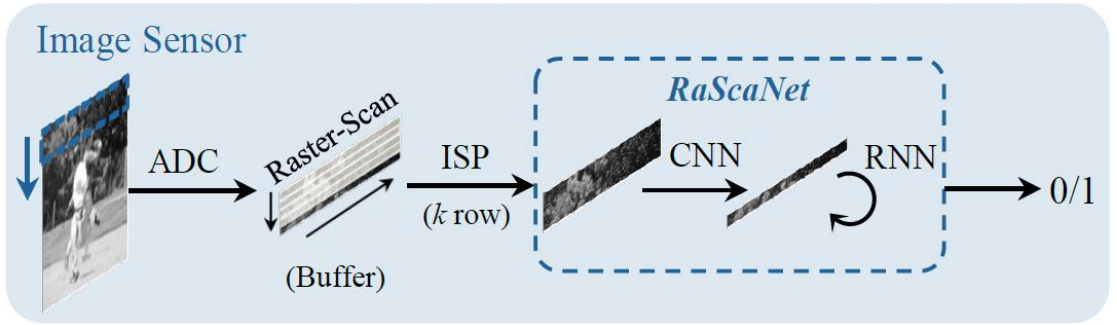
Memory Constraint Neural Network (Raster-Scanning Network) 개발 (SAIT)

- Human existence is detected by On-Sensor processing (Wake up Application Processor)
  → Peak / Weight memory (8KB / 50KB) reduced dramatically (~24× / ~14×)

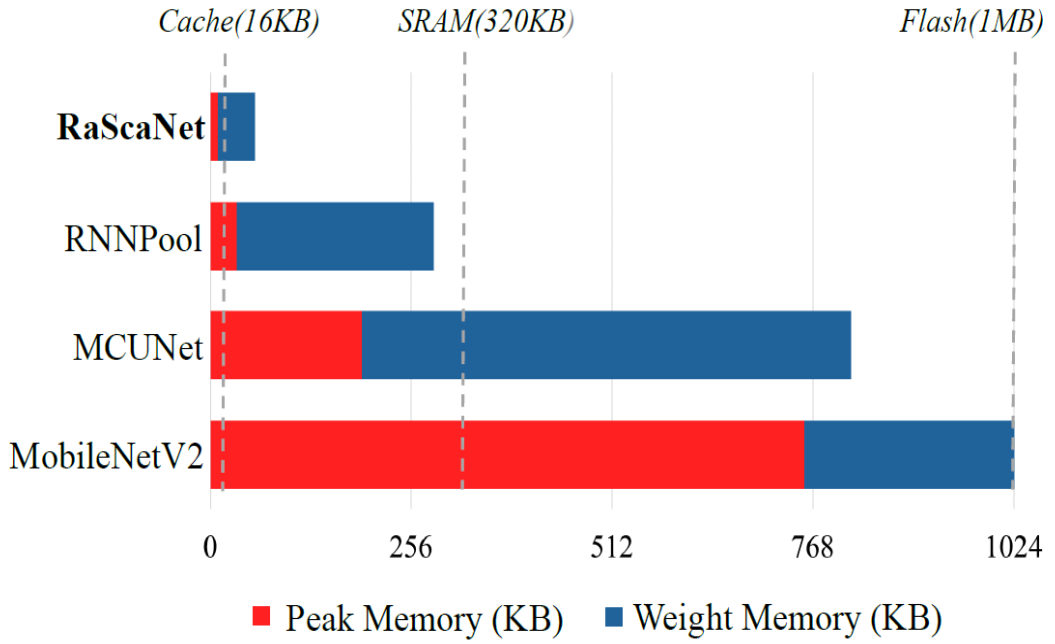
[Conventional]



[Proposed: CIS specialized DNN]



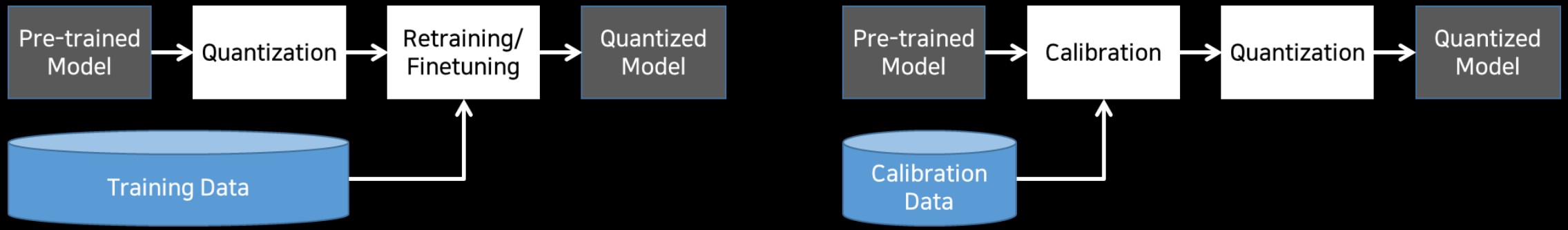
[Tiny models과 Memory usage 비교]



