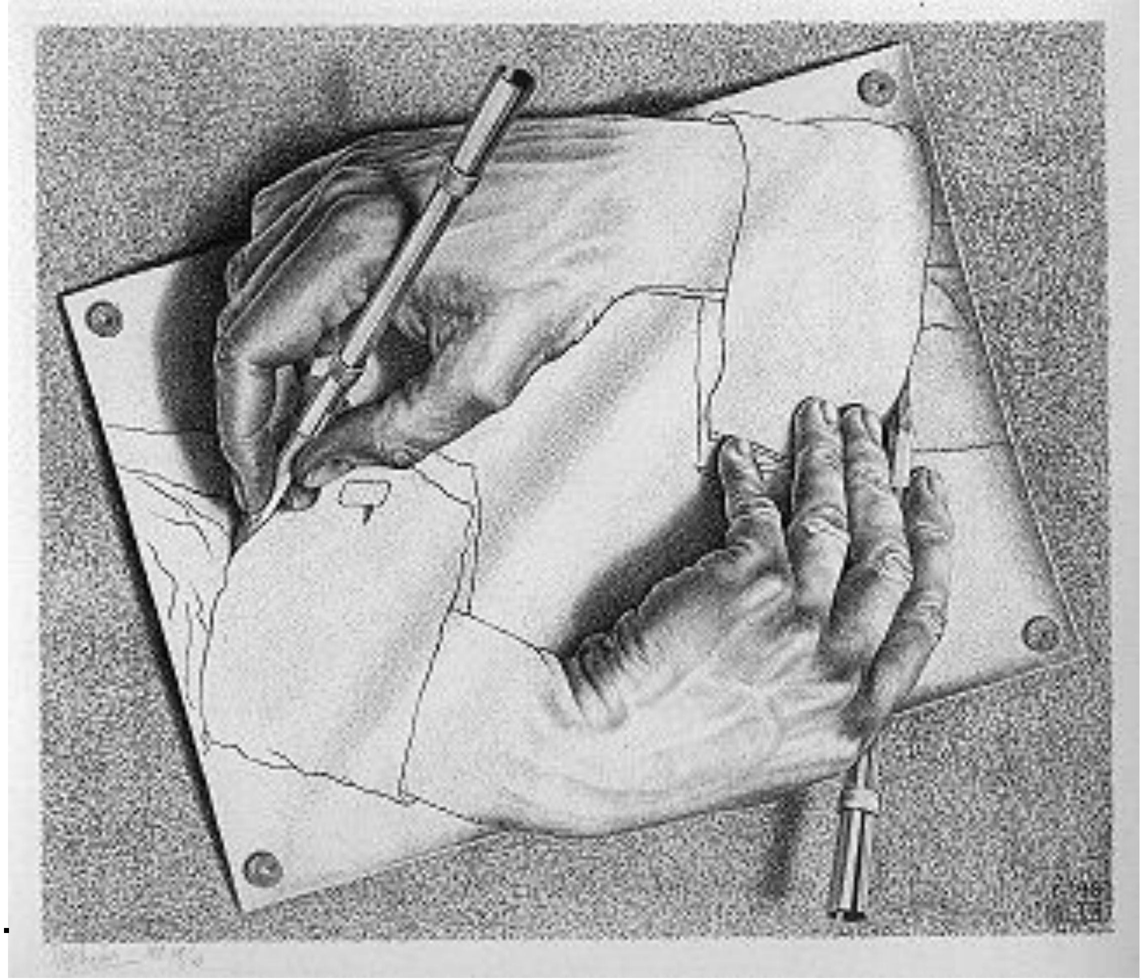
Coevolution of Neural Network and Computer Architecture

zsc@megvii.com

Aug. 2019

Propose new kind of Neural Network for hardware.

