Spring Project：Multi Backend Neural Network Auto Quantization and Deployment over ONNX

商汤 sensetime

Fengwei Yu (SenseTime-China)
yufengwei@sensetime.com

Spring

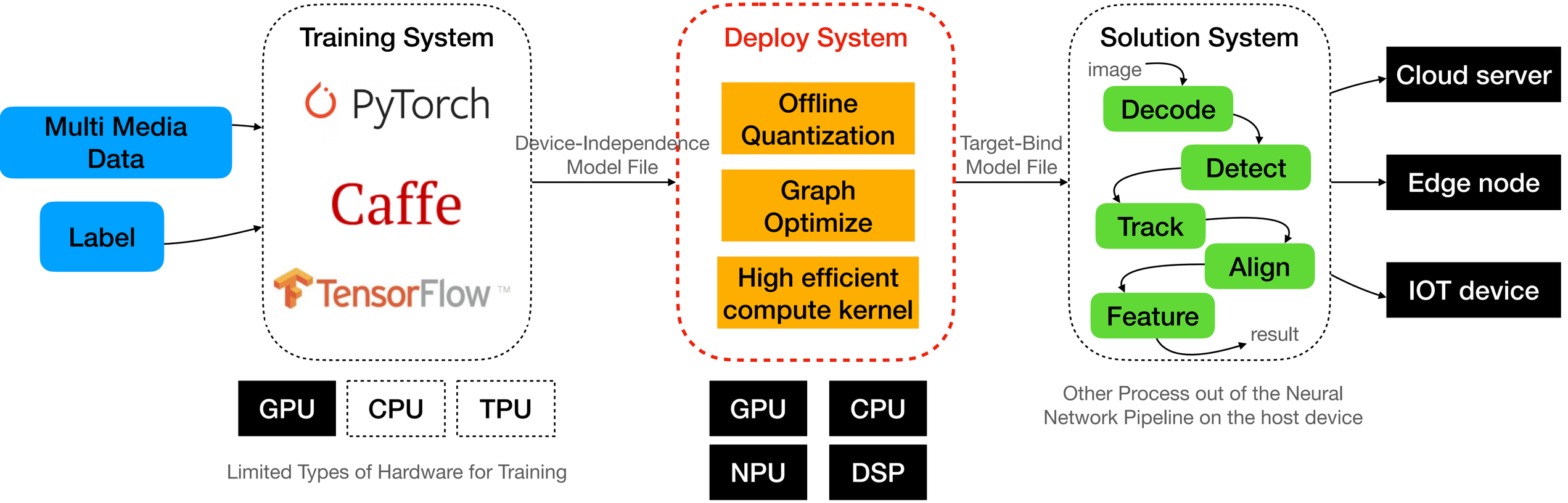
Contents

- What is Spring Project.
- Neural Network Deployment.
  - Multi-device：NART
  - Efficiency：Quantization
  - Automatic：Adela system
- From Caffe to ONNX.

What is Spring Project?