"RaScaNet: Learning Tiny Models by Raster-Scanning Images", CVPR 2021

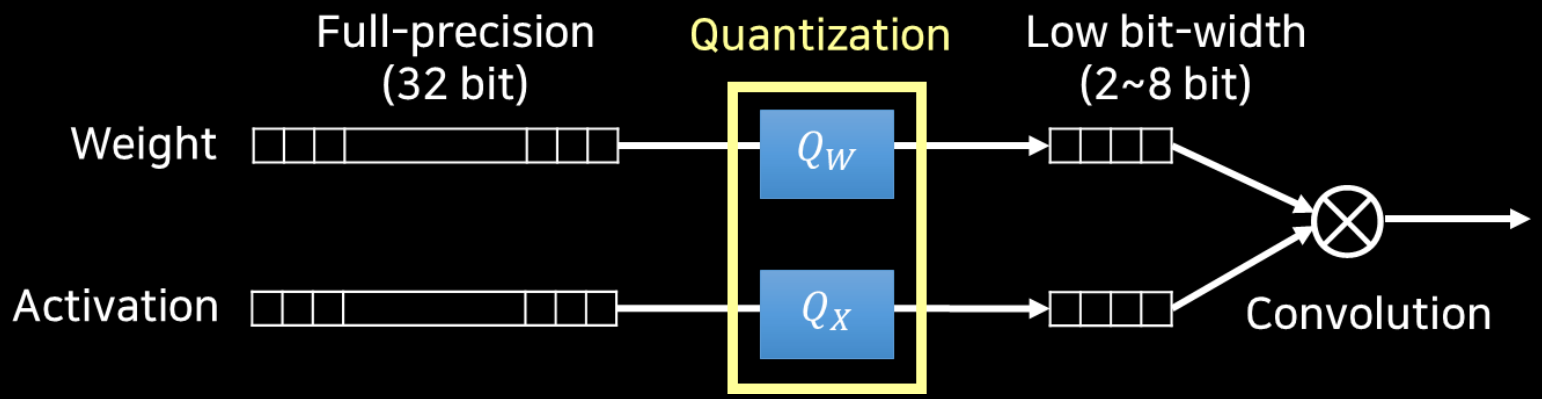
② Low-bit Network Quantization (1/3)

Quantization-Aware Training (QAT) vs. Post-Training Quantization (PTQ) quantization



Mixed-Precision Quantization (@Nvidia Tesla T4 GPU)

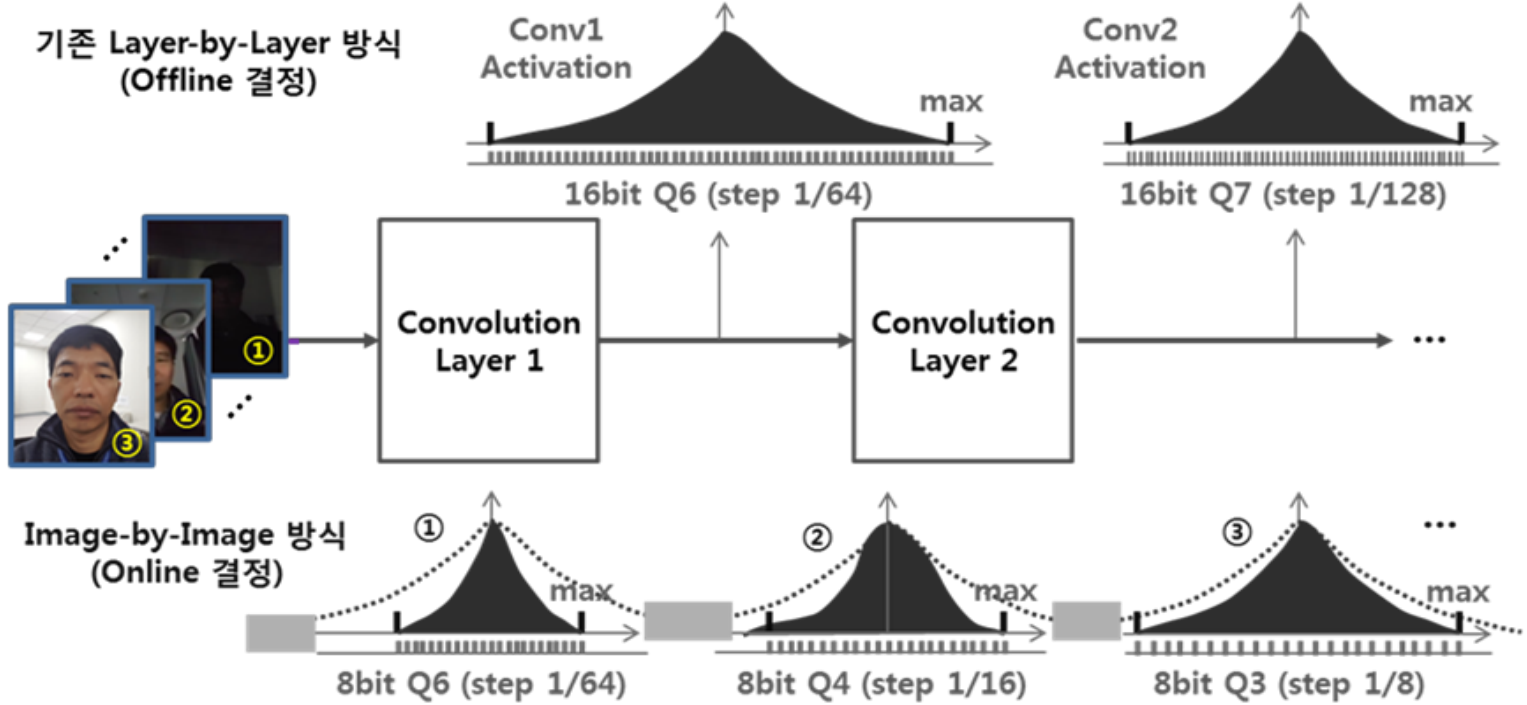
- 50% speed up with mixed-precision (INT4/INT8) quantization as compared to INT8 quantization

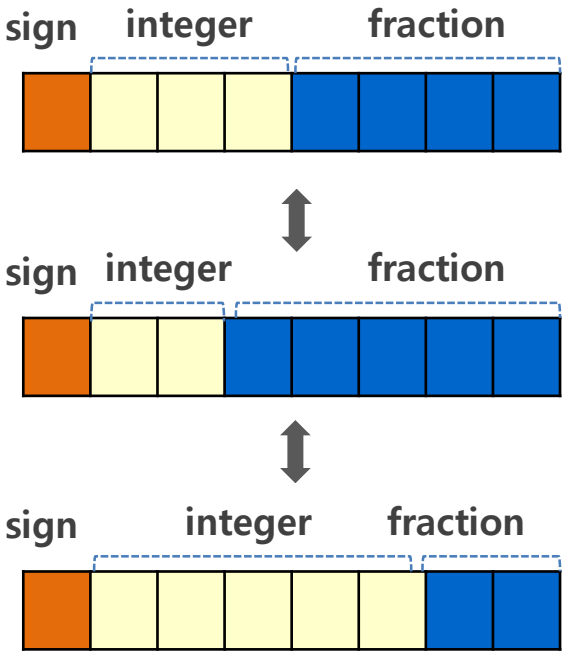


② Low-bit Network Quantization (2/3)

