香港中文大學
The Chinese University of Hong Kong

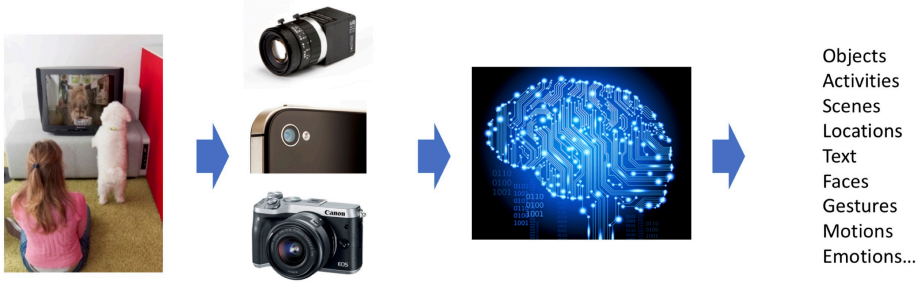
Deep Neural Network Design Automation

Bei Yu

Department of Computer Science and Engineering
The Chinese University of Hong Kong

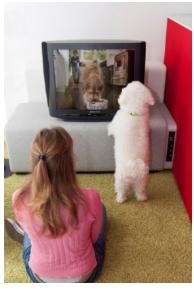
Computer Vision

- Humans use their eyes and their brains to visually sense the world.
- Computers user their cameras and computation to visually sense the world

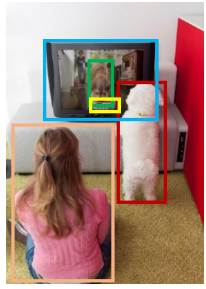


Jian Sun, “Introduction to Computer Vision and Deep Learning”.

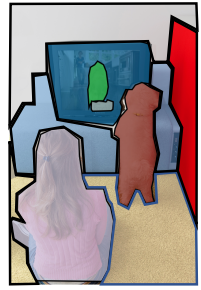
Few More Core Problems



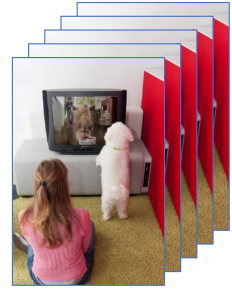
Classification



Detection



Segmentation

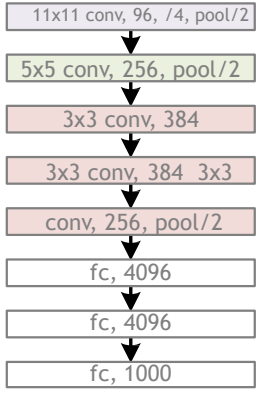


Sequence

Image → Region → Pixel → Video

Revolution of Depth

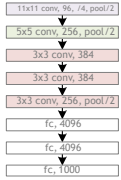
AlexNet, 8 layers (ILSVRC 2012)



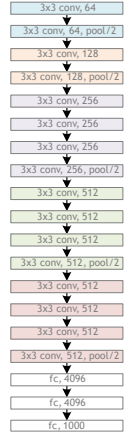
Slide Credit: He et al. (MSRA)

Revolution of Depth

AlexNet, 8 layers (ILSVRC 2012)



VGG, 19 layers (ILSVRC 2014)



GoogleNet, 22 layers (ILSVRC 2014)

Slide Credit: He et al. (MSRA)

Revolution of Depth

AlexNet, 8 layers (ILSVRC 2012)

VGG, 19 layers (ILSVRC 2014)

ResNet, 152 layers (ILSVRC 2015)

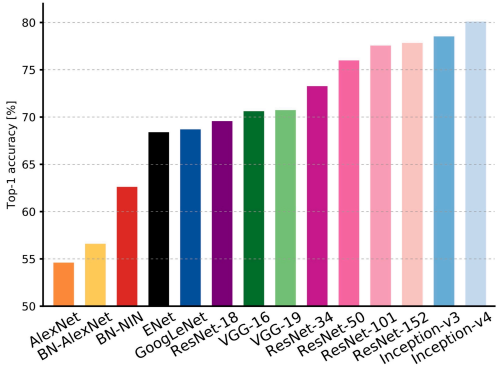
Slide Credit: He et al. (MSRA)

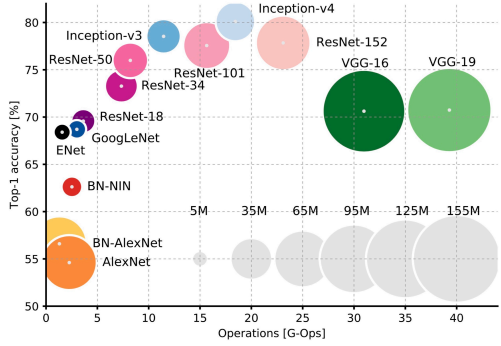
Some Recent Classification Architectures