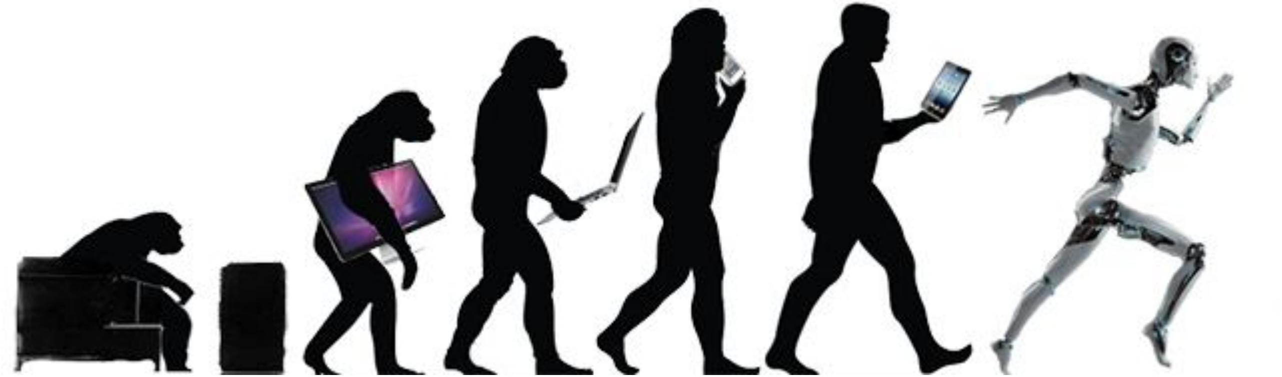


Define new hardware for Neural Network.

Software-hardware co-evolution

Deep Learning Challenge

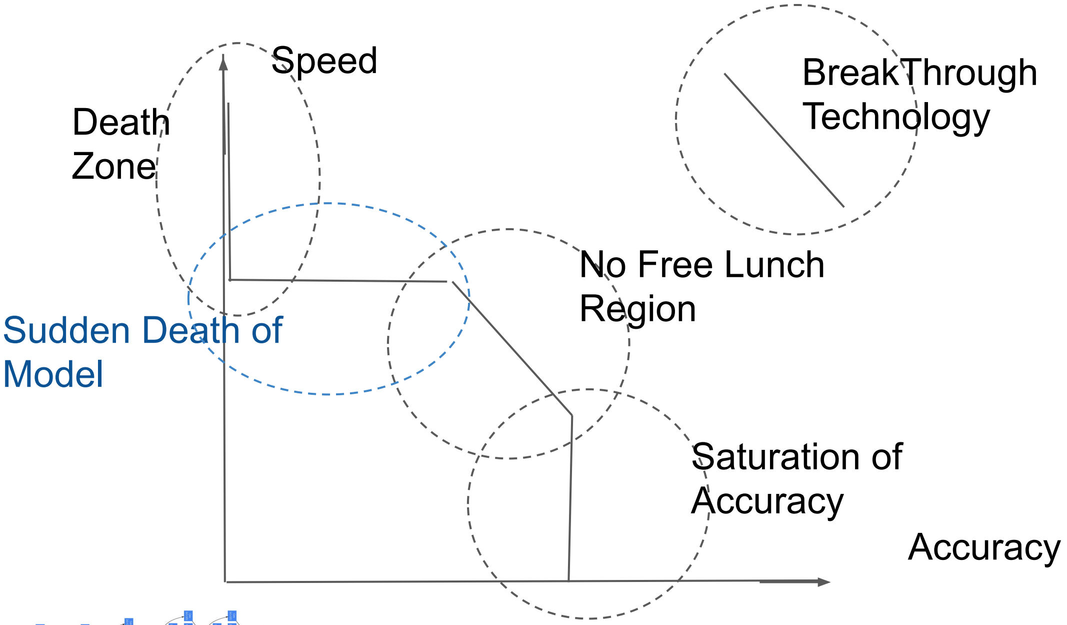
- Make it start: Conceptual Breakthrough
- Make it work: Building product
- Make it cheap: Democratize



https://medium.com/global-silicon-valley/the-evolution-of-mobile-computing-d273f23eda61

Tradeoff between Accuracy and Speed

- Breakthroughs improve both accuracy and speed
  - Factorized Convolution (GoogleNet)
  - Skip connection (ResNet)
  - Fully Convolutional Network
  - Better Loss Function
  - Batch Normalization
  - Cyclic Learning Rate