非학습 동적 양자화 기술 (SAIT)

- Full-Precision (32bit) 대비 정확도 성능 저하 없는 8X8 bit Convolution 연산 처리 구현
- 입력 영상의 조도 변화에 따라서 Q-Format (Fraction bit 수)을 변경하는 동적 양자화 기술

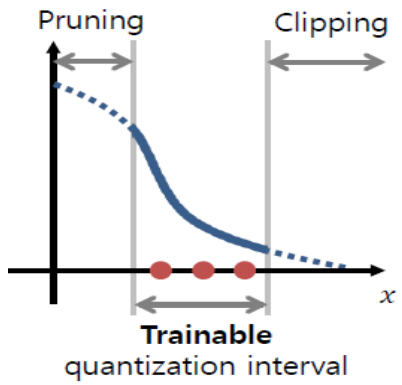


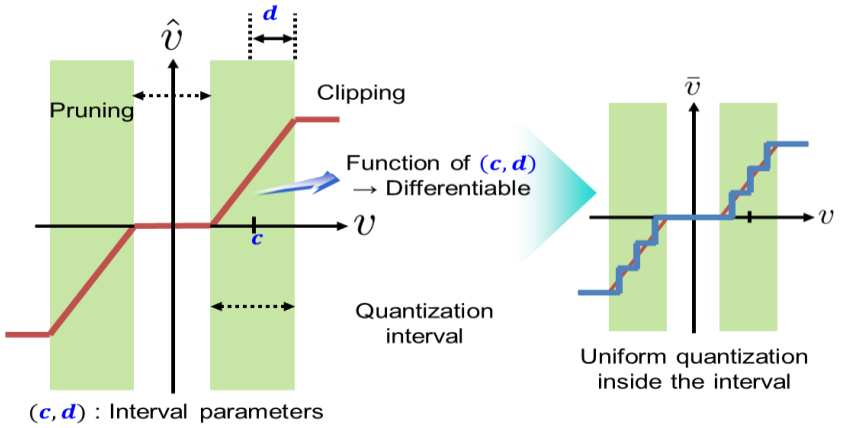


② Low-bit Network Quantization (3/3)

뉴럴넷 양자화 구간 (QIL: Quantization Interval Learning) 학습 기술 (SAIT)

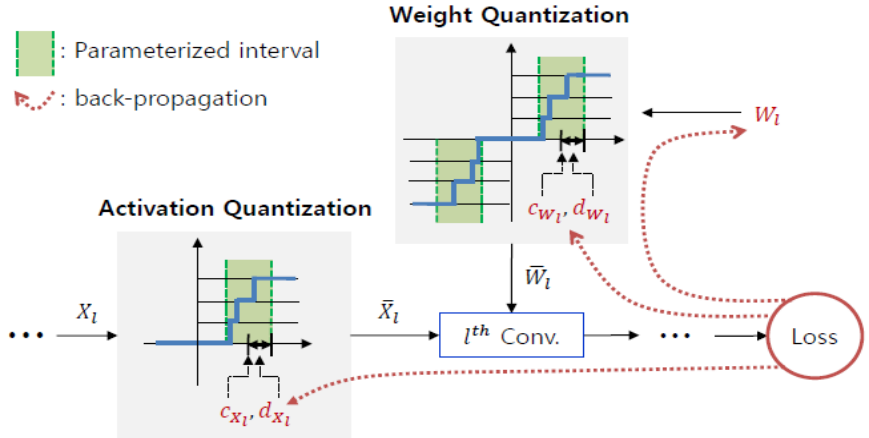
- Pruning과 Clipping의 양자화 변수와 Weight를 동시에 학습하는 양자화 구간 학습 방식

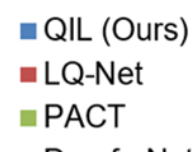


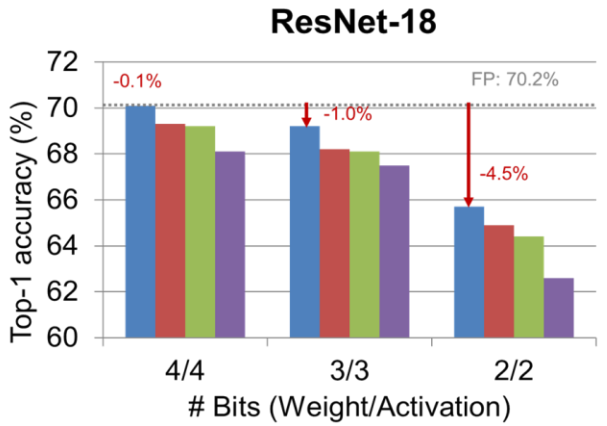


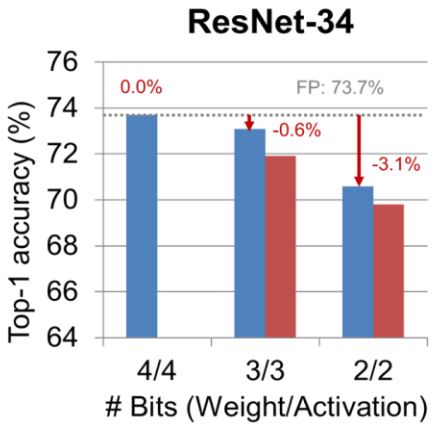
*Pruning/Clipping/Trainable

Low Bit-width Network	Weight P/C/T*	Activation P/C/T
Deep Comp. (Stanford, 2016)	✓ / × / ×	× / × / ×
DoReFa-Net (Megvii, 2016)	× / × / ×	× / ✓ / ×
QNN (Montreal, 2016)	× / ✓ / ×	× / ✓ / ×
HWGQ (MS, 2017)	× / × / ×	× / ✓ / ×
PACT(IBM, 2018)	× / × / ×	× / ✓ / ✓
Ours	✓ / ✓ / ✓	✓ / ✓ / ✓