- AlexNet (Krizhevsky, Sutskever, and E. Hinton 2012) 233MB
- Network in Network (Lin, Chen, and Yan 2013) 29MB
- VGG (Simonyan and Zisserman 2015) 549MB
- GoogleNet (Szegedy, Liu, et al. 2015) 51MB
- ResNet (K. He et al. 2016) 215MB
- Inception-ResNet (Szegedy, Vanhoucke, et al. 2016)
- DenseNet (Huang et al. 2017)
- Xception (Chollet 2017)
- MobileNetV2 (Sandler et al. 2018)
- ShuffleNet (Zhang, Zhou, et al. 2018)

Some Recent Classification Architectures

- AlexNet (Krizhevsky, Sutskever, and E. Hinton 2012) 233MB
- Network in Network (Lin, Chen, and Yan 2013) 29MB
- VGG (Simonyan and Zisserman 2015) 549MB
- GoogleNet (Szegedy, Liu, et al. 2015) 51MB
- ResNet (K. He et al. 2016) 215MB
- Inception-ResNet (Szegedy, Vanhoucke, et al. 2016) 23MB
- DenseNet (Huang et al. 2017) 80MB
- Xception (Chollet 2017) 22MB
- MobileNetV2 (Sandler et al. 2018) 14MB
- ShuffleNet (Zhang, Zhou, et al. 2018) 22MB

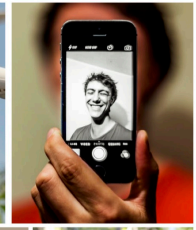




1

[1] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello (2017). "An analysis of deep neural network models for practical applications". In: *arXiv preprint*.

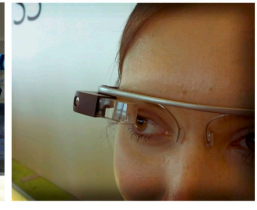










1

[1] Fei-Fei Li & Justin Johnson & Serena Yeung, Stanford cs231n.

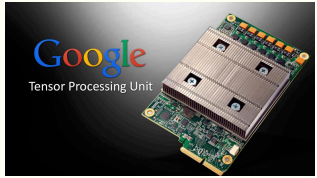
When Machine Learning Meets Hardware

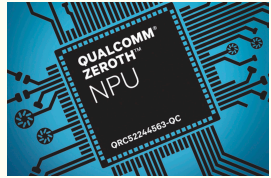
Convolution layer is one of the most expensive layers

- Computation pattern
- Emerging challenges

More and more end-point devices with limited memory

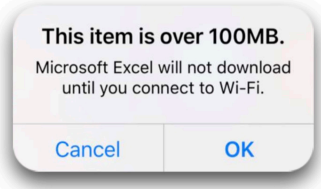
- Cameras
- Smartphone
- Autonomous driving

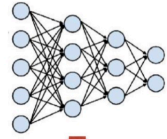




1st Challenge: Model Size

Hard to distribute large models through over-the-air update





1

[1] Song Han and William J. Dally (2018). "Bandwidth-efficient Deep Learning". In: *Proc. DAC*, 147:1–147:6.

2nd Challenge: Energy Efficiency

AlphaGo: 1920 CPUs and 280 GPUs,
$3000 electric bill per game

on mobile: drains battery
on data-center: increases TCO

1

[1] Song Han and William J. Dally (2018). "Bandwidth-efficient Deep Learning". In: *Proc. DAC*, 147:1–147:6.

Outline

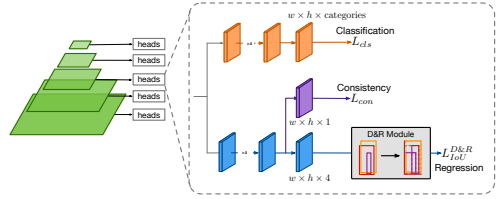
Outline

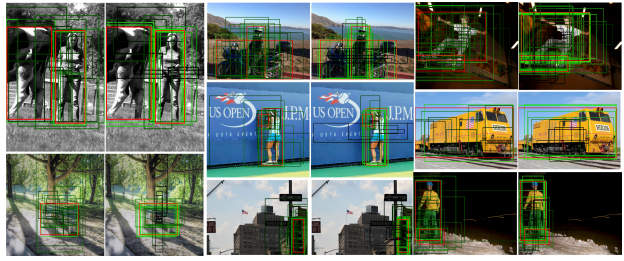
Algorithm EX1: Object Detection [ECCV'20]

Dive Deeper Into Box for Object Detection

Ran Chen[1], Yong Liu[2], Mengdan Zhang[2], Shu Liu[3],
Bei Yu[1], and Yu-Wing Tai[2]

[1] The Chinese University of Hong Kong
{rchen,byu}@cse.cuhk.edu.hk
[2] Tencent Youtu Lab
{ly.chaos,zhangmengdanrz,yuwingtai}@gmail.com
[3] SmartMore
sliu@smartmore.com