Speed

Death Zone

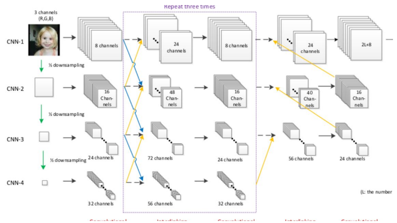
Sudden Death of Model

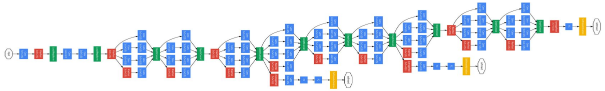
BreakThrough Technology

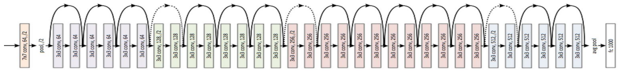
No Free Lunch Region

Saturation of Accuracy

Accuracy

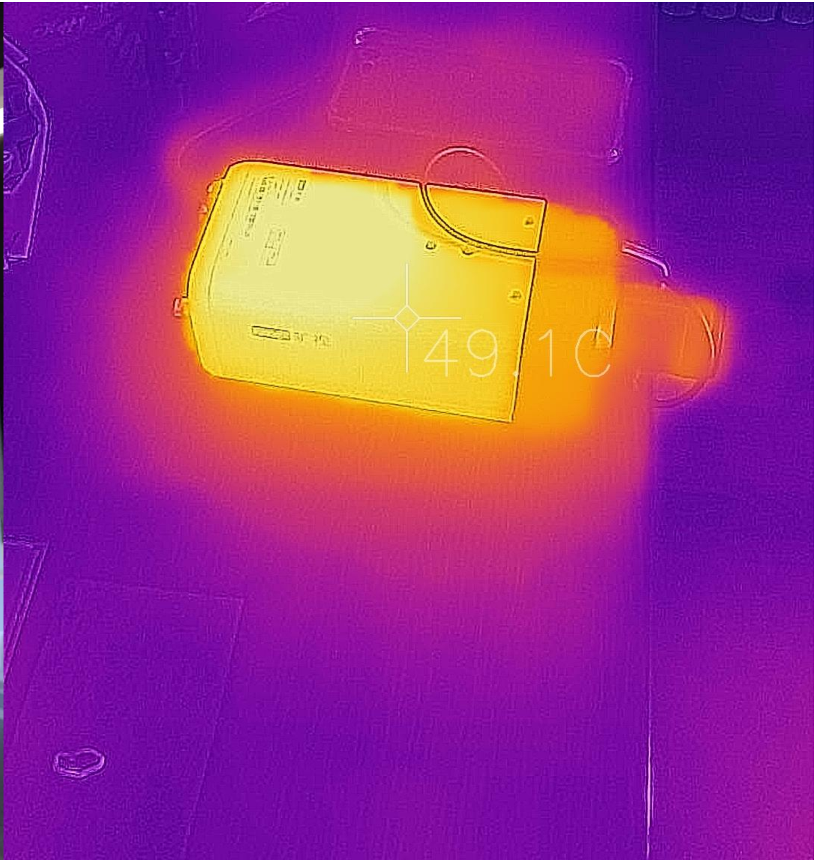






, QA: 0.81
6, QA: 0.86
7, QA: 0.87
8, QA: 0.89
2, QA: 0.86
9, QA: 0.84
10, QA: 0.84
11, QA: 0.92
12, QA: 0.88
3, QA: 0.88
13, QA: 0.9
4, QA: 0.9
5, QA: 0.9
14, QA: 0.8
15, QA: 0.88
16, QA: 0.81
17, QA: 0.86
18, QA: 0.85
19, QA: 0.81
20, QA: 0.92
21, QA: 0.9
22, QA: 0.89
23, QA: 0.87
24, QA: 0.91
25, QA: 0
26, QA: 0.89
27, QA: 0.87
28, QA: 0.92
29, QA: 0.87
30, QA: 0.9
31, QA: 0.88
32, QA: 0.88
33, QA: 0.83
34, QA: 0.89
35, QA: 0.85
36, QA: 0.83
37, QA: 0.89
38, QA: 0.9
39, QA: 0.89
40, QA: 0.89
41, QA: 0.9
42, QA: 0.84
43, QA: 0.88
44, QA: 0.9
45, QA: 0.84
46, QA: 0.9
47, QA: 0.88
48, QA: 0.85
49, QA: 0.88
50, QA: 0.84
51, QA: 0
52, QA: 0.86
53, QA: 0.83
54, QA: 0.8
55, QA: 0.8
56, QA: 0.85
57, QA: 0.92
58, QA: 0.9
59, QA: 0.85
60, QA: 0.87
61, QA: 0.8
62, QA: 0.82
63, QA: 0
64, QA: 0.86
65, QA: 0.89
66, QA: 0.87
67, QA: 0.78
68, QA: 0.9
69, QA: 0.89
70, QA: 0.89
71, QA: 0.91
72, QA: 0.84
73, QA: 0.85
74, QA: 0.9
75, QA: 0.91
76, QA: 0.9
77, QA: 0.89
78, QA: 0.9
79, QA: 0.84
80, QA: 0.8
81, QA: 0.85
82, QA: 0.87
83, QA: 0.89
84, QA: 0.86
85, QA: 0.86
86, QA: 0.9
87, QA: 0.88
88, QA: 0
89, QA: 0.8
90, QA: 0.87
91, QA: 0.88
92, QA: 0.87
93, QA: 0.87
94, QA: 0.83
95, QA: 0.8

