The Evolution of Accelerators upon Deep Learning Algorithms

Song Yao, Shuang Liang, Junbin Wang, Zhongmin Chen, Shaoxia Fang, Lingzhi Sui,
Qian Yu, Dongliang Xie, Xiaoming Sun, Song Han, Yi shan, and Yu Wang

August 21st, 2018, Cupertino

*The authors represent the combined efforts from many talented and hard-working engineers in DeePhi

CONTENT 目录

Part 1

What's New About DeePhi Tech

Discover the Philosophy behind Deep Learning Computing

DeePhi: Now Part of Xilinx

Long History, Close Collaboration, and Better Future

Collaboration with Xilinx University Program

Deep learning acceleration
Time series analysis
Stereo vision
……

清華大學 Tsinghua University

Development of products on Xilinx FPGA platform since inception of DeePhi

Face recognition
Video analysis
Speech recognition acceleration
……

DEEPHi 深鉴科技

Co-Marketing and Co-Sales with Xilinx Team

Data Center
Automotive
Video surveillance
……

DEEPHi 深鉴科技

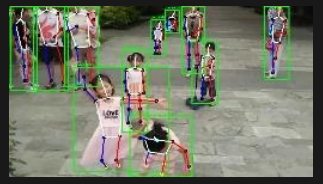
DEEPHi 深鉴科技
Now Part of Xilinx

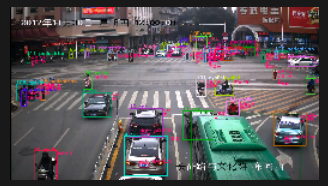
We Provide Full-Stack Deep Learning Solutions

DEEPHI 深鉴科技 Now Part of XILINX

Models


Face detection


Pose estimation


Video analytics


Lane detection


Vehicle detection


Segmentation

Framework

Caffe

TensorFlow

dmlc mxnet

Tools

DEEPHI DNNDK

IP & AI Platforms


Z7020

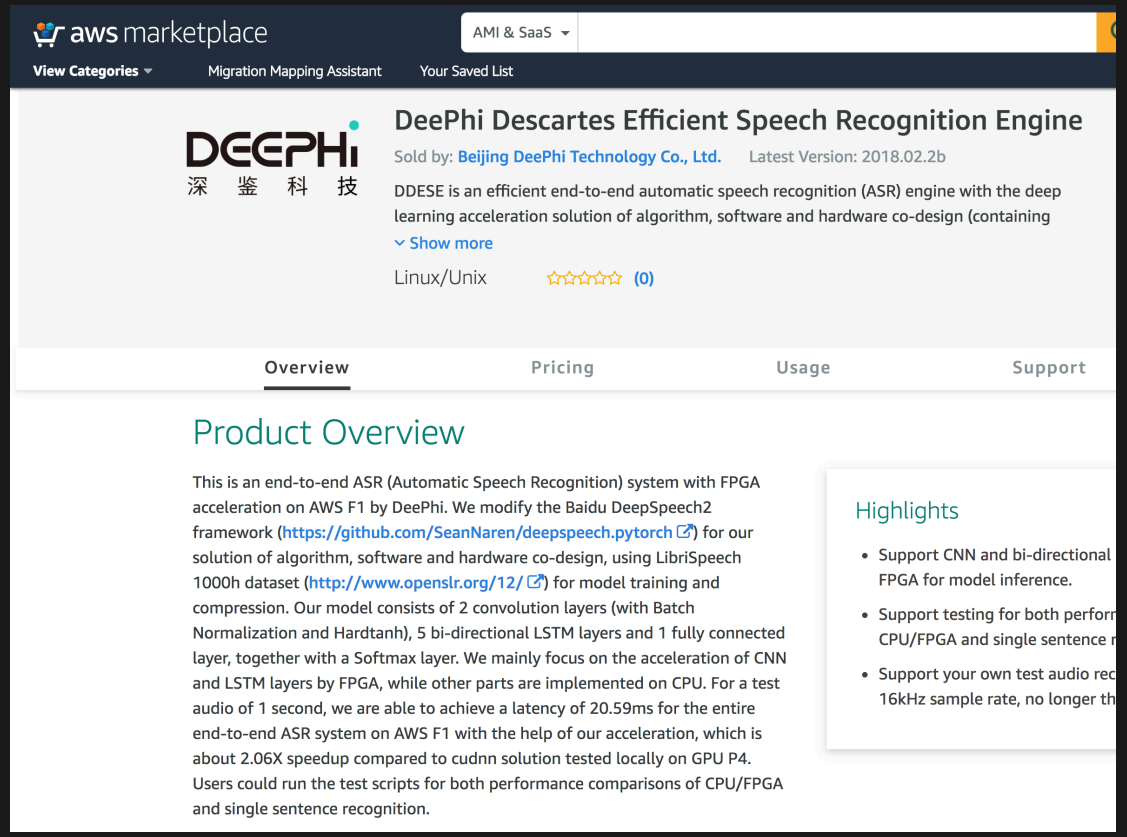

Z7020


ZU2


ZU2

ZU9

DeePhi's Solution now on AWS and Huawei Cloud!



https://aws.amazon.com/marketplace/pp/B079N2J42R?from=timeline&isappinstalled=0

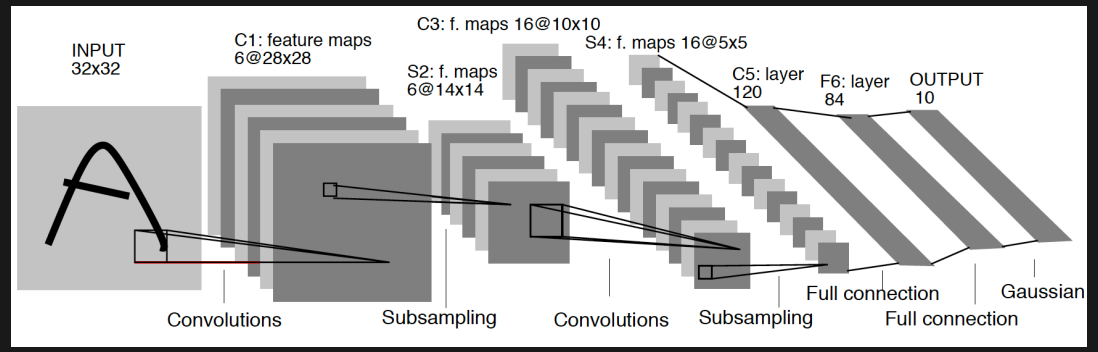
https://market.huaweicloud.com/product/00301-110982-0--0

Part 2

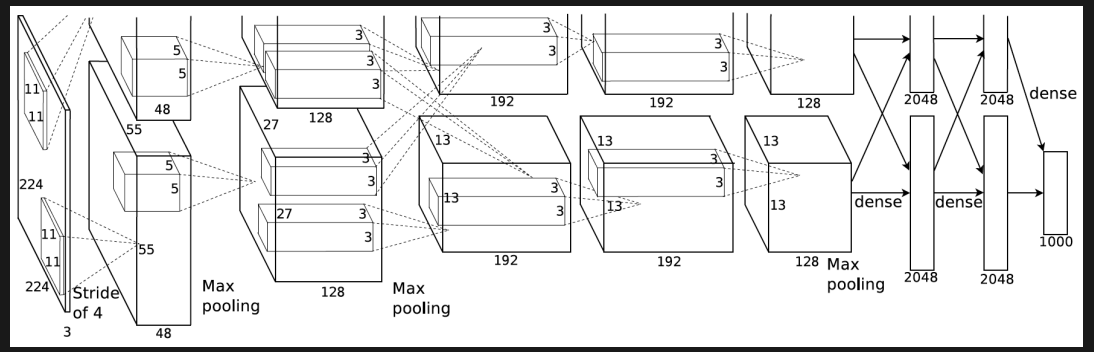
Evolution of Algorithms

Discover the Philosophy behind Deep Learning Computing

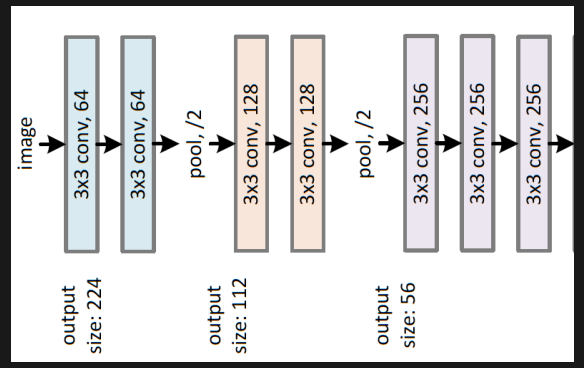
In 2018, We Are Still Using Old Algorithms as Benchmark