Spring

:Industrial Grade AI Model Production Framework



Neural Network Deployment： Features and Challenges

： **From** Device-Independence Model File to Target-Bind Model File

★3 key features of a Neural Network Deployment Framework

Multi-Device

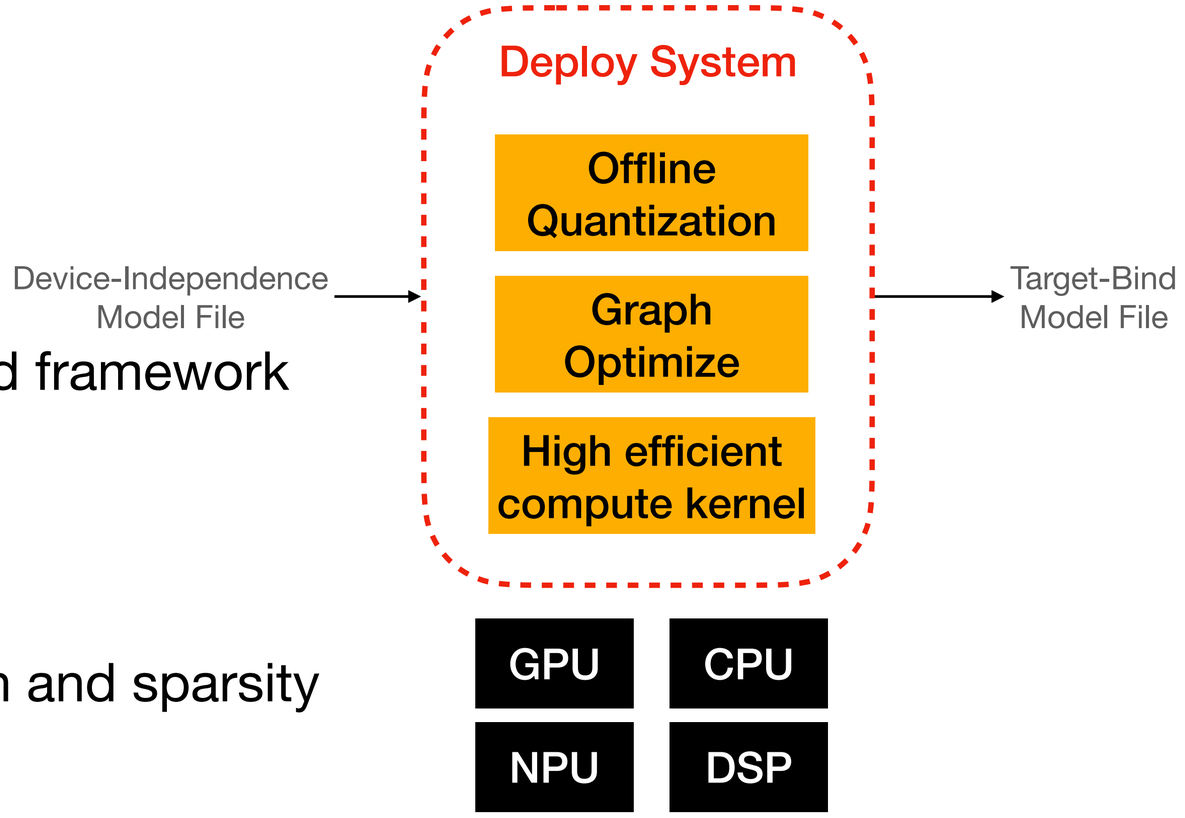
support more and more devices with a unified framework
enable to run

Efficiency

high efficiency on a special target
support model compression like quantization and sparsity

Automatic

more “end2end” in one model
we want “once-for-all”
we don’t want “compile failed”



Our Solutions：Multi-Device

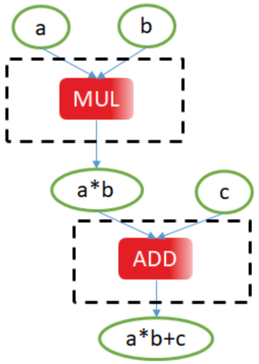
A little fact for codegen solution (like TVM) to support a new device：Almost all nowaday NPUs provide only an end2end compiler (of a net) instead of releasing their low level instruction for the developer.

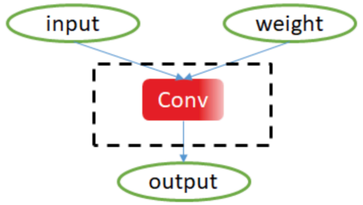
We need a framework which supports any type of integration：

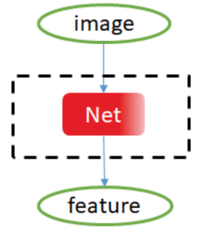
- **Code-level**：arm、x86、cuda、BangC、openCL
- **Op-level**：cuDNN、mklDNN、ARM-Compute Library
- **Net-level**：openvino、TensorRT、ncnn/mnn/tnn、davinci（huawei）、nnie（hisicon）、jlq、neuware（cambricon）、sgstar、rknn（rockchip）