User cases: Deep Learning



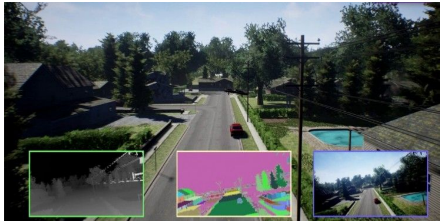
User cases: Reinforcement Learning

Characteristics: require fast & complex simulations

OpenSim

A human skeleton model for locomotive task modeling.

GTA 5
AirSim

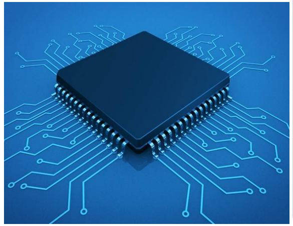


Simulation for self-driving car/ADAS and Drones.

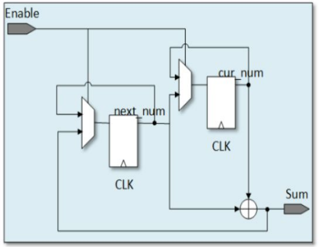
Computer Architecture answer to Deep Learning Challenge

- Make it start: Conceptual Breakthrough
    - GPU: flexible powerhouse
- Make it work: Building product
    - ISA & Programming models: Graph Compiler and Execution Engine
- Make it cheap: Democratize
    - ASIC, Edge Computing, Cloud computing: mass production of all-in-one chips

Computation Stack

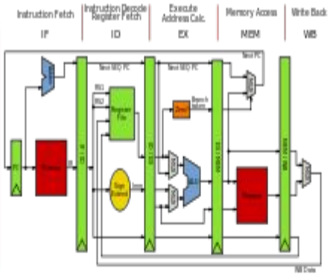


Silicon
- Partitioning & Planning
- Place & Route
- Timing Closure

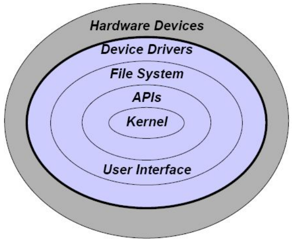


Circuit to Generate Fibonacci Series *(Fig. 13)*

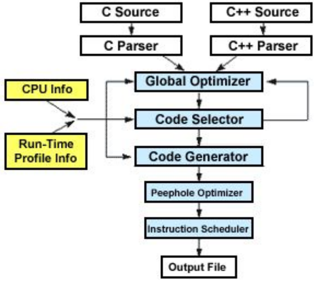
Verilog
- Karnaugh map
- Finite State Machine



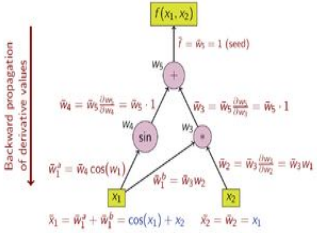
Architecture
- ISA
- Micro-code
- Resource allocation