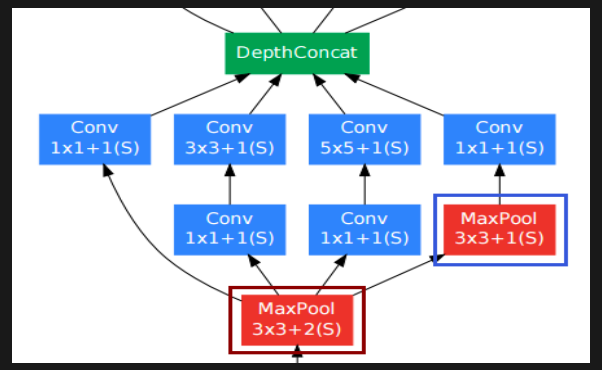
LeNet-5: 1998



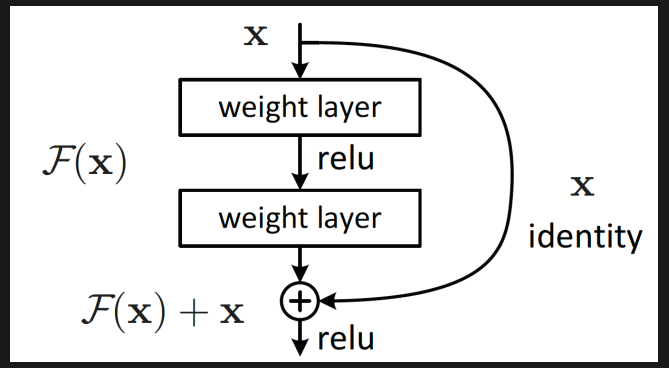
AlexNet: 2012



VGG-Net: 2014



GoogLeNet: 2014



ResNet: 2015

Changes in 2015-2018 Are More than That before 2015

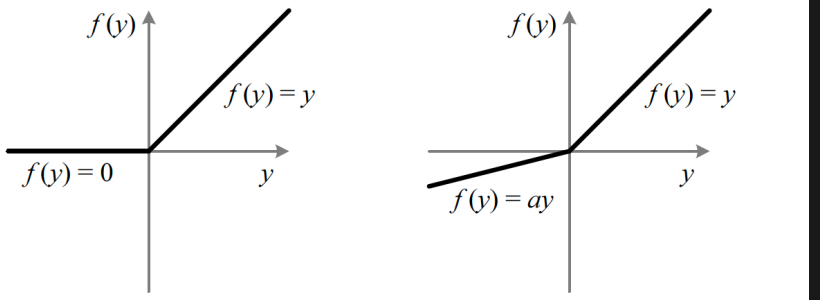


Figure 1. ReLU vs. PReLU. For PReLU, the coefficient of the negative part is not constant and is adaptively learned.

ReLU is not the only widely used activation function

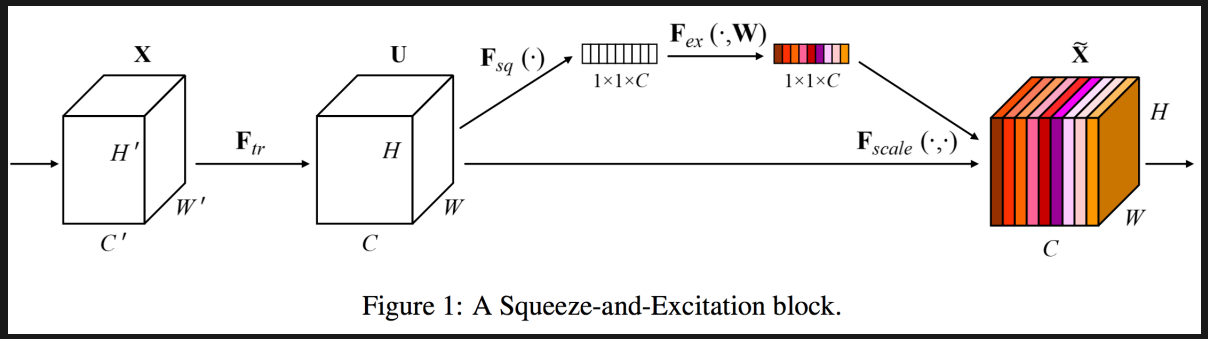
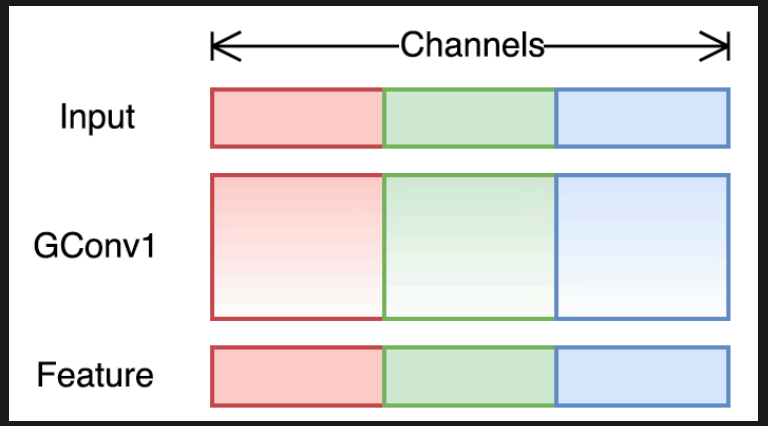
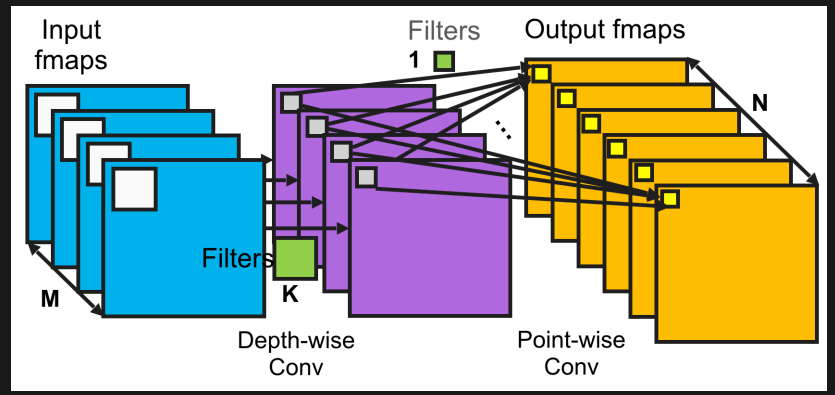


Figure 1: A Squeeze-and-Excitation block.