"Learning to Quantize Deep Networks by Optimizing Quantization Intervals with Task Loss", CVPR 2019

② Pruning in Convolution Layers (1/3)

Zero에 가까운 값을 갖는 weight을 zero로 변경 후 재-학습으로 정확도 복원

- 인간 뇌의 synapse 수도 유사하게 변화

Age	Number of Connection	Stage
At birth	50조	Newly formed
1 yr	1,000조	Peak
10 yr	500조	Pruned and stabilized

- Weight의 값이 zero일 경우 해당 연산을 skip 가능

Original Kernel

-0.01	0.26	-0.05
0.07	-0.10	0.10
-0.01	0.01	0.01

Error Rate = 1.19%

Pruned Kernel

0	0.26	0
0	-0.10	0.10
0	0	0

Error Rate = 3.73%

Retrained Kernel

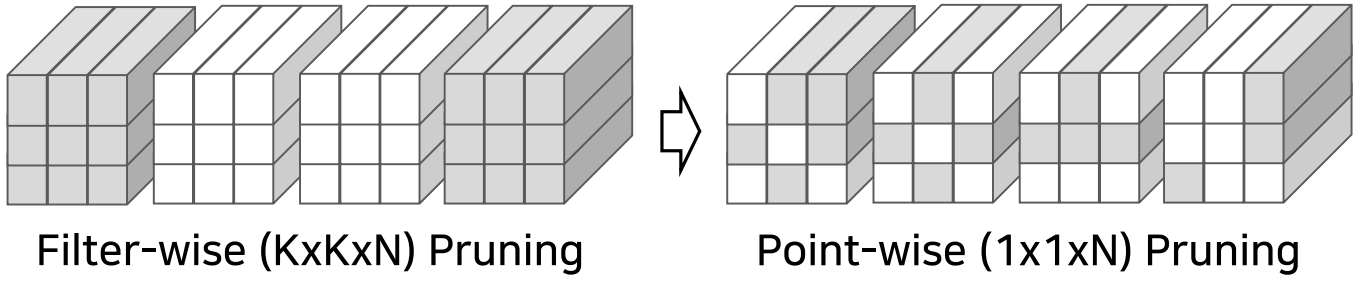
0	0.28	0
0	-0.10	0.12
0	0	0

Error Rate = 1.21%

② Pruning in Convolution Layers (2/3)

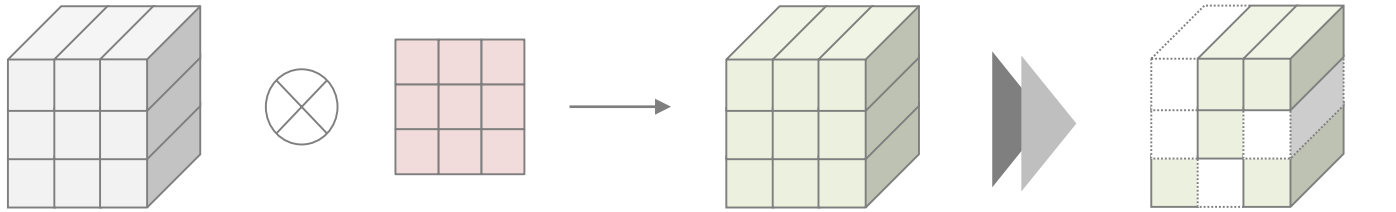
Point-wise (Shape-wise) Pruning 기술 (SAIT)

- 중요도가 낮은 네트워크 연결을 1x1xN 단위로 제거하는 Point-wise (Shape-wise) Pruning 방법
  → 기존 Channel (Filter-wise) Pruning 보다 30% 이상 효율적인 제거 가능