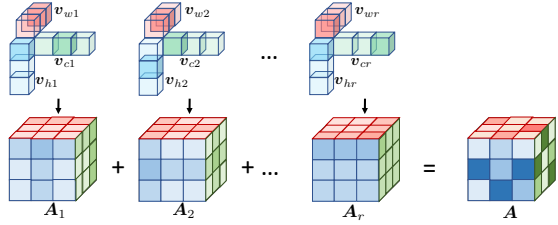


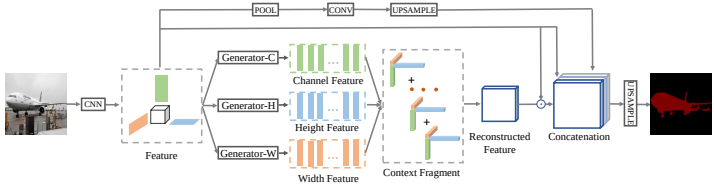
Algorithm EX2: Semantic Segmentation [ECCV'20]

Tensor Low-Rank Reconstruction for Semantic Segmentation

Wanli Chen[1], Xinge Zhu[1], Ruoqi Sun[2], Junjun He[2], Ruiyu Li[3], Xiaoyong Shen[3], and Bei Yu[1]

[1] The Chinese University of Hong Kong
{wlchen,byu}@cse.cuhk.edu.hk,zx018@ie.cuhk.edu.hk
[2] Shanghai Jiao Tong University
{hejunjun,ruoqisun7}@sjtu.edu.cn
[3] SmartMore
{ryli,xiaoyong}@smartmore.com

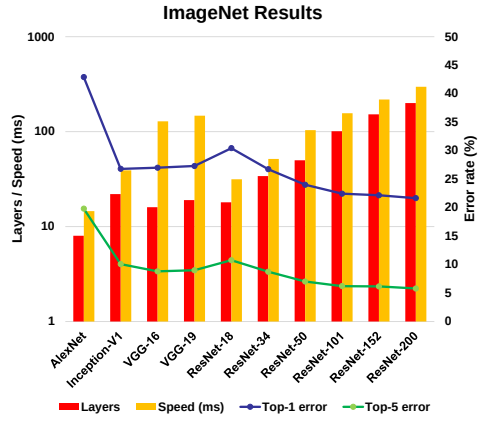




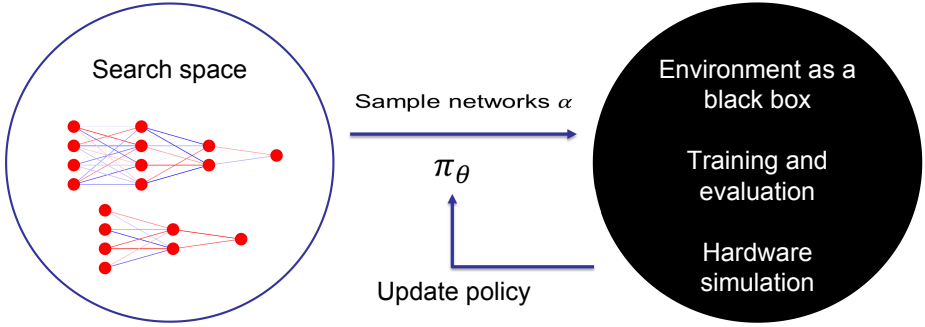
Outline

Neural Architecture Search (NAS)

- Designing neural architecture is extremely challenging.
- Mechanism of neural networks is not well interpreted.
- Can we advance AI/ML using artificial intelligence instead of human intelligence?



Neural Architecture Search (NAS)



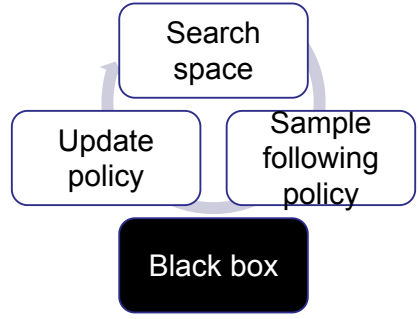
Neural Architecture Search (NAS)

Black box Optimization

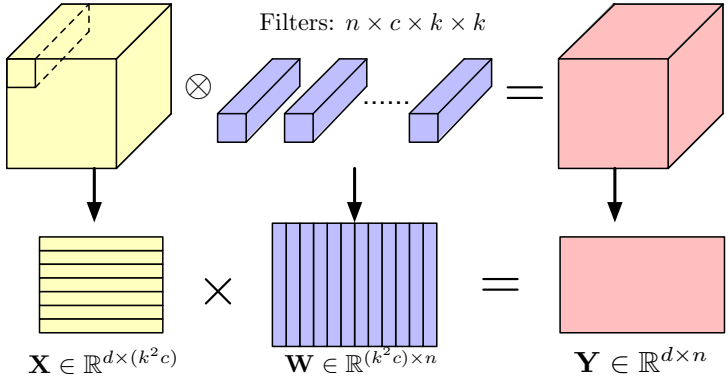
- Find the optimal network configuration to maximize the performance.
- Huge search space: *e.g.* 1.28×10^{54} settings.

Available methods

- Reinforcement learning.
- Evolutionary algorithm.
- Differentiable architecture search.