Channels can have different weights

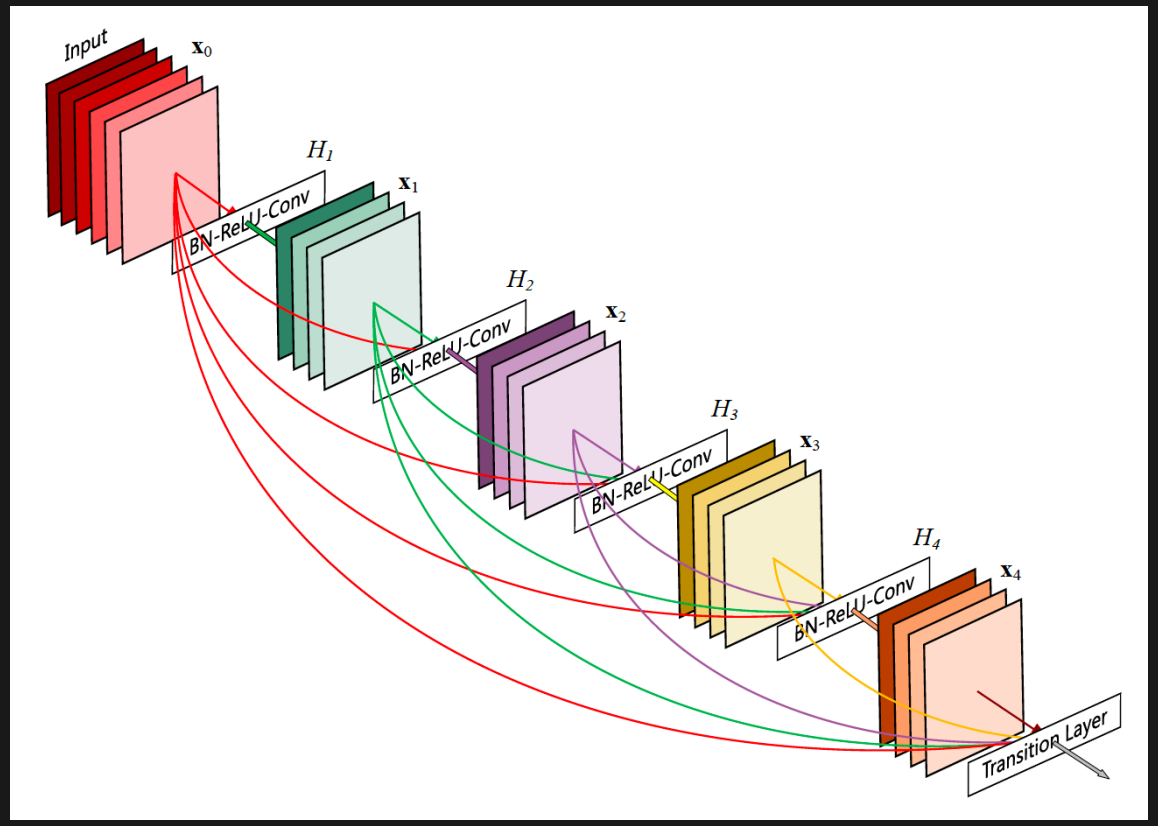


Group Conv: Not every input channel is connected to every output channel

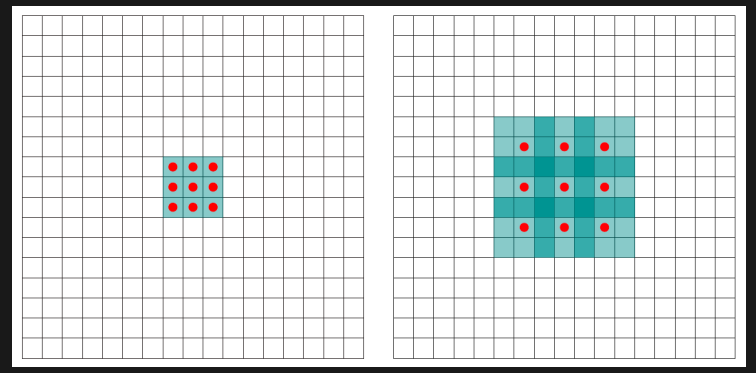


Depth-wise Conv and point-wise Conv are widely used

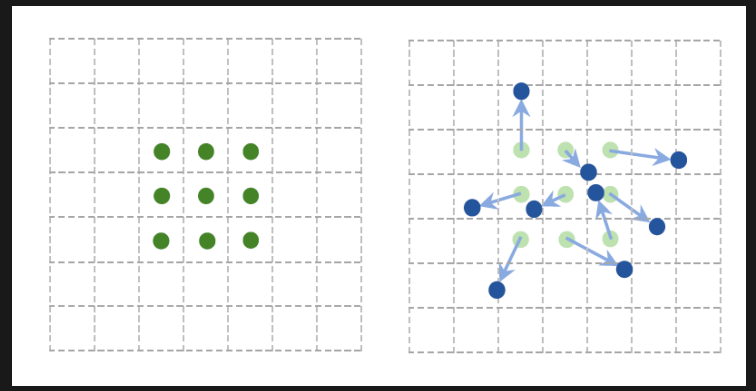
Changes in 2015-2018 Are More than That before 2015



DenseNet: Layers are not necessarily serial



Dilated Conv: Conv kernels are not necessarily dense



Deformable Conv: Conv kernels are not necessarily rectangle

Part 3

Evolution of DeePhi's DPU

Discover the Philosophy behind Deep Learning Computing

From DPU-V1 to DPU-V2.5

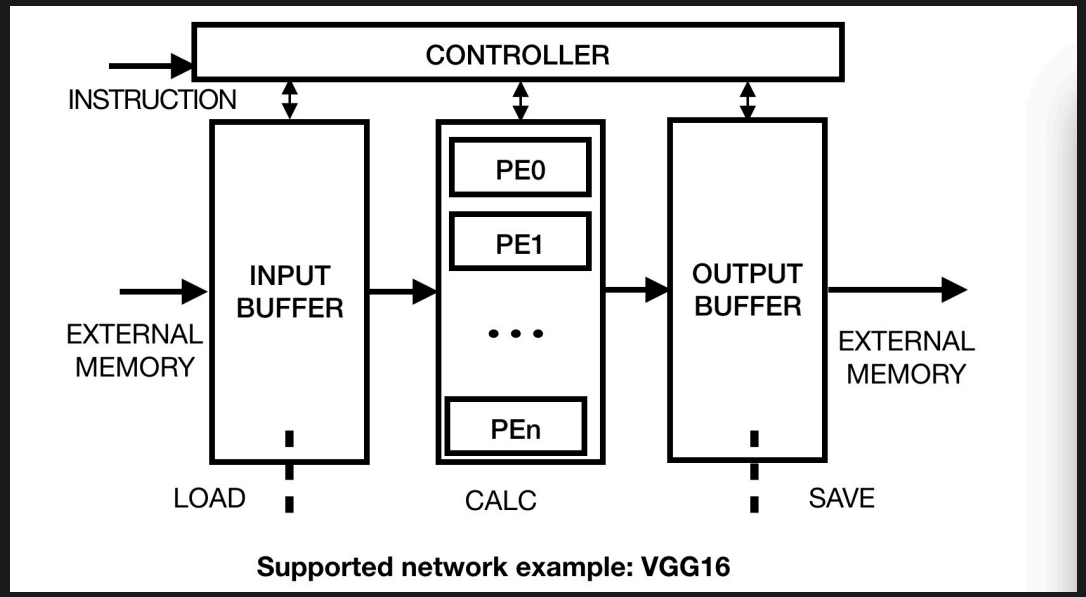
Architecture (single core)	DPU-V1	DPU-V2	DPU-V2.5
Peak performance (GOPS)	120	1350	1367
On-chip Memory (KB)	300	1123	1123
LUT	30k	75k	37k
FF	35k	146k	80k
Frequency (MHz)	214	333	333
Typical FPGA Platform	Zynq 7020 (28nm)	ZU9 (16nm)	ZU9 (16nm)
Total Cores	1	2	3
Total Peak Perf (GOPS)	120	2700	4100

Direction of improvements:

Better scheduling strategy, higher resource utilization, more supported features, and more flexibility