Filter-wise (KxKxN) Pruning

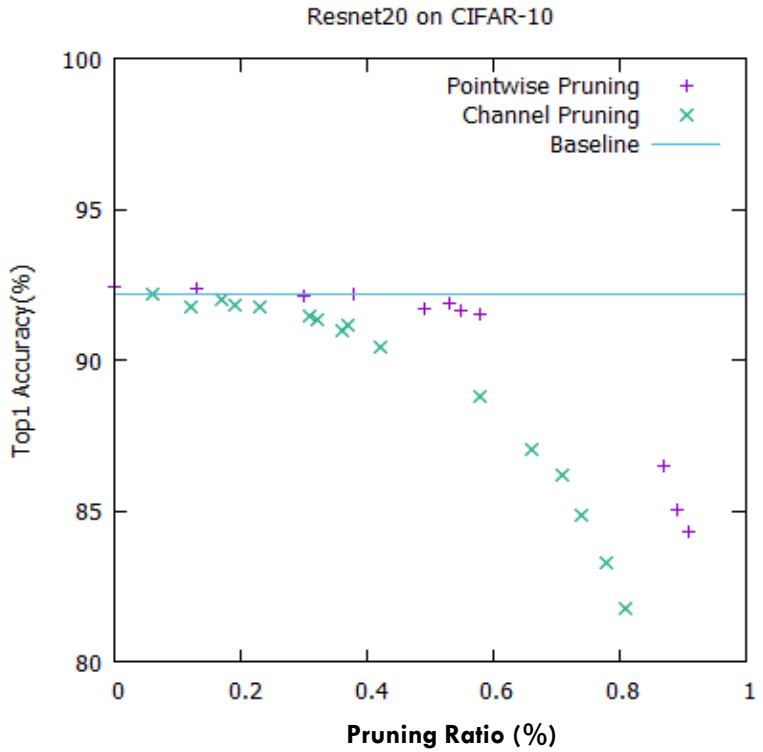
Point-wise (1x1xN) Pruning



Weight

Scale matrix
: Learning w/ Sparsity-inducing Penalty

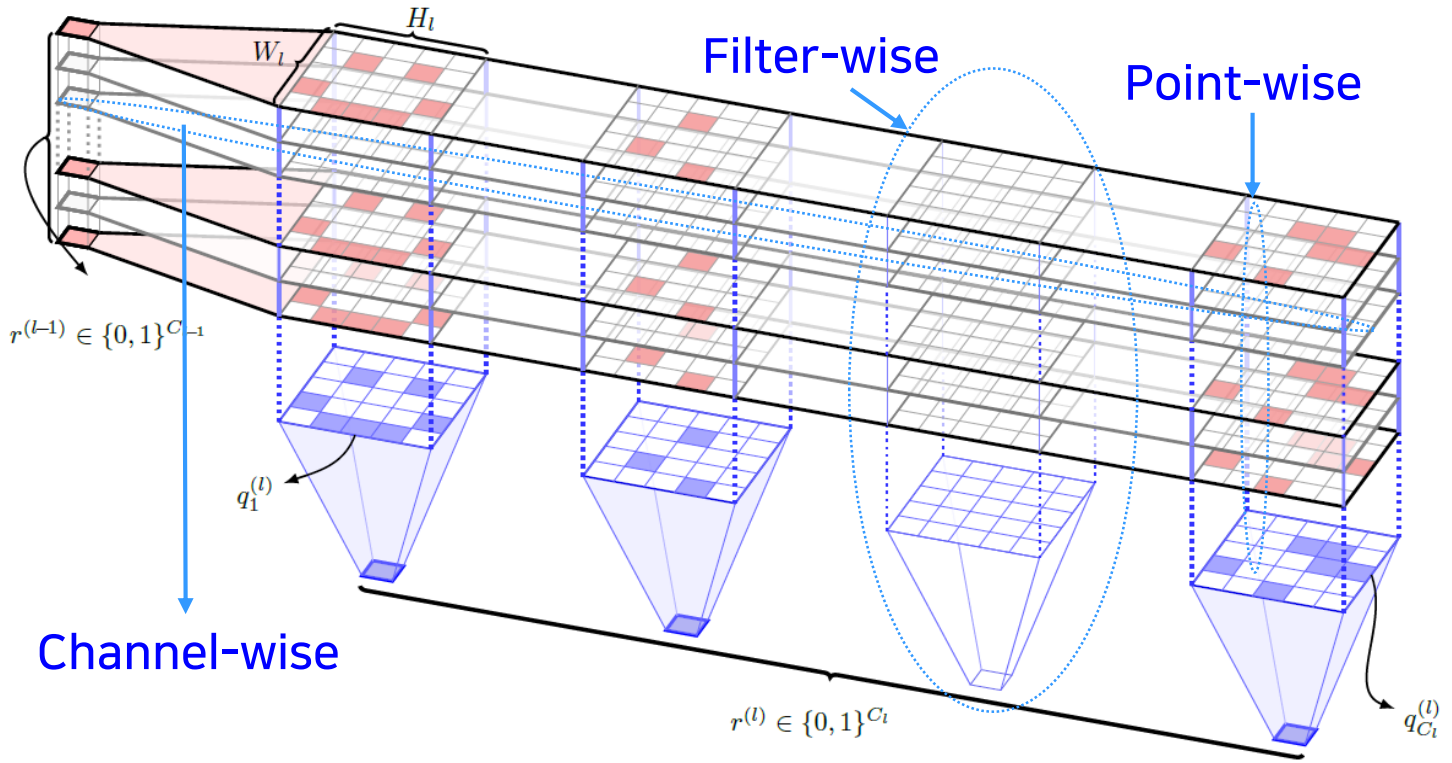
Conv. Weight

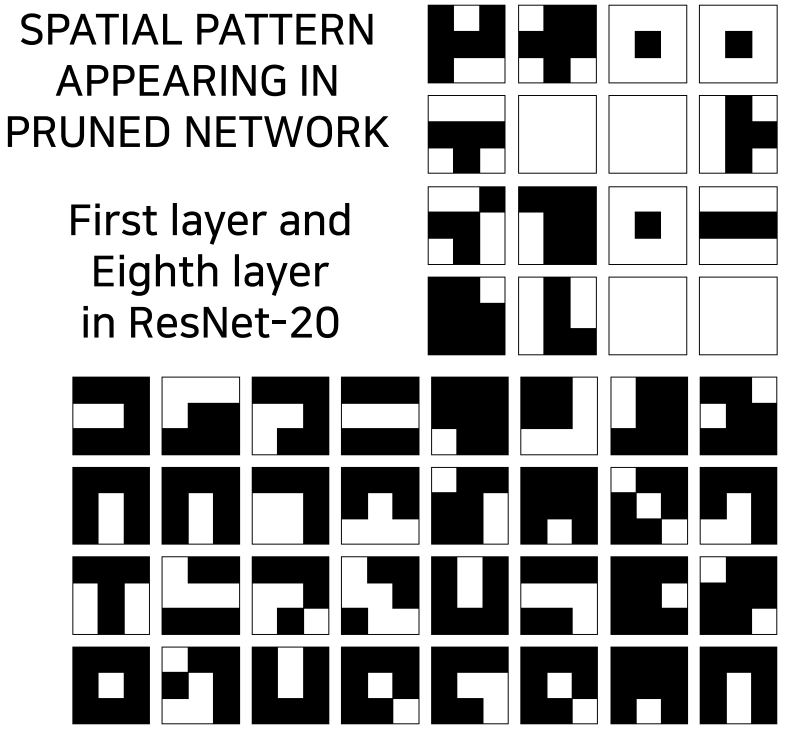


② Pruning in Convolution Layers (3/3)

Spatial Pruning 기술 (SNU/SAIT)

- Jointly pruning channel and spatial filters (channel-wise + filter-wise + point-wise)
  → 동일한 Pruning 비율 기준 기존 SOTA 대비 높은 정확도 성능 제공 (1~3% ↑ @ ResNet-18/50)