We build a compiler system named spring.nart：

How we build NART







Feature 1：multi-scope OP

view a compiled net as a type of OP

serialize file as OP setting

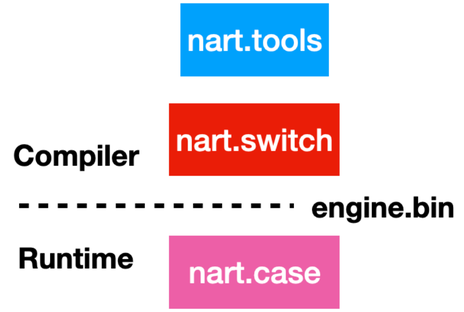
Feature 2：compile / runtime separate

runtime (named nart-case):

only contain a large set of OP

compiler (named nart-switch):

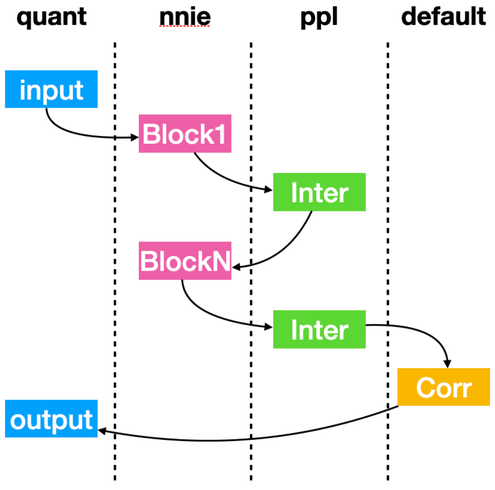
deal with quantization/graph optimization/format transform/ bypass compile



Feature 3：mixed operation

Deal with fall back：op not support in NPU

Mix precision：auto insert quant and de-quant OP



with NART, we can now support more than **16** type of devices now（**with a single ONNX file input**），include：davinci、cambricon、nvidia、ambarella、vp6、nnie、jlq、movidius、sigmstar、ceva、opencl、rockchip、snpe、x86、arm and etc….

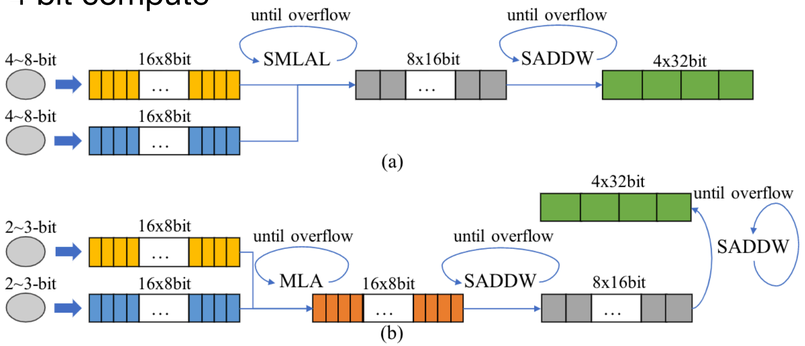
Our Solutions： Efficiency

Programable hardware（GPU、CPU）： openvino、TensorRT perform very well on fp32/fp16/int8、But lack the support of low bit (< 8bit)