From DPU-V1 to DPU-V2.5

DPU-V1



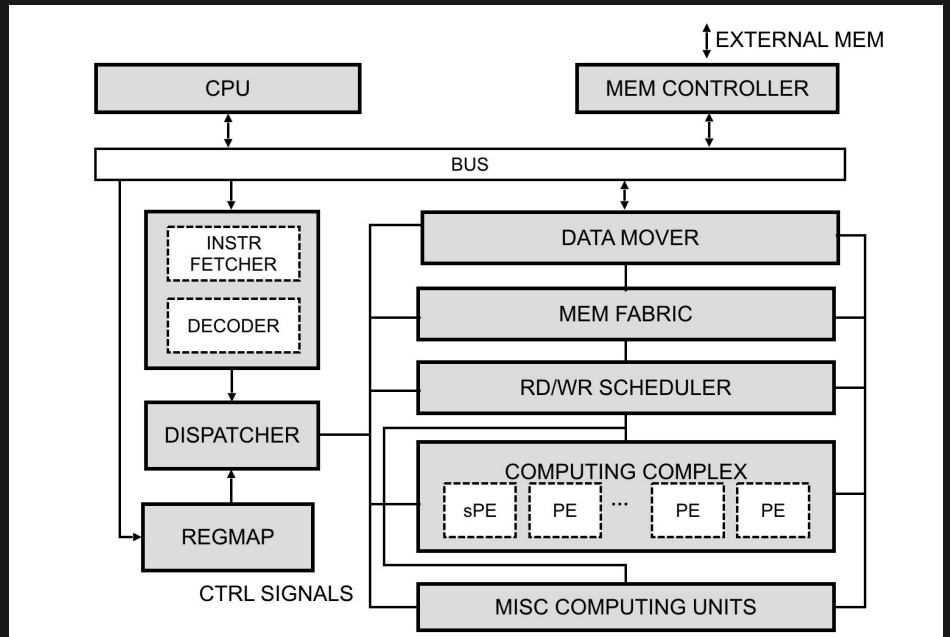
Supported network example: VGG16

1. Convolution
(a) Kernel size=3*3
(b) Stride=1
(c) Arbitrary padding size

2. Pooling
(a) Kernel size=2*2
(b) Stride=2

3. Activation function
(a) ReLU

DPU-V2



Supported network example: LeNet/ResNet50/InceptionV1/InceptionV2/MobileNet*/YOLO*/SegNet*/FPN*
note: the networks with * is only supported on DPU_V2.5

1. Feature map
(a) Elementwise

2. Convolution
(a) Arbitrary kernel size
(b) Arbitrary stride
(c) Arbitrary padding size

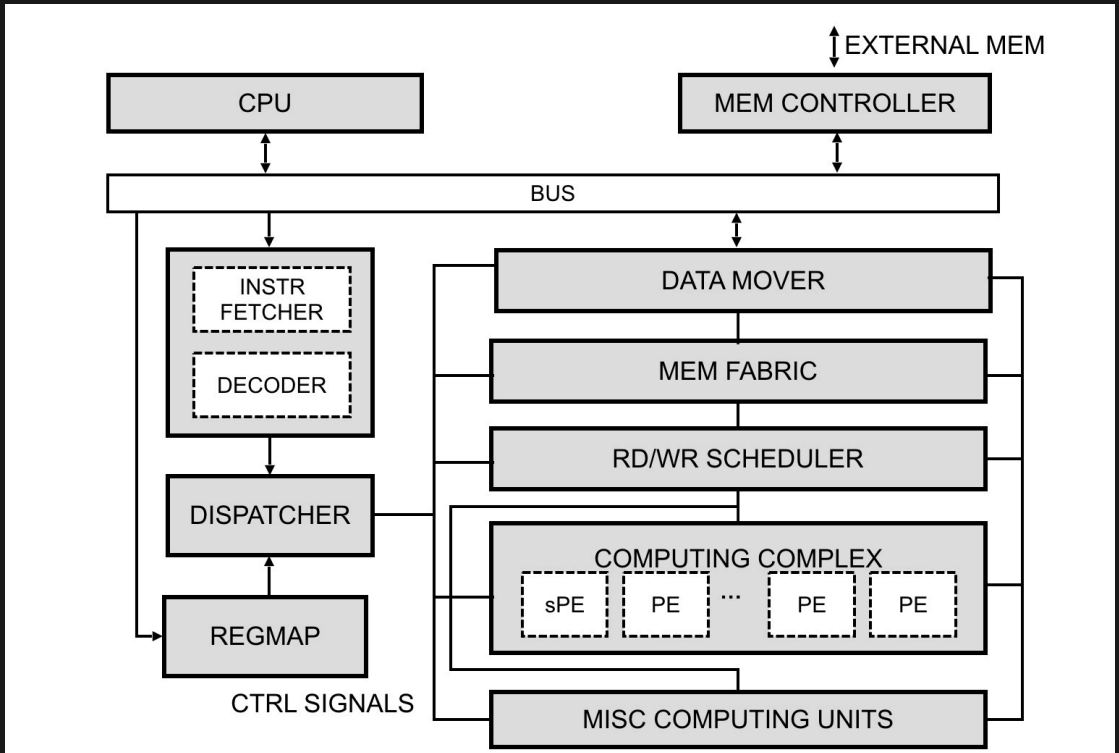
3. Pooling
(a) avg/max pooling
(b) Size=2*2,3*3
(c) Stride=1,2
(d) Arbitrary padding size

4. Activation function: (a) ReLU

5. FC(INT8)

6. Multi-Batch

From DPU-V1 to DPU-V2.5



Supported network example: LeNet/ResNet50/InceptionV1/InceptionV2/MobileNet*/YOLO*/SegNet*/FPN*
note: the networks with * is only supported on DPU_V2.5

DPU-V2.5

1. Feature map
(a) Elementwise
(b) Split
(c) Concat
(d) Resize
(e) Batch Normalization

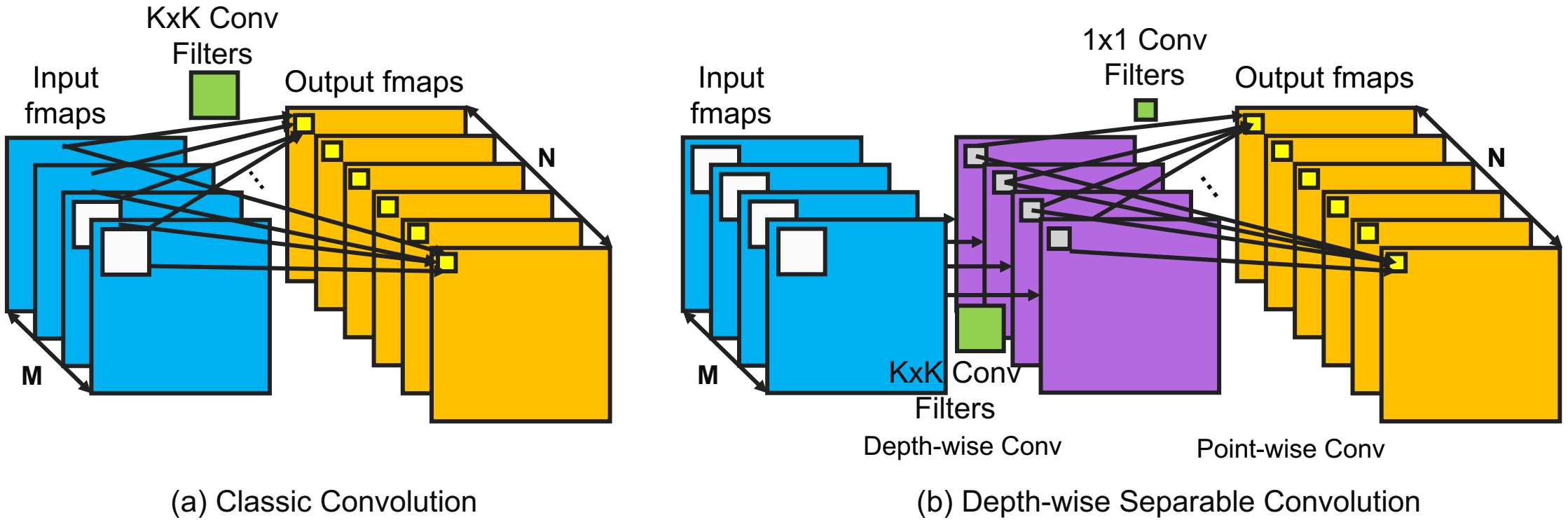
2. Convolution
(a) arbitrary kernel size
(b) arbitrary stride
(c) arbitrary padding size
(d) Deconv
(e) Dilated Conv
(f) Depthwise Conv

3.Pooling
(a) avg/max pooling
(b) arbitrary size
(c) arbitrary stride
(d) arbitrary padding
(e) ROI pooling