Operating System
- Page table
- File system
- Interrupts

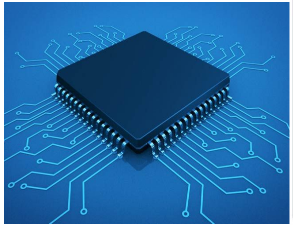


Compiler
- Parallelism mining
- Memory latency hiding


Backward propagation of derivative values
$f(x_1, x_2)$
$\bar{f} = \bar{w}_5 = 1$ (seed)
w_5
+
$\bar{w}_4 = \bar{w}_5 \frac{\partial w_5}{\partial w_4} = \bar{w}_5 \cdot 1$
$\bar{w}_3 = \bar{w}_5 \frac{\partial w_5}{\partial w_3} = \bar{w}_5 \cdot 1$
w_4
sin
w_3
*
$\bar{w}_1^a = \bar{w}_4 \cos(w_1)$
$\bar{w}_2 = \bar{w}_3 \frac{\partial w_3}{\partial w_2} = \bar{w}_3 w_1$
$\bar{w}_1^b = \bar{w}_3 w_2$
x_1
x_2
$\bar{x}_1 = \bar{w}_1^a + \bar{w}_1^b = \cos(x_1) + x_2$
$\bar{x}_2 = \bar{w}_2 = x_1$

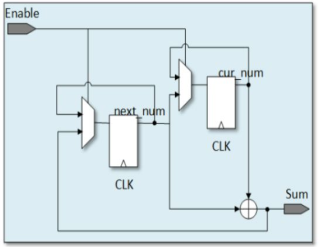
Computation Graph Engine
- Kernels
- Execution Plan

Computation Stack



Silicon

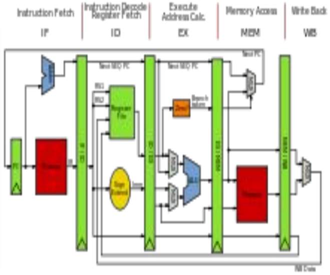
- Partitioning & Planning
- Place & Route
- Timing Closure



Circuit to Generate Fibonacci Series *(Fig. 13)*

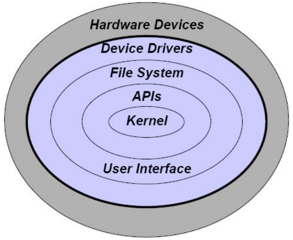
Verilog

- Karnaugh map
- Finite State Machine



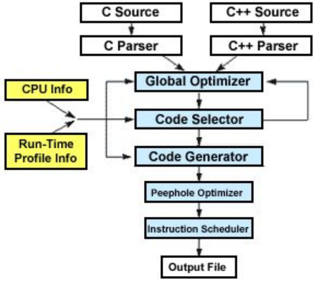
Architecture

- ISA
- Micro-code
- Resource allocation



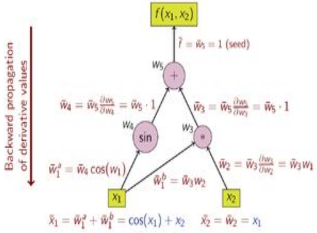
Operating System

- Page table
- File system
- Interrupts



Compiler

- Parallelism mining
- Memory latency hiding


$f(x_1, x_2)$
$\bar{f} = \bar{w}_5 = 1$ (seed)
w_5
+
$\bar{w}_4 = \bar{w}_5 \frac{\partial w_5}{\partial w_4} = \bar{w}_5 \cdot 1$
$\bar{w}_3 = \bar{w}_5 \frac{\partial w_5}{\partial w_3} = \bar{w}_5 \cdot 1$
w_4
sin
w_3
*
$\bar{w}_1^a = \bar{w}_4 \cos(w_1)$
$\bar{w}_2 = \bar{w}_3 \frac{\partial w_3}{\partial w_2} = \bar{w}_3 w_1$
$\bar{w}_1^b = \bar{w}_3 w_2$
x_1
x_2
$\bar{x}_1 = \bar{w}_1^a + \bar{w}_1^b = \cos(x_1) + x_2$
$\bar{x}_2 = \bar{w}_2 = x_1$
Backward propagation of derivative values

Computation Graph Engine

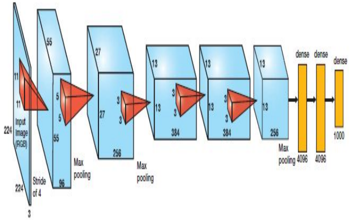
- Kernels
- Execution Plan

How will this stack deal with changes?