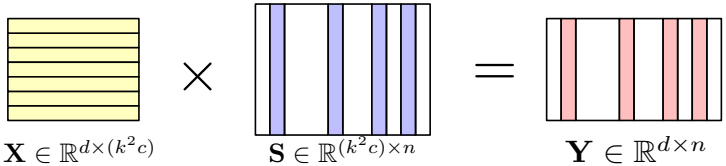


`Im2col` (Image2Column) Convolution



- Transform convolution to matrix multiplication
- Unified calculation for both convolution and fully-connected layers

Compression Approach: Sparsity[1],[2]



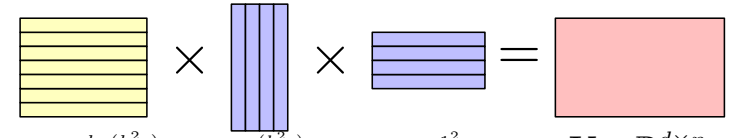
Sparse DNN

- *Sparsification*: weight pruning;
- *Compression*: compressed sparse format for storage;
- *Potential acceleration*: sparse matrix multiplication algorithm.

[1] Wei Wen et al. (2016). "Learning structured sparsity in deep neural networks". In: *Proc. NIPS*, pp. 2074–2082.

[2] Yihui He, Xiangyu Zhang, and Jian Sun (2017). "Channel Pruning for Accelerating Very Deep Neural Networks". In: *Proc. ICCV*.

Compression Approach: Low-Rank[1],[2]



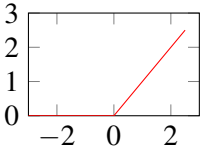
Low-rank DNN

- *Low-rank approximation*: matrix decomposition or tensor decomposition.
- *Compression and acceleration*: less storage required and less FLOP in computation.

[1] Xiangyu Zhang, Jianhua Zou, et al. (2015). "Efficient and accurate approximations of nonlinear convolutional networks". In: *Proc. CVPR*, pp. 1984–1992.

[2] Xiyu Yu et al. (2017). "On compressing deep models by low rank and sparse decomposition". In: *Proc. CVPR*, pp. 7370–7379.

Non-linearity Approximation[1]



ReLU

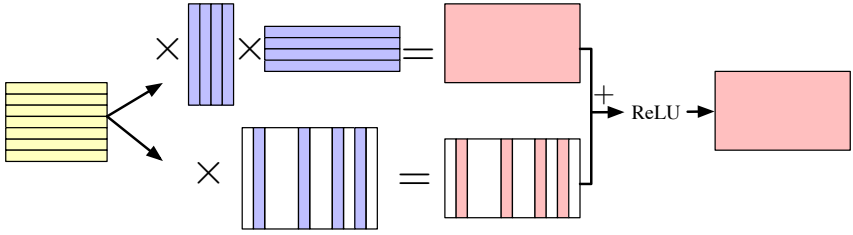
- Activation unit: `ReLU`
- Error more sensitive to positive response;
- Enlarge the solution space.

$$\min_{\boldsymbol{W}} \sum_{i=1}^{N} \|\boldsymbol{W}\boldsymbol{X}_i - \boldsymbol{Y}_i\|_F \rightarrow \min_{\boldsymbol{W}} \sum_{i=1}^{N} \|r(\boldsymbol{W}\boldsymbol{X}_i) - \boldsymbol{Y}_i\|_F$$

- $\boldsymbol{X}$: input feature map
- $\boldsymbol{Y}$: output feature map

[1] Xiangyu Zhang, Jianhua Zou, et al. (2015). "Efficient and accurate approximations of nonlinear convolutional networks". In: *Proc. CVPR*, pp. 1984–1992.

Our Idea: Unified Structure[1] (Best Student Paper Award)

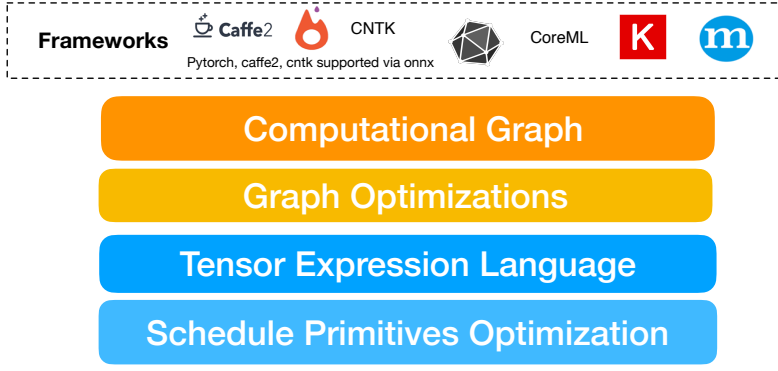


- Simultaneous low-rank approximation and network sparsification;
- Non-linearity is taken into account.
- Acceleration is achieved with structured sparsity.

[1] Yuzhe Ma et al. (2019). "A Unified Approximation Framework for Non-Linear Deep Neural Networks". In: *Proc. IJCAI*.