4.Activation function
(a) ReLU
(b) PReLU
(c) LeakyReLU
(d) Sigmoid

5.FC (INT8/FP32)
6.Multi-Batch

Case Study: Depth-Wise (DW) Convolution

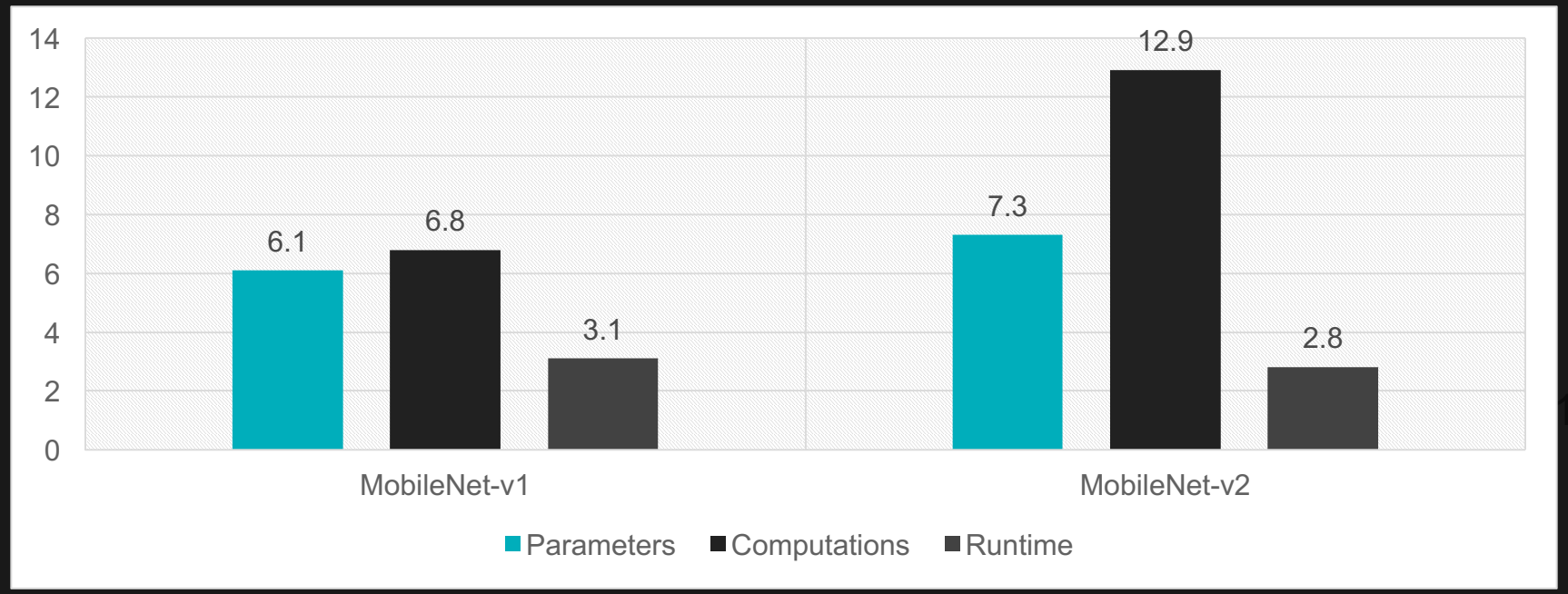


(a) Classic Convolution

(b) Depth-wise Separable Convolution

Depth-Wise: Number of groups = Number of channels

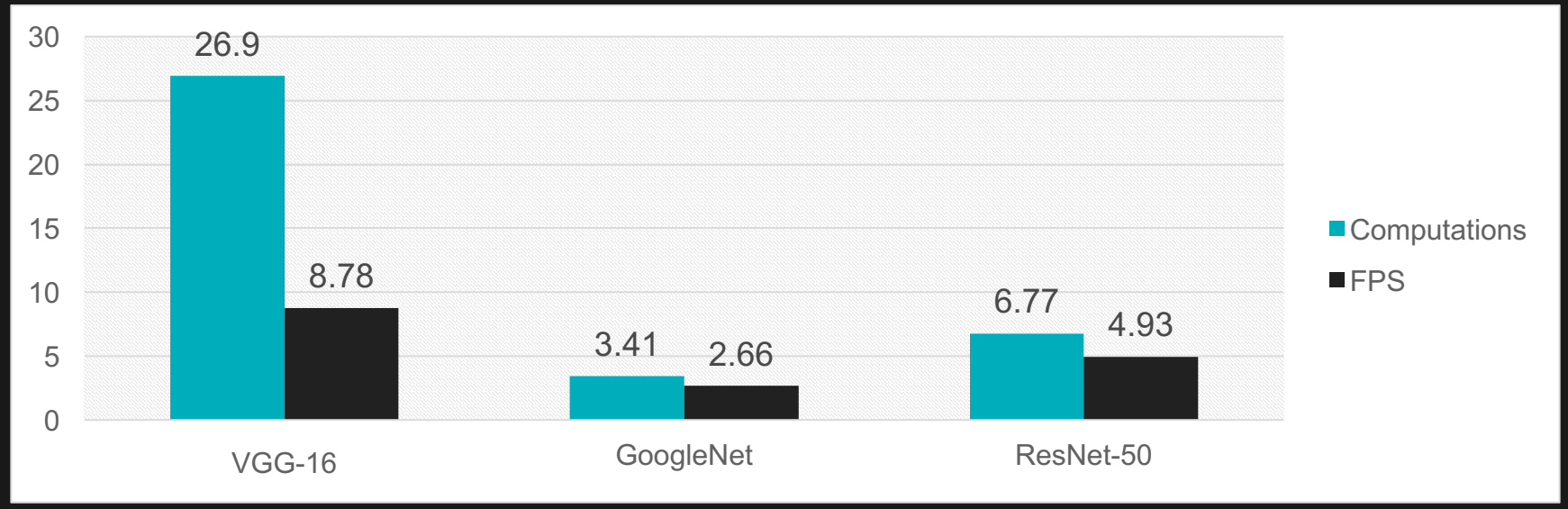
Efficient Networks May Not Be Friendly to Hardware



Times of improvement of MobileNet-v1/v2 over ResNet-50 on **Tesla P100 GPU**

Model	# Parameters	# Computations	Runtime (ms)	Reference Top-1 Accuracy
ResNet-50	25.5M	7.72 GOPS	4.20	76.1%
MobileNet-v1	4.2M	1.14 GOPS	1.37	70.6%
MobileNet-v2	3.5M	0.60 GOPS	1.49	72.0%

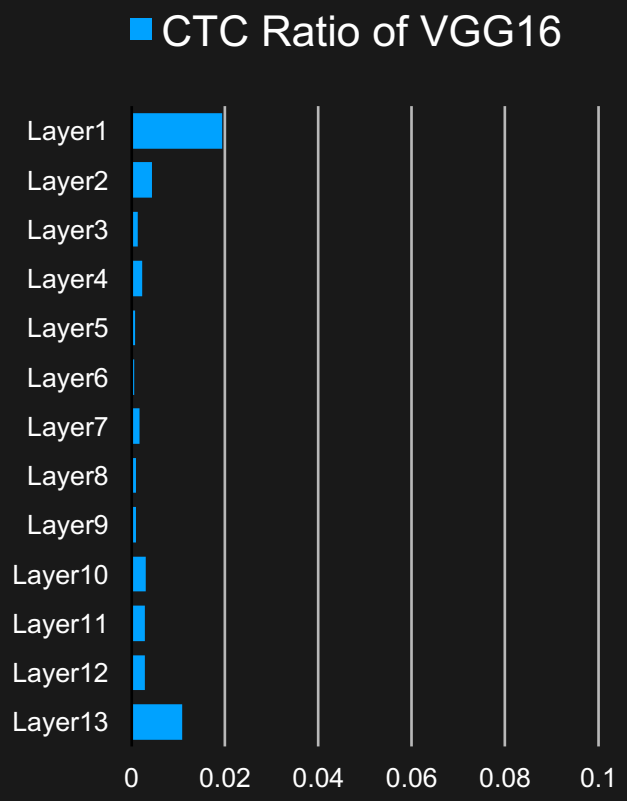
Efficient Networks May Not Be Friendly to Hardware