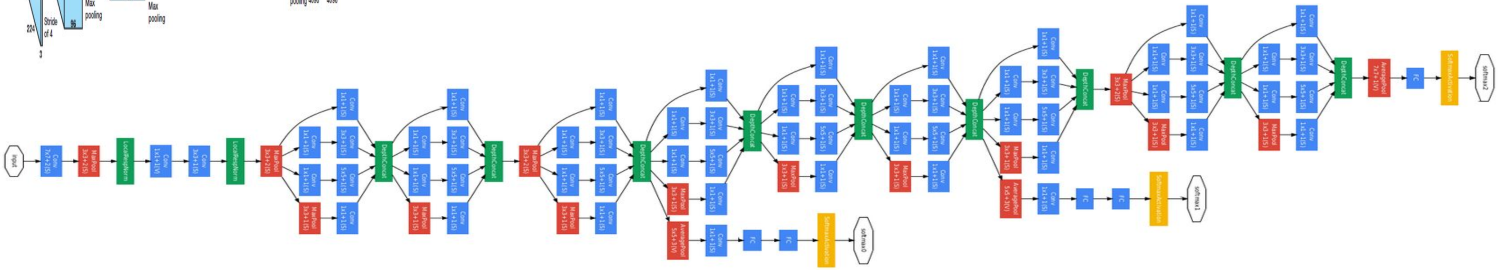
Case study: Large Neural Networks

Characteristics: many channels + side-branches + many layers

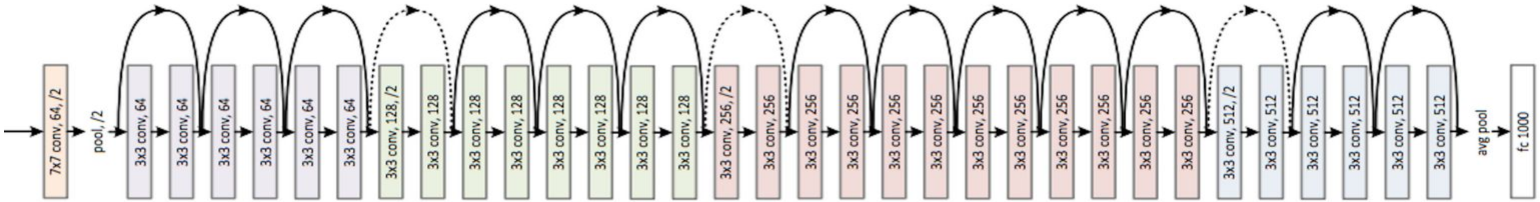
AlexNet



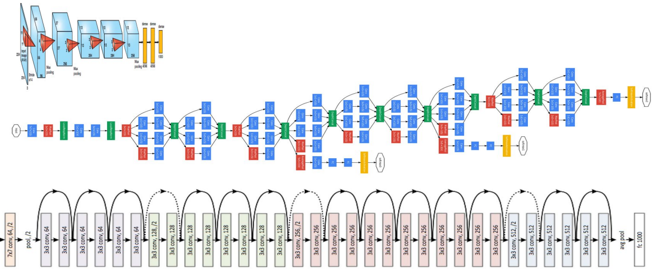
GoogLeNet



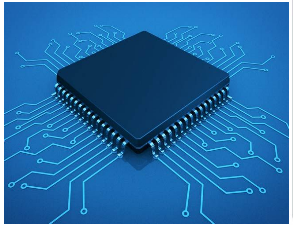
ResNet



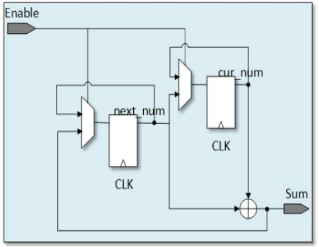
Case study: Large Neural Networks



On-Chip-Memory for caching feature maps



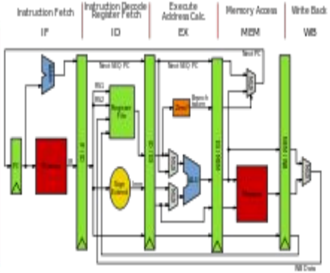
Silicon



Circuit to Generate Fibonacci Series *(Fig. 13)*

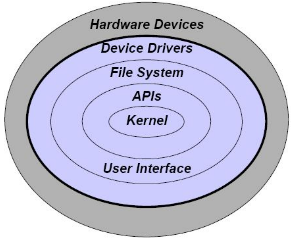
Verilog

- Instructions for convolutions & non-linearity
- Systolic Array



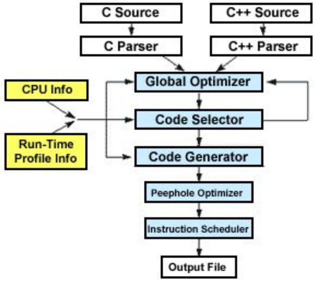
Architecture

Large page-table