Outline

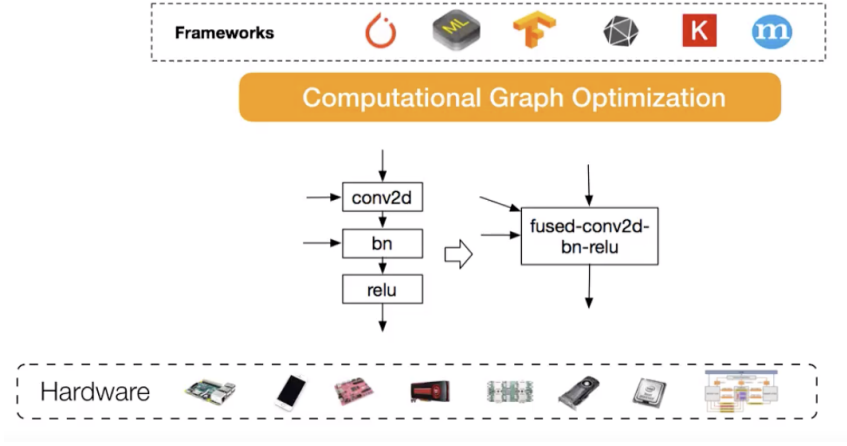
Deep Learning Compiler - TVM[1]



[1]Some materials are from CSE 599W: Systems for ML http://dlsys.cs.washington.edu/

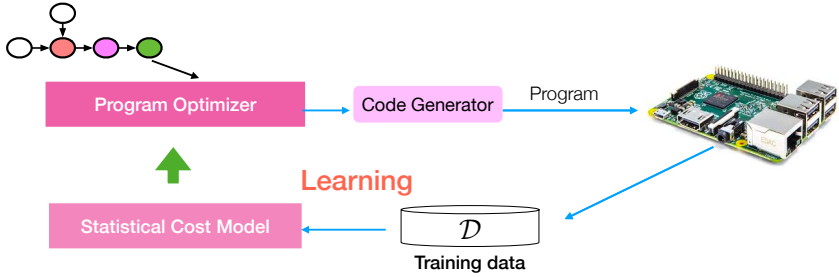
TVM: End to End Optimization

Computational Graph Optimization: Operator Fusion



TVM: End to End Optimization

Layer-wise Optimization: Autotuning



Tuning algorithms:

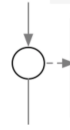
- Active learning.
- Transfer learning.
- Reinforcement learning.

TVM Domain Specific Language

Decoupling scheduling and algorithms.

- Specify the algorithm.
- Specify the schedule.

```
A = t.placeholder((1024, 1024))
B = t.placeholder((1024, 1024))
k = t.reduce_axis((0, 1024))
C = t.compute((1024, 1024), lambda y, x:
                t.sum(A[k, y] * B[k, x], axis=k))
s = t.create_schedule(C.op)
```

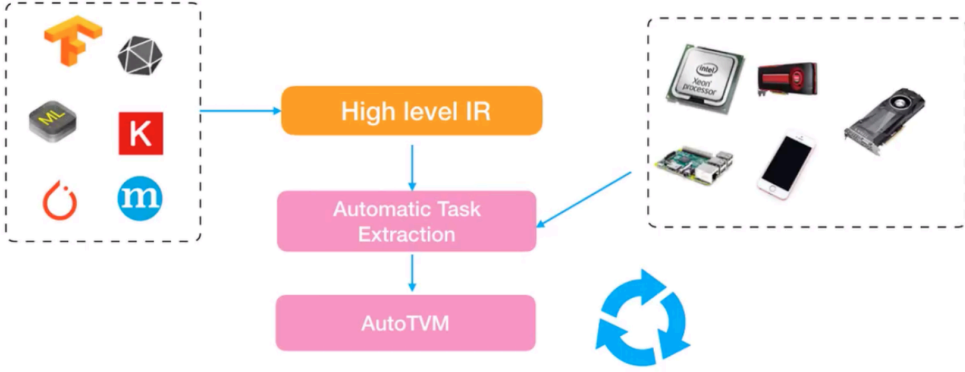


```
for y in range(1024):
  for x in range(1024):
    C[y][x] = 0
    for k in range(1024):
      C[y][x] += A[k][y] * B[k][x]
```

TVM/VTA: Full Stack Open Source System

- JIT compile accelerator micro code.
- Support heterogenous devices, 10x better than CPU on the same board.
- Move hardware complexity to software.

TVM: End-to-End Integration



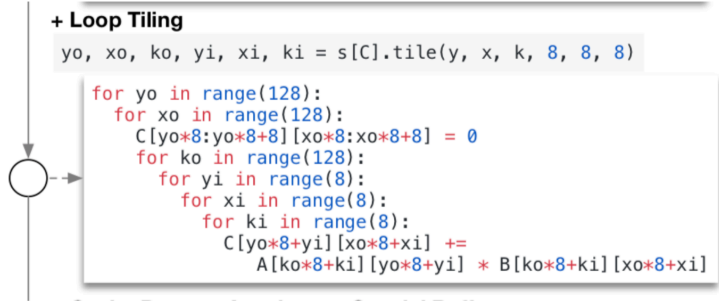
TVM Domain Specific Language + Loop Tiling