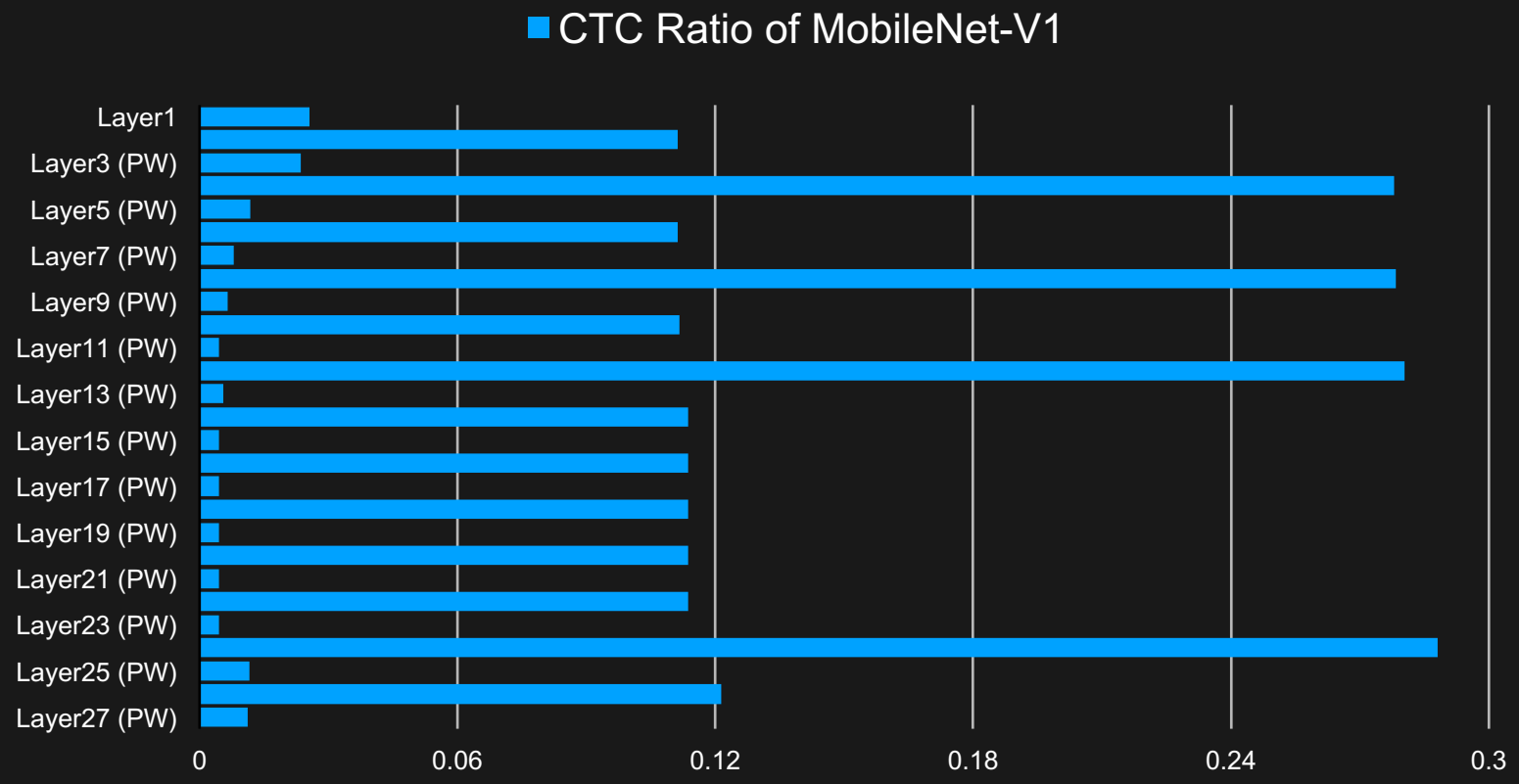


Times of improvement of MobileNet-v1 over three benchmark networks on **DeePhi's DPU on ZU9 FPGA**

Model	Computations	Reference Top-1 Accuracy	Performance	FPS	Utilization Rate	Year
VGG-16	30.7 GOPS	71.9%	2.36 TOPS	76.9	87.30%	2013
GoogleNet	3.89 GOPS	71.0%	0.99 TOPS	254	36.80%	2014
ResNet-50	7.72 GOPS	76.1%	1.06 TOPS	137	39.30%	2015
MobileNet-v1	1.14 GOPS	70.6%	0.77 TOPS	675	28.62%	2017

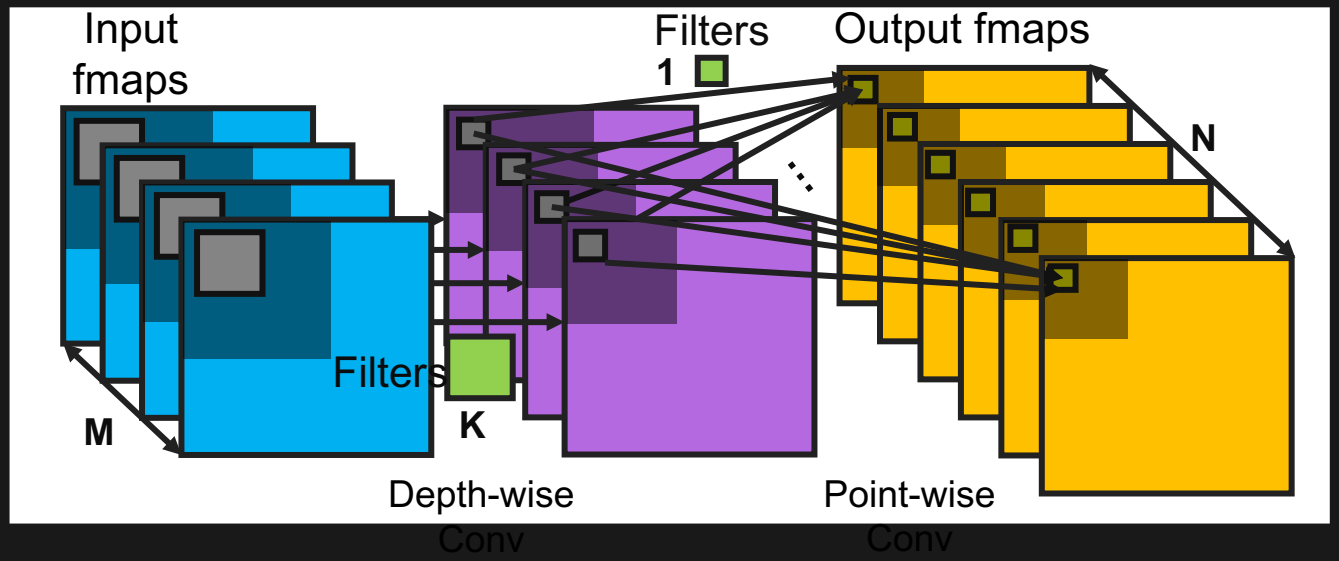
Why Depth-Wise Convolution so Slow?