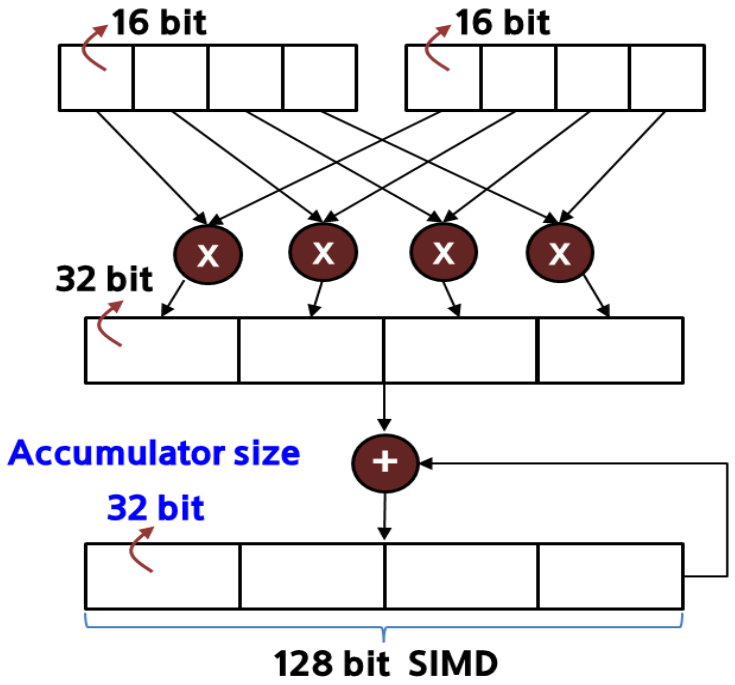
https://openreview.net/forum?id=IkYEJ5Cps5H

③ Optimization and Acceleration

SIMD (Single Instruction Multiple Data)-supporting Accelerator

- ARM Neon : 128-bit 단일 명령 다중 데이터 처리
- X86 (intel) SSE : 128-bit SIMD, AVX (Advanced Vector Extensions) 256-bit ~ 512-bit SIMD

MAC (Multiplication and Accumulation)



In a 128-bit SIMD operation

Four 32-bit accumulators
Can accommodate
Four 16-bit Activations and
Four 16-bit Weights
➡ **4-way MAC**

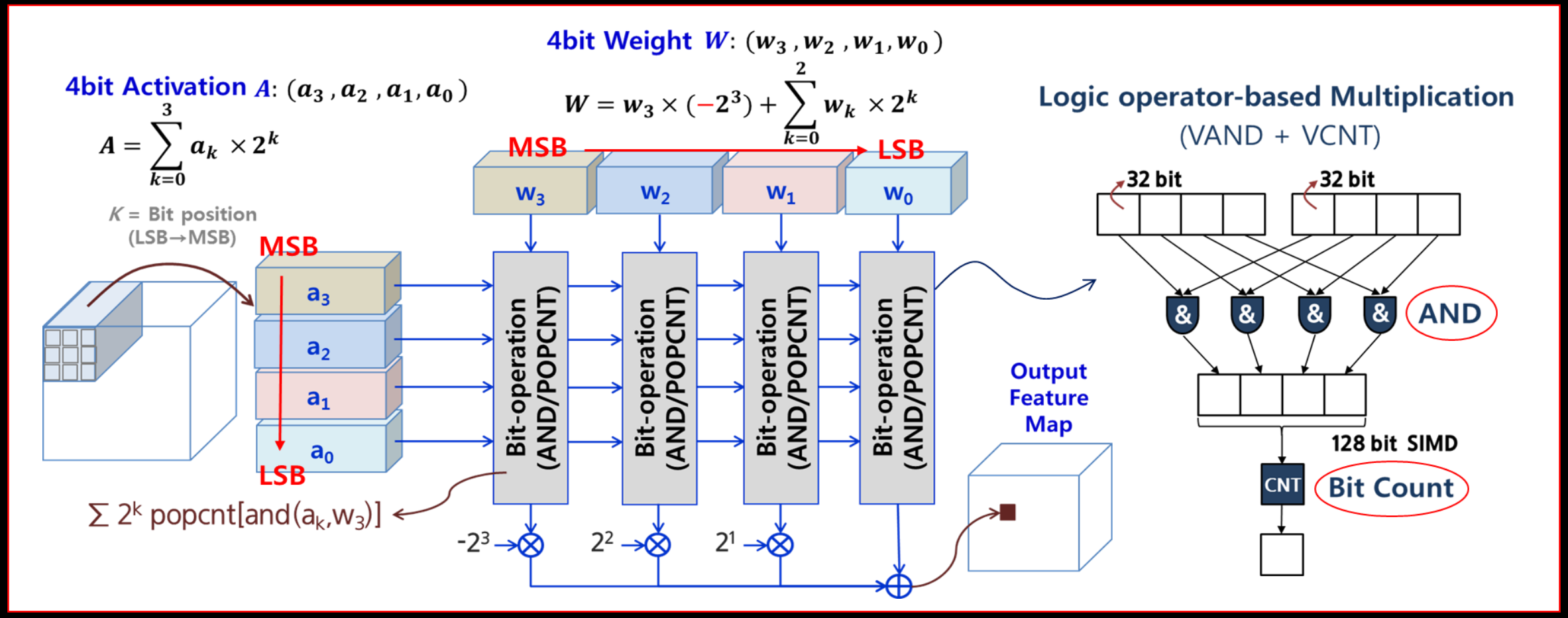
Sixteen 8-bit accumulators
Can accommodate
Sixteen 4-bit Activations and
Sixteen 4-bit Weights
➡ **16-way MAC**

For existing CPUs and DSPs
8-bit is the minimum-size container.
So, 4-bit data is perfect to speed up!