- Optimize data locality
- Minimize memory conflict
- Optimize for device cache

+ Loop Tiling

```
yo, xo, ko, yi, xi, ki = s[C].tile(y, x, k, 8, 8, 8)
```

```
for yo in range(128):
  for xo in range(128):
    C[yo*8:yo*8+8][xo*8:xo*8+8] = 0
    for ko in range(128):
      for yi in range(8):
        for xi in range(8):
          for ki in range(8):
            C[yo*8+yi][xo*8+xi] +=
               A[ko*8+ki][yo*8+yi] * B[ko*8+ki][xo*8+xi]
```



TVM Domain Specific Language + New Features

- Allow read and write to special memory scope
- Allow hook into hardware instructions
- Allow optimize for pipeline parallelism via reordering

+ Cache Data on Accelerator Special Buffer

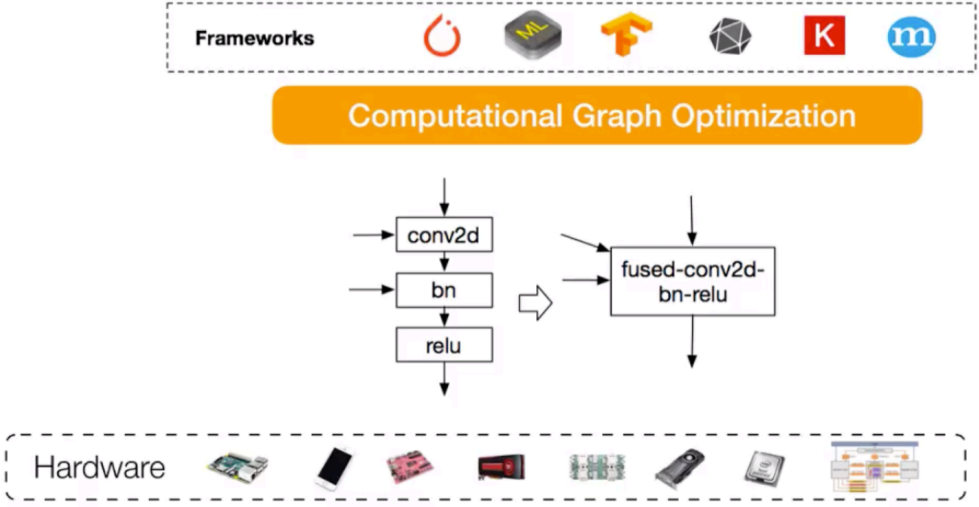
```
CL = s.cache_write(C, vdla.acc_buffer)
AL = s.cache_read(A, vdla.inp_buffer)
# additional schedule steps omitted …
```

+ Map to Accelerator Tensor Instructions

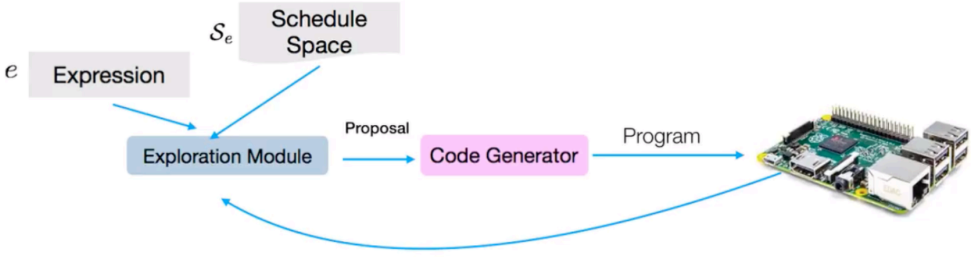
```
s[CL].tensorize(yi, vdla.gemm8x8)
```

```
inp_buffer AL[8][8], BL[8][8]
acc_buffer CL[8][8]
for yo in range(128):
  for xo in range(128):
    vdla.fill_zero(CL)
    for ko in range(128):
      vdla.dma_copy2d(AL, A[ko*8:ko*8+8][yo*8:yo*8+8])
      vdla.dma_copy2d(BL, B[ko*8:ko*8+8][xo*8:xo*8+8])
      vdla.fused_gemm8x8_add(CL, AL, BL)
    vdla.dma_copy2d(C[yo*8:yo*8+8,xo*8:xo*8+8], CL)
```