★first support extremely low-bit convolution on modern computer architectures

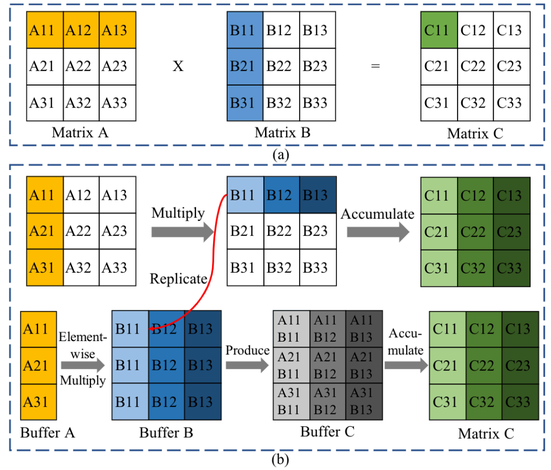
For arm neon：

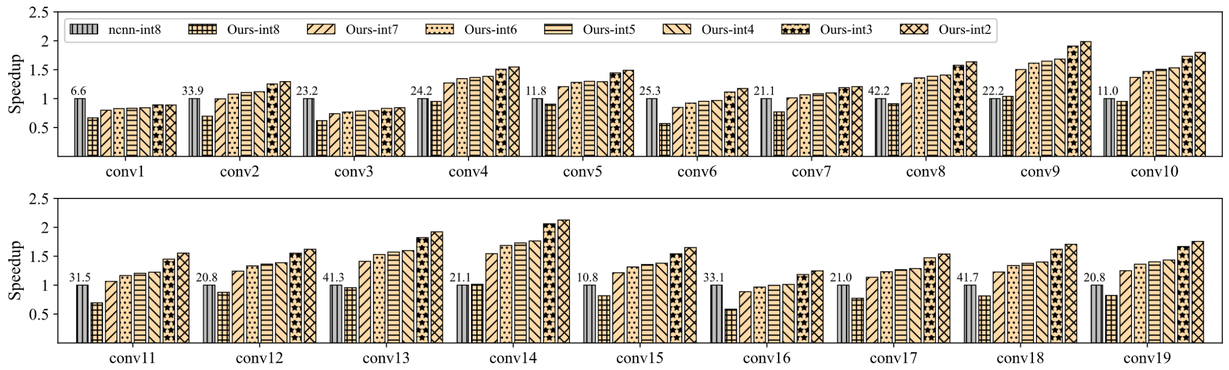
use smlal for 5~8 bit compute
use mla for 2~4 bit compute

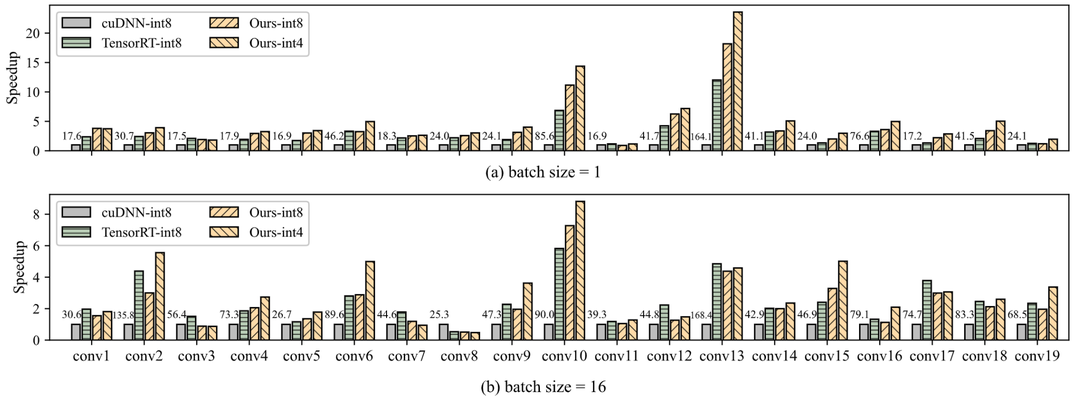


For turning tensorcore：

single load-replicate instruction on GEMM-based convolution
data padding and packing optimization.