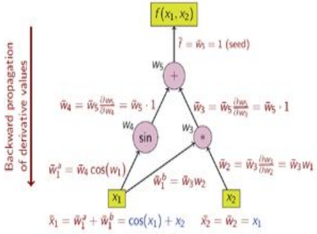
Operating System

Auto-SIMD



Compiler

Static analysis + dynamic profiling for kernel selection + execution plan

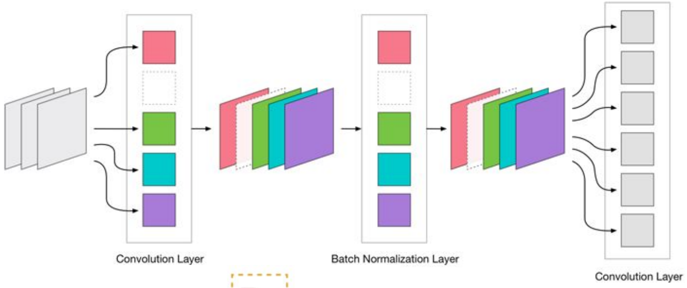


Computation Graph Engine

Case study: Small Neural Networks

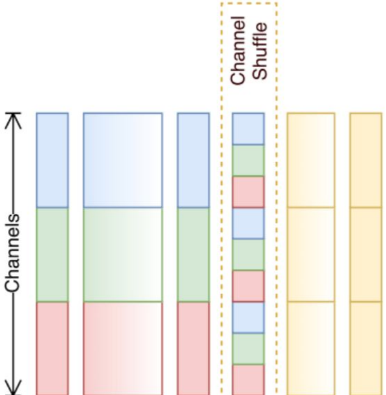
Characteristics: few channels + 1x1 convolutions

MobileNet



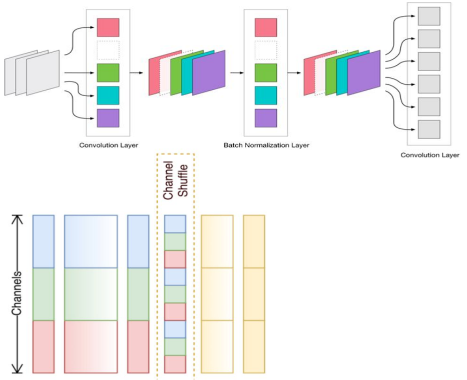
Lack of shortcut hurts its transfer learning ability.

ShuffleNet

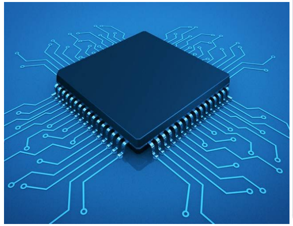


The shuffle operation is an efficient way of information mixing, but its uniqueness slows its adoption.

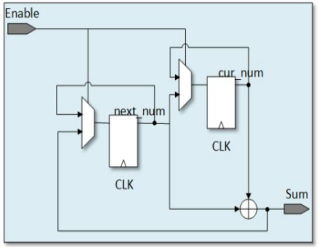
Case study: Small Neural Networks



On-Chip-Memory may be more important.