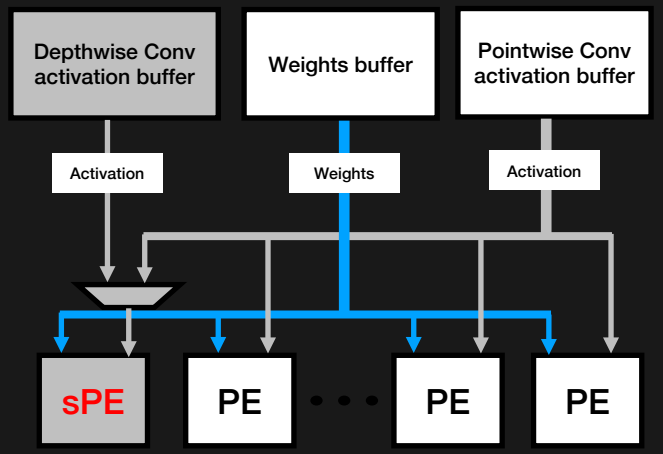


- The Communication/Computation (CTC) Ratio of depth-wise is very high
- Point-wise (1x1, PW) conv is memory-bound

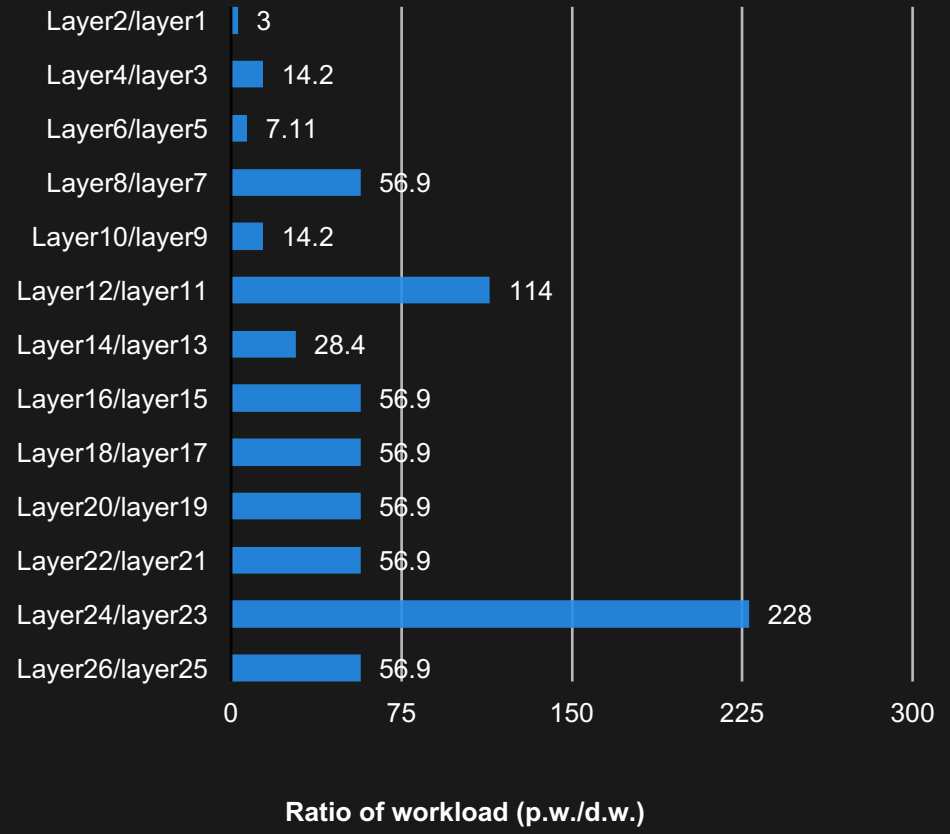
Our Strategy Resolving the Problem



(a) Fused-layer Convolution for MobileNet

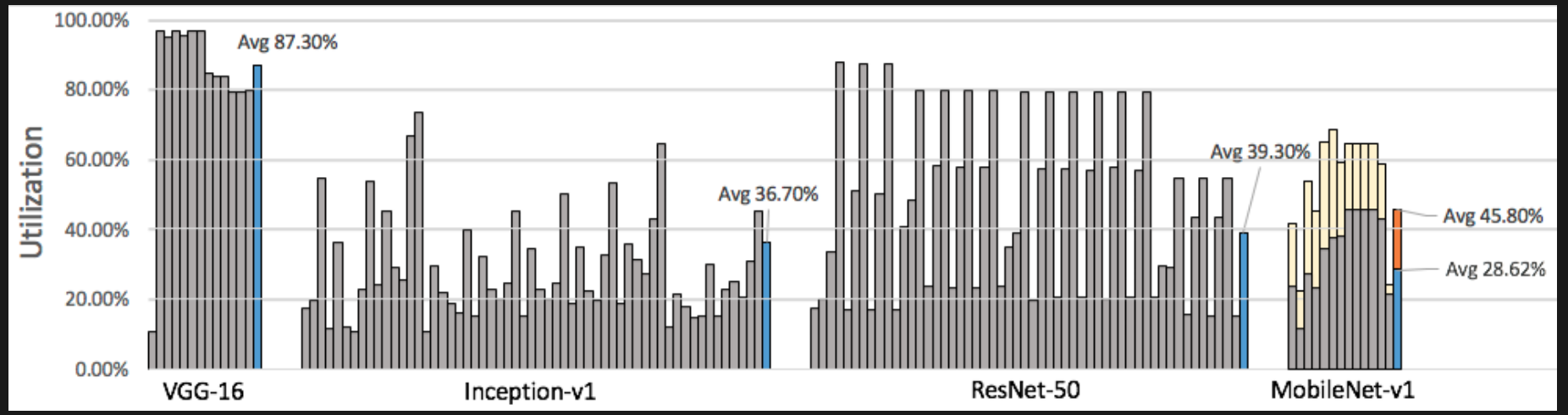


(c) Computing Architecture



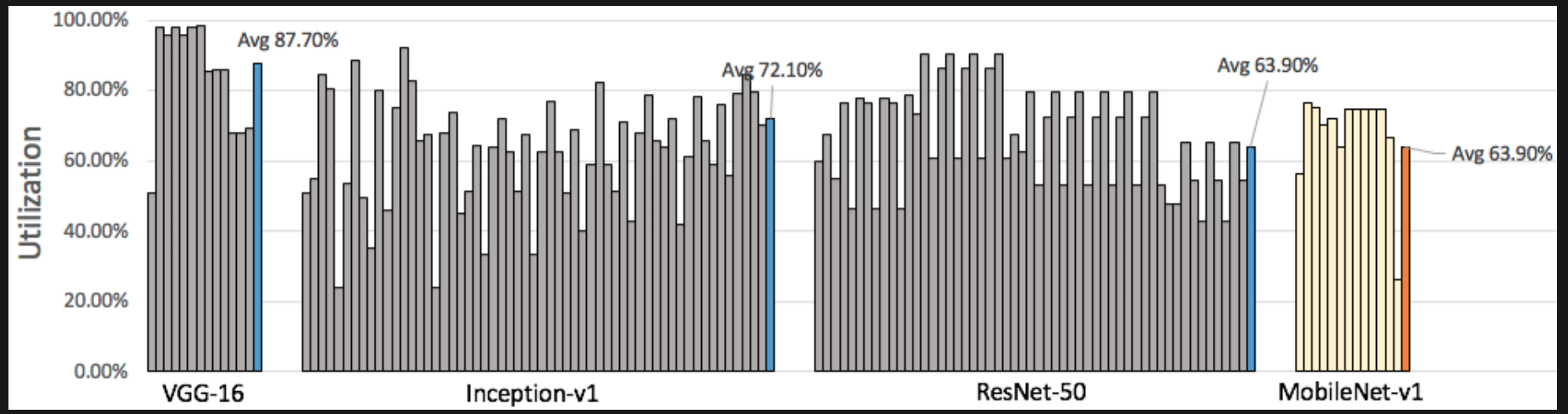
(b) The workload ratio between adjacent PW-Conv and DW-Conv

Utilization Improvement on MobileNet-v1 with Layer Fusion



Results on ZU9 FPGA

- Dual cores on ZU9 FPGA
- Each core uses 1024KB BRAM
- In total 3648KB BRAM on ZU9 FPGA