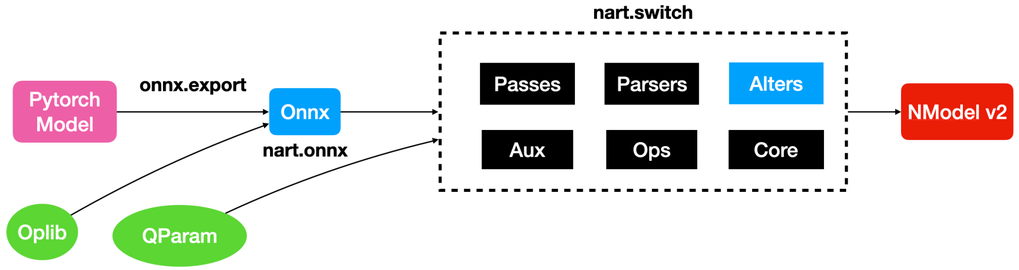


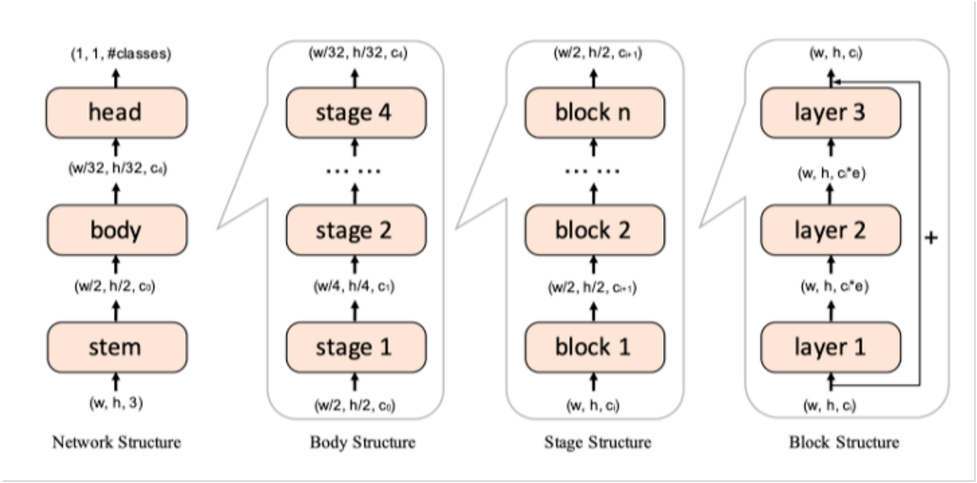


Our Solutions： Efficiency

almost every NPU supports int8、but the accuracy is not good enough

★ NART compiler allows user-defined quantization parameter
★ BRECQ： a new advance offline quantization algorithm that improves the quantized accuracy.





Hardware	Corp	Library	Bit number	Type
GPU	Nvidia	TensorRT	8	Per channel+ asym
			Fp16	IEEE 754
		NART-quant	4/8	Per channel/layer
3559/3519/3516	Hisilicon	Nnie	8/16	Log type
Ceva DSP	Ceva	PPL	8/16	Per channel/layer
Hexagon DSP	Qualcomm	PPL	8/16	Per channel/layer
		SNPE	8	Per layer + asym
Adreno 5/6 seria	Qualcomm	PPL	Fp16	IEEE 754 without Subnormal
ARM	ARM	NART-quant	2~8	Per layer + asym
WUQI	WUQI tech	WUQI sdk	8/16	Ristretto type
SigmaStar	SigmaStar	SigmaStar sdk	8/16	Per-channel, asym A+ sym W
Davinci	Huawei	ACL	8	Per-channel
			Fp16	IEEE 754
MLU2xx	Cambricom	Neuware	8/16	Per-channel
JLQ	JLQ	jnne	8/16	Per-layer
RockChip	RockChip	rknn	8/16	Per-layer