TVM: End to End Optimization

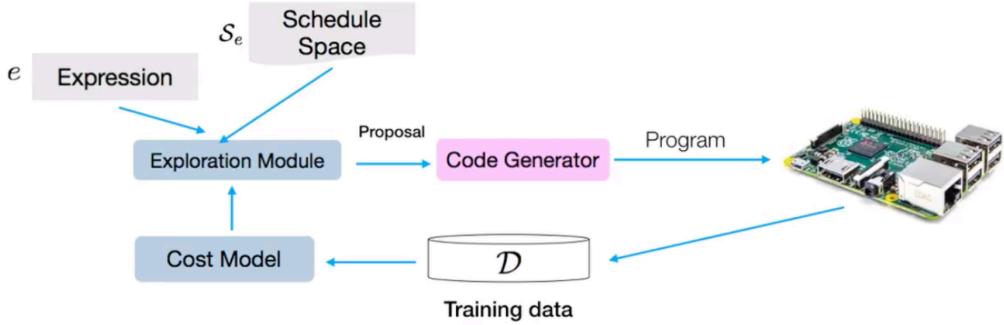


TVM: Blackbox Autotuning



High experiment cost

TVM: Statistical Cost Model based Approach

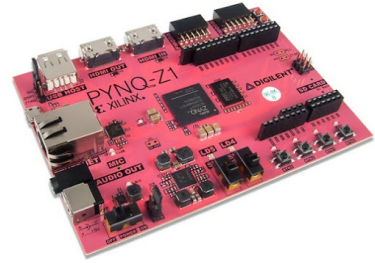


Outline

Object Detection on FPGA

- http://www.pynq.io/
- DAC-2018 System Design Contest
- Detailed Info:
  www.cse.cuhk.edu.hk/~byu/2018-DAC-HDC/

PYNQ™



Sponsored by:

dji NVIDIA® XILINX®

Overall System Diagram

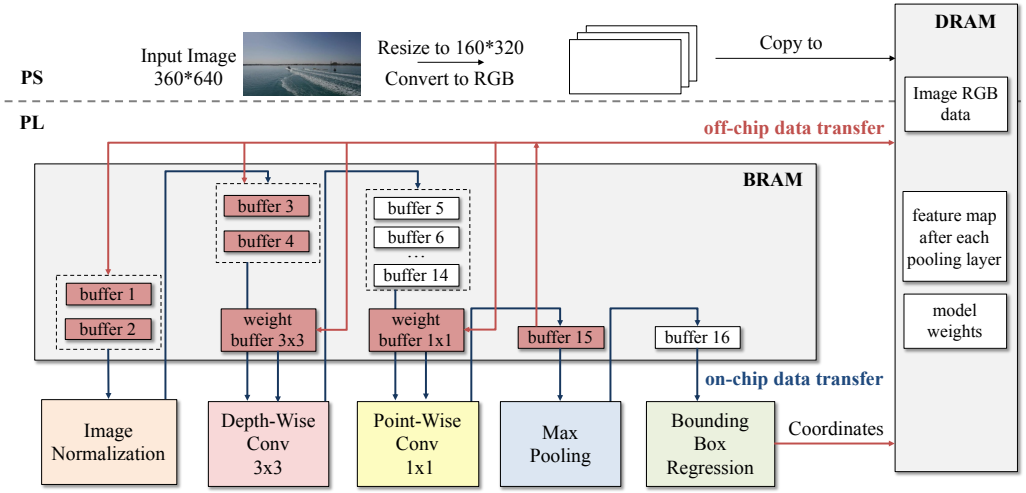
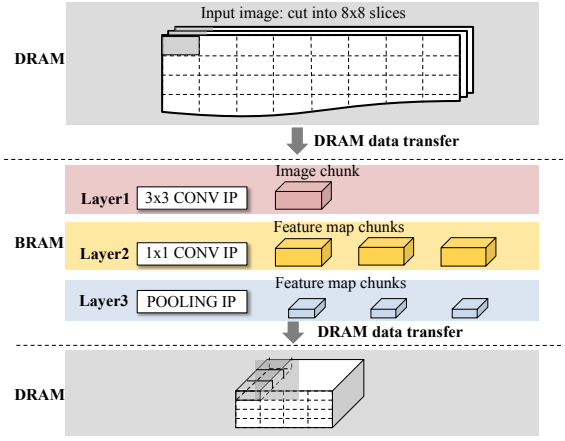
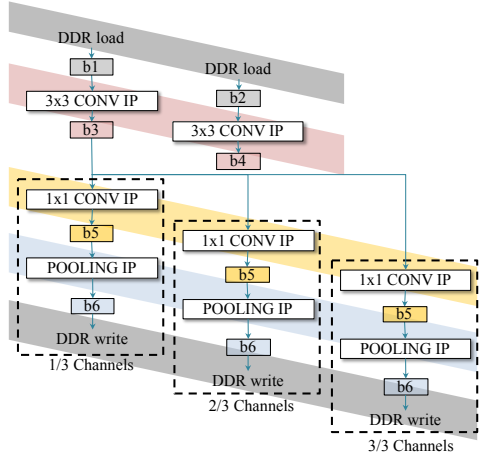