Simulation results with doubled BRAM

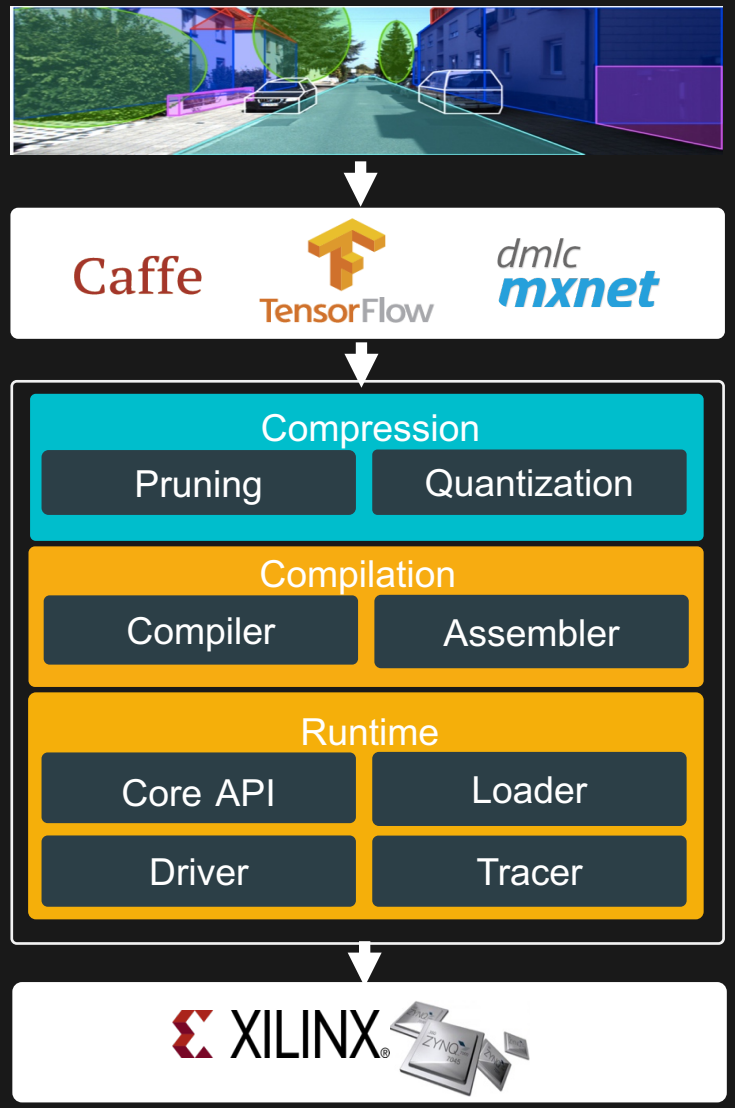
- Each core uses 2048KB BRAM

Part 4

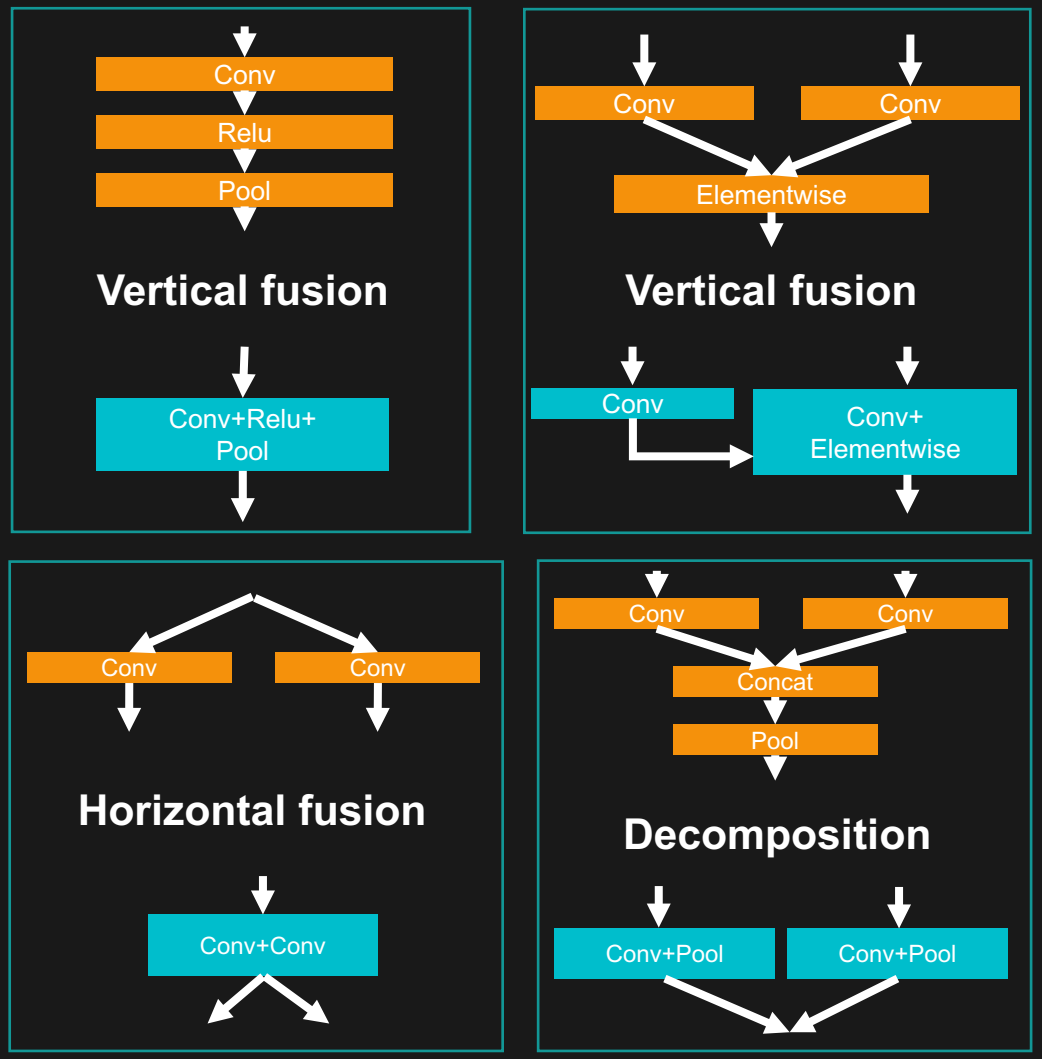
DL Software: New Changes

Discover the Philosophy behind Deep Learning Computing

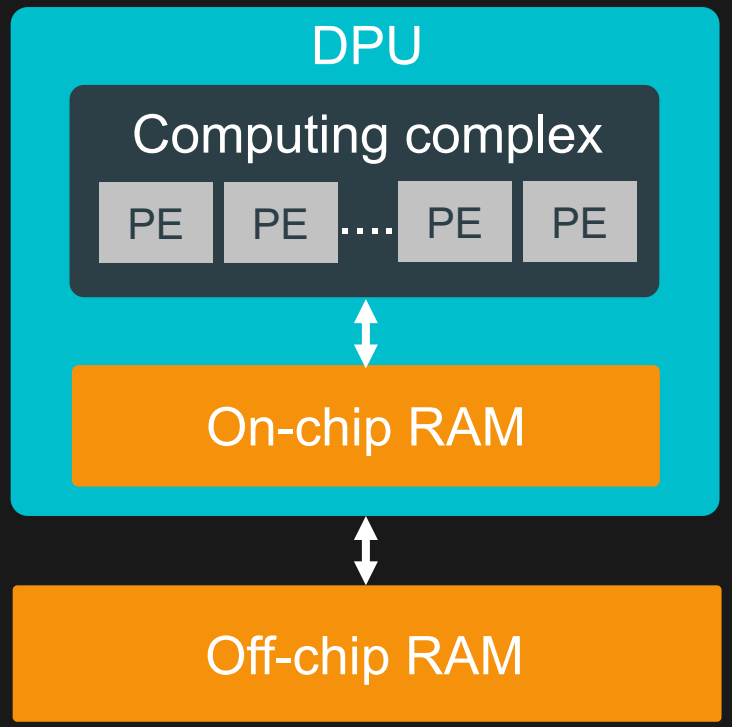
DNNDK: Make it Easy to Develop



More Optimizations Should be Considered in Compiler



Node fusion/decomposition & data stream optimization



Memory allocation/scheduling/reuse to improve performance

Part 5

Summary

Discover the Philosophy behind Deep Learning Computing

Summary

- Algorithms are evolving at an increasingly faster rate
- Modern neural networks like MobileNet are not necessarily friendly for hardware acceleration on existing ASICs
- We propose a fusing-layer strategy together with compiling optimization to better accelerate Depth-wise/Point-wise convolutions
- More optimization strategies for new types of networks should be considered

- Pure hardware evolves slowly due to the long period in designing and manufacturing chips
- FPGA can be benefited from latest DL techniques in both hardware and software side

DEEPHI 深鉴科技 Now Part of XILINX

THANK YOU!

官方微信

www.deephi.com

Friday, August 17, 2018

Discover the Philosophy behind Deep Learning Computing