③ Efficient Convolution using Logical Operations

MAC → Bitwise Operation AND / POPCount (VAND, VCNT @ ARM-Neon)

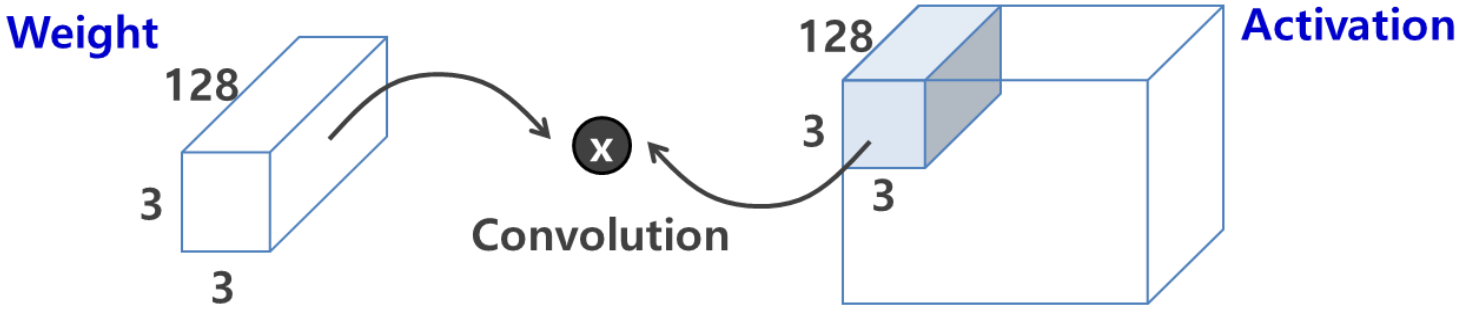
- Data를 Bit 단위 Slicing → bit position 별로 저장 → Logic 연산 처리 (VAND + VCNT)



③ Optimization using MAC vs. AND/COUNT

If Weight/Activation are less than 4-bit, AND/CNT becomes more efficient

Computational Cost for Single Output @ 128-bit SIMD



W/A	16/16bit	8/8bit	4/4bit	4/4bit	3/3bit	2/2bit
No. of Operations	**288** (9x32) **MAC** (4-way†)	**144** (9x16) **MAC** (8-way†)	**72** (9x8) **MAC** (16-way†)	**144** (9x16*) **AND/CNT**	**81** (9x9*) **AND/CNT**	**36** (9x4*) **AND/CNT**
No. of Data Loads	**288** (144x2) 16bit (8-way)	**144** (72x2) 8bit (16-way)	**144** (72x2) 8bit (16-way)	**72** (9x4x2) 1bit (128-way)	**54** (9x3x2) 1bit (128-way)	**36** (9x2x2) 1bit (128-way)

† Depending on accumulator size (16x16bit input → 32bit)

* Number of inner product for the bitwise operation

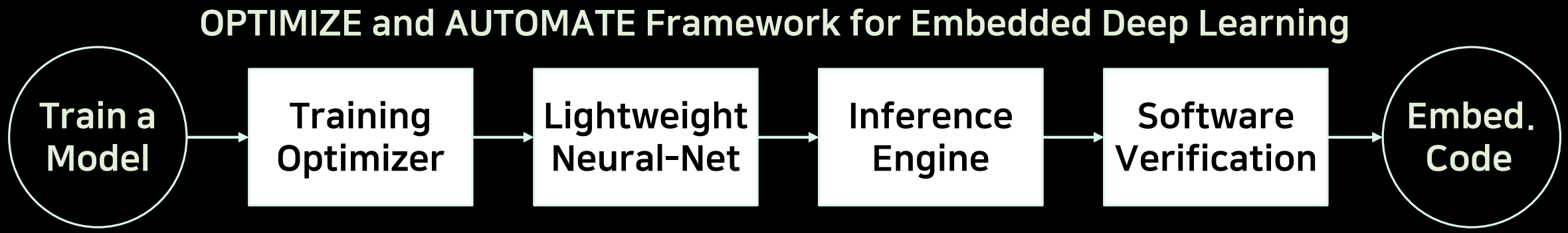
④ Automatic AI Software Framework (1/3)

딥러닝 경량화 / 고속화 기술의 Fully Automation化 및 Framework化 (SAIT)

- 경량화 모델 개발에서 Embedded 코드 배포까지 손쉽고 신속하게 해결하는 AI S/W Framework 개발
  → 사용자의 개입없이 개발 단계를 모두 자동화하여 혁신적인 개발 기간 단축 및 Agile SW 배포 환경 제공

SOLVE (Samsung Optimized Lightweight & Verification Engine)에서 제공하는 주요 기술

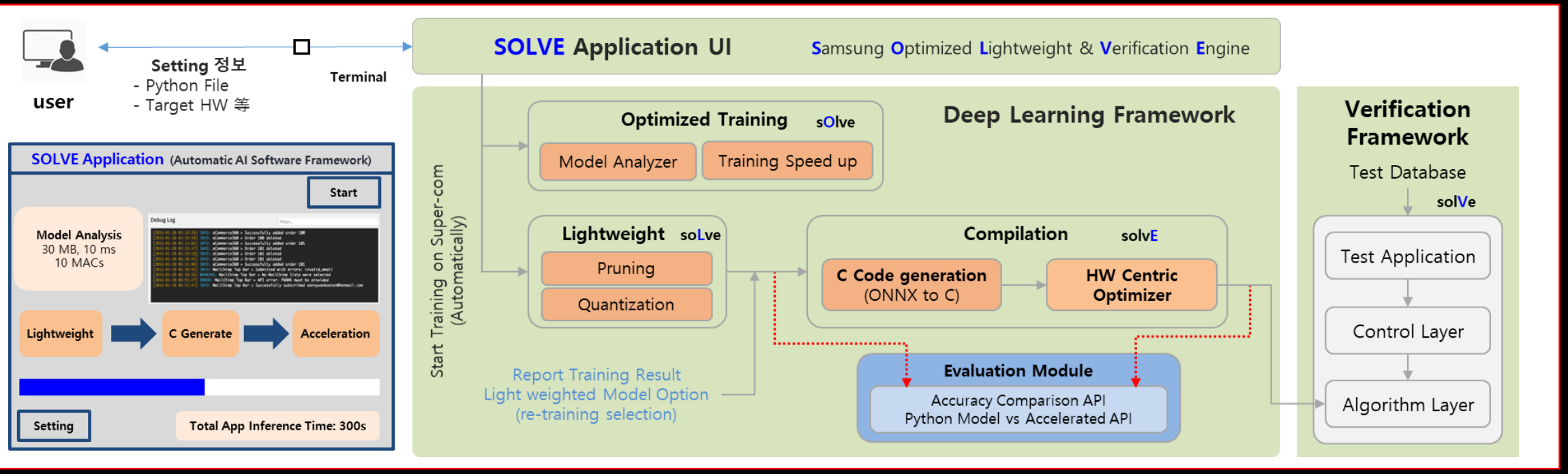
- sOlve (Optimized Training) 단계: 학습 속도 및 정확도 향상을 위한 학습 최적화 기술
- soLve (Lightweight Neural Network) 단계: 경량화를 통한 모델 사이즈 축소 및 동작 속도 향상 기술
- solVe (Verification Tool) 단계: 모듈화된 기능을 Layer 단위로 단계적인 평가 수행 기술
- solvE (Embedded Inference Engine) 단계: 대상 하드웨어 환경에 최적화된 실행 코드를 자동 생성하는 기술