Image partition, fine grained buffer scheduling, IP pipeline



DRAM data transfer pattern



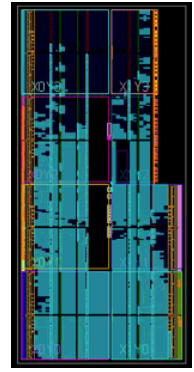
Fine Grained IP/Buffer Scheduling

FPGA Design Flow

1. DNN Design in C/C++ (`example`)
2. Generate RTL (`example`) by tool Vivado HLS

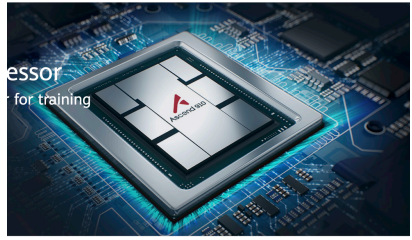
```
CONV_1x1.v                  fifo_w8_d2_A.v              mobilenet_FM_buf_bTr.v          mobilenet_fpext_3vdy.v
CONV_3x3_group.v            load_bias_from_axi.v        mobilenet_FM_buf_bTr_ram.dat    mobilenet_mac_muladEe.v
Loop_0_proc.v               load_image_chunk_bkb.v      mobilenet_IMG_m_axi.v           mobilenet_mac_mulhbi.v
Relu.v                      load_image_chunk_bkb_ram.dat mobilenet_INPUT_r_m_axi.v      mobilenet_mac_mulsc4.v
buffer_copy_from_axi.v      load_image_chunk_nor.v      mobilenet_OUTPUT_r_m_axi.v      mobilenet_mac_mulxd5.v
clear_buf.v                 load_pool3_from_axi.v       mobilenet_am_addmcq.v           mobilenet_mul_mulibs.v
compute_bounding_box.v      load_pool6_from_axi.v       mobilenet_amo_addcud.v          mobilenet_mul_multde.v
compute_engine_16.v         load_weight_2D_from_s.v     mobilenet_amo_adddEe.v          mobilenet_mux_164jbC.v
copy_to_DDR_pool3.v         load_weight_3D_from_s.v     mobilenet_amo_addeOg.v          mobilenet_sdiv_17wdI.v
copy_to_DDR_pool6.v         load_weights.v              mobilenet_amo_addg8j.v          mobilenet_uitofp_udo.v
copy_to_DDR_pool9.v         max_pooling.v               mobilenet_amo_adddkM.v          mobilenet_weight_OgC.v
dataflow_in_loop.v          mobilenet.v                 mobilenet_amo_addlbW.v          mobilenet_weight_OgC_ram.dat
dataflow_parent_loop_1.v    mobilenet_AXILiteS_s_axi.v  mobilenet_amo_addmb6.v          mobilenet_weight_yd2.v
exp_generic_float_s.v       mobilenet_FM_buf1Rg6.v      mobilenet_amo_addmcq.v          mobilenet_weight_yd2_ram.dat
exp_generic_floatqcK.v      mobilenet_FM_buf1Rg6_ram.dat mobilenet_amo_addpcA.v         my_exp_fix.v
exp_generic_floatqcK_ram.dat mobilenet_FM_buf1bDo.v     mobilenet_ap_fpext_0_no_dsp_32_ip.tcl set_bias.v
exp_generic_floatrcU.v      mobilenet_FM_buf1bDo_ram.dat mobilenet_ap_uitofp_4_no_dsp_32_ip.tcl
exp_generic_floatrcU_ram.dat mobilenet_FM_buf3bak.v     mobilenet_bias_bulhA.v
fifo_w2_d1_A.v              mobilenet_FM_buf3bak_ram.dat mobilenet_bias_bulhA_ram.dat
```

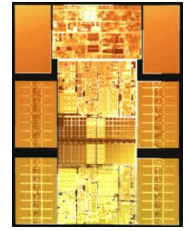
3. Generate bitstream by tool Vivado IDE
4. Load bitstream to FPGA board



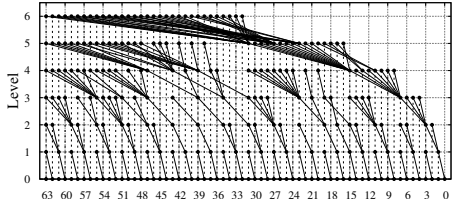
Outline

Huawei Ascend 910

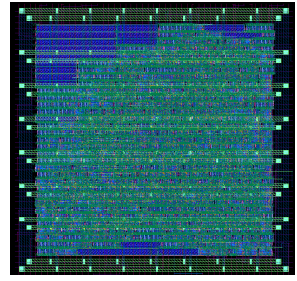




Adder is one of the most important component!

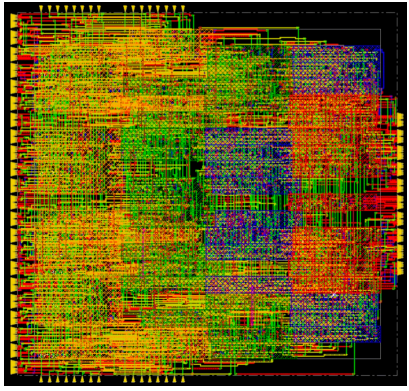


(a) Logic perspective

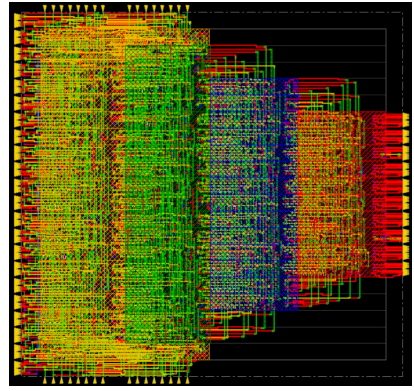


(b) Physical synthesis perspective

EDA Challenges: How to Design an AI Chip Component?



(c) Current EDA tool output



(d) Manual design

C/C++ Programming skills are heavily required – welcome students with ICPC background