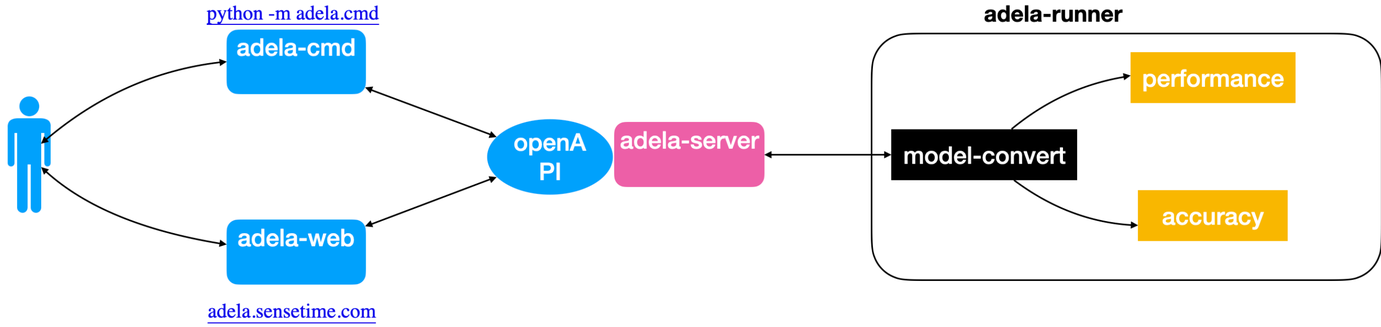
ICLR 2021： BRECQ: Pushing the Limit of Post-Training Quantization by Block Reconstruction

Our Solutions： Automatic

★ Adela System (internal)： An end2end SaaS solution

★ Model file （ONNX）

- upload once, then deploy and test everywhere！



- More than **500** users
- **200+** deployments and tests per day
- **30+** new model uploads per day

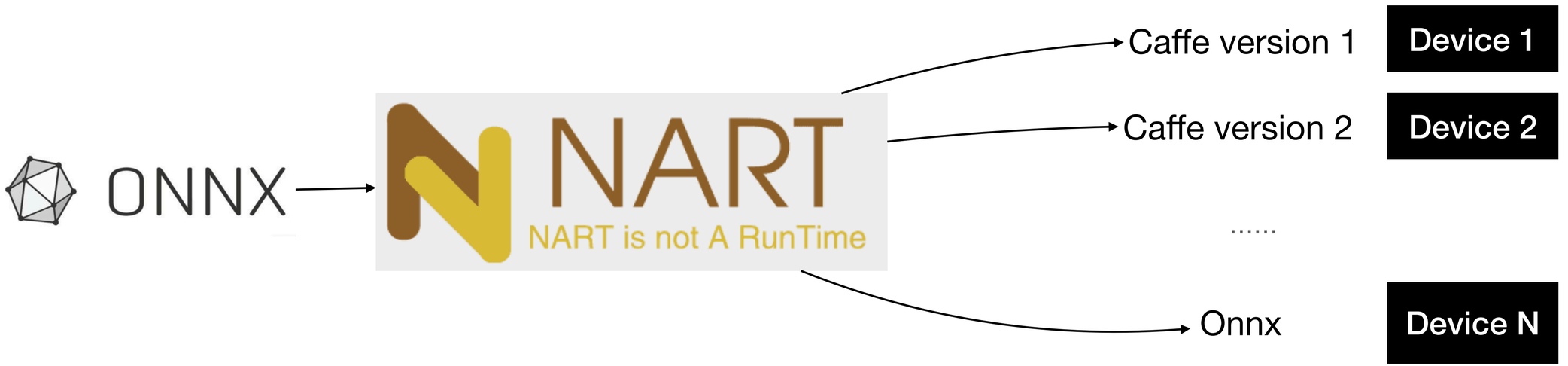
From Caffe format to Onnx

Caffe

- × centralized caffe.proto
- × non standard
- × hard to extend a new op
- × not support constant tensor

ONNX

- ✓ extendable onnx.proto
- ✓ opset version
- ✓ easy to extend a new op
- ✓ support constant tensor



NART.switch compiles onnx to various types of caffe version

Thanks!