④ Automatic AI Software Framework (2/3)

학습된 Python 모델을 입력 받아서 재-학습 가속화 (FP32→ FP16)와 경량화 (FP32→INT8/INT4) 처리를 거쳐 최적화된 실행 코드 (ARM NEON/x86 AVX 벡터 연산 지원)를 생성하는 솔루션 제공

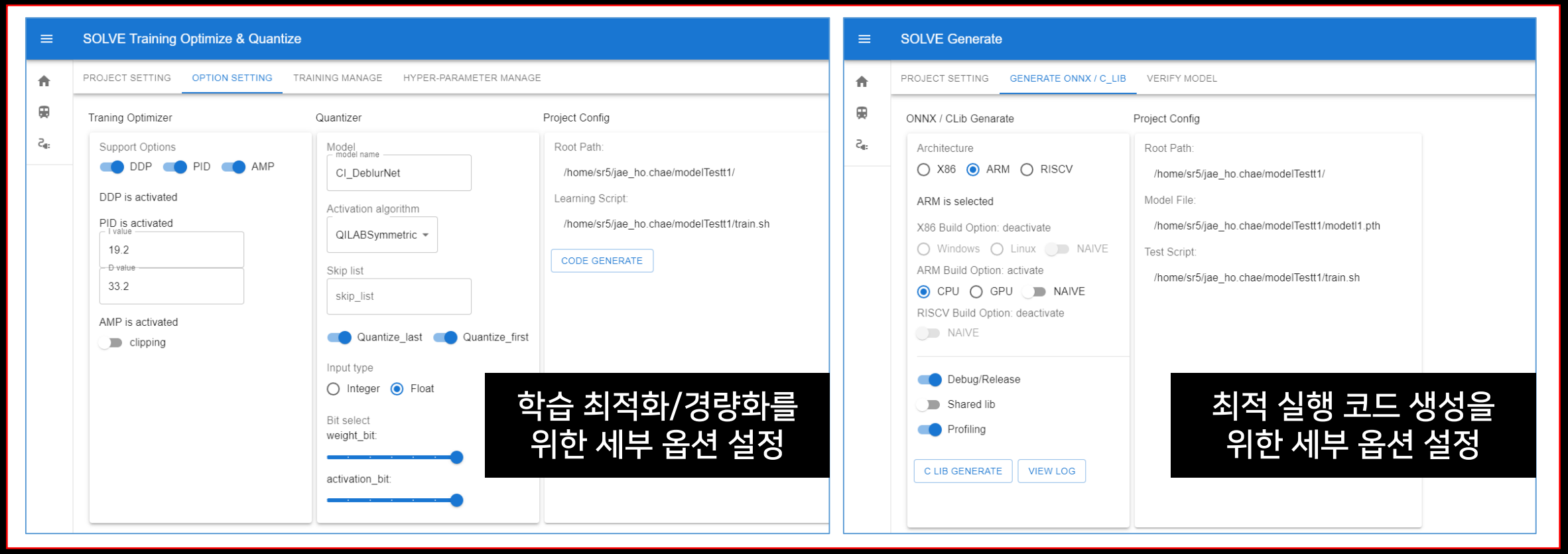
- 딥러닝 모델 학습 속도를 가속화하는 최신 기술 지원 및 모델 정확도를 향상하는 다양한 Optimizer 제공
- Low-Bit 양자화 학습의 최적 매개변수 탐색 시간을 단축하는 방법 (Hyper-parameter Search) 제공
- PyTorch 모델에서 ONNX 모델로 중간 변환을 거쳐서 최적화된 추론 코드를 자동 생성하는 실행 코드 변환 제공



④ Automatic AI Software Framework (3/3)

사용자의 편의성 향상을 위한 Web 형태의 GUI 환경 제공

- 사전 학습된 Python 모델과 사용자의 동작 환경 설정 정보를 받아서 전체 과정을 모두 자동화하여 일주일 이내로 상용화 수준 Embedded SW 개발 가능



Summary (Take-home Messages)

Embedded AI 제품의 SW 경쟁력 강화를 위한 Embedded 딥러닝 기술의 중요성 증대

- 앞으로 모든 센서는 인공 지능을 탑재하는 On-sensor AI로 발전 전망
  → 전력 및 메모리가 제한된 센서를 위한 AI 경량화 기술은 지속 연구 필요
- 하드웨어 특성을 반영한 경량화된 네트워크 설계 중요
  → 모델 크기와 연산량 절감과 함께 메모리 접근 횟수와 벡터연산 (병렬처리) 적합성을 고려한 설계 필요
- 딥러닝 경량화 기술
  → 재-학습 방식을 사용하면 4비트 수준까지는 성능 저하 없이 양자화 가능하나, 아직 2비트는 도전 과제
  → 구조화된 Point-wise (1x1xN 형태) Pruning 방식으로 효율적인 Latency 절감 가능
- Inference SW 최적화 기술
  → On-Device 프로세서에서 지원하는 벡터 연산 가속기에 최적화된 연산 구조와 제로 연산 제거 방식 활용

딥러닝 응용 개발자에게 손쉽고 빠른 개발을 제공하는 SW 프레임워크 확보가 AI 경쟁력 제고

- 다수 경쟁사에서 오픈 F/W을 공개하고 있으나, 선행 알고리즘 지원이나 추가 기능 구현에 제약 존재
  → 반복적인 AI 개발 과정을 자동화하는 기술 (경량화 모델 학습 → 제품 탑재 소스코드 생성) 자체 확보

Thank you