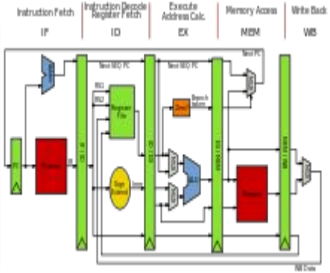
Silicon



Circuit to Generate Fibonacci Series *(Fig. 13)*

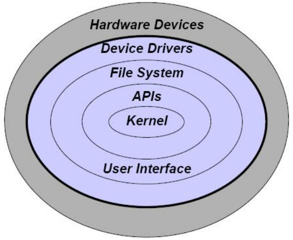
Verilog

- Specialized support for few channel layers and 1x1 convolutions.
- Different batching



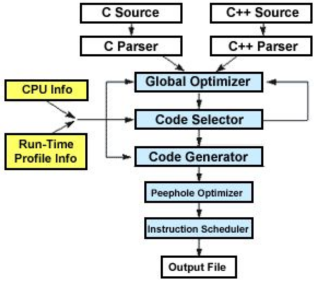
Architecture

Lower overhead



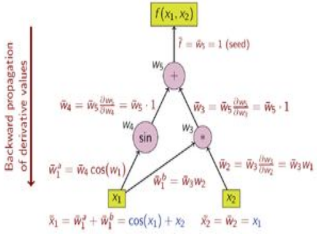
Operating System

Auto-SIMD



Compiler

Fusion of layers + handcrafted kernels


Backward propagation of derivative values
$f(x_1, x_2)$
$\bar{f} = \bar{w}_5 = 1$ (seed)
w_5
+
$\bar{w}_4 = \bar{w}_5 \frac{\partial w_5}{\partial w_4} = \bar{w}_5 \cdot 1$
$\bar{w}_3 = \bar{w}_5 \frac{\partial w_5}{\partial w_3} = \bar{w}_5 \cdot 1$
w_4
sin
w_3
*
$\bar{w}_1^a = \bar{w}_4 \cos(w_1)$
$\bar{w}_2 = \bar{w}_3 \frac{\partial w_3}{\partial w_2} = \bar{w}_3 w_1$
$\bar{w}_1^b = \bar{w}_3 w_2$
x_1
x_2
$\bar{x}_1 = \bar{w}_1^a + \bar{w}_1^b = \cos(x_1) + x_2$
$\bar{x}_2 = \bar{w}_2 = x_1$

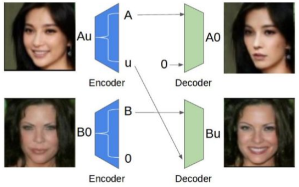
Computation Graph Engine

When a Neural Network Designers, a Computer Architect, a Compiler Expert and an OS Guru meet

- Designer wants
  - A reliable performance model
    - Open architecture design and assembly/microcode level exposure
  - Better profilers for runtime diagnostics and analyzers
  - Support for sparse matrices, dynamic operations
- Architect wants
  - Batch operations with constant delays
  - Regular memory access pattern subject to locality and many reuses
  - Streamlined memory/computation usage, no overwhelming peaks
  - Less number of operators
- Compiler Expert and OS Guru wants
  - To broker between the Designer and the Architect
    - Have a slow fallback for bizarre operators
    - Cutting peaks

IC team @ Megvii

GeneGAN



Learning2Paint

Neural Network Designer

We train our DL models and design our networks!

Computer Achitects

We build our processors and computers, from ISA to PCB!

[redacted] [redacted] [redacted]

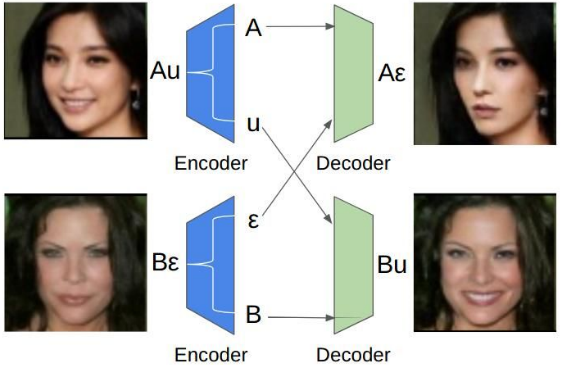
AI-product Programmers

We build our AI-products! From Javascript to Linux Kernels!

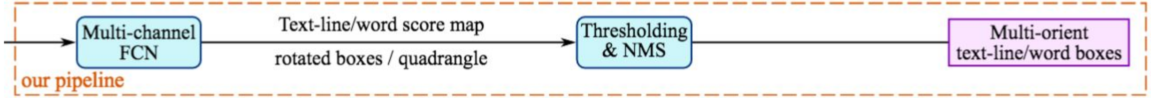
S-platform
Edge-computing platform

A-firmware
Edge Device Firmware

Neural Network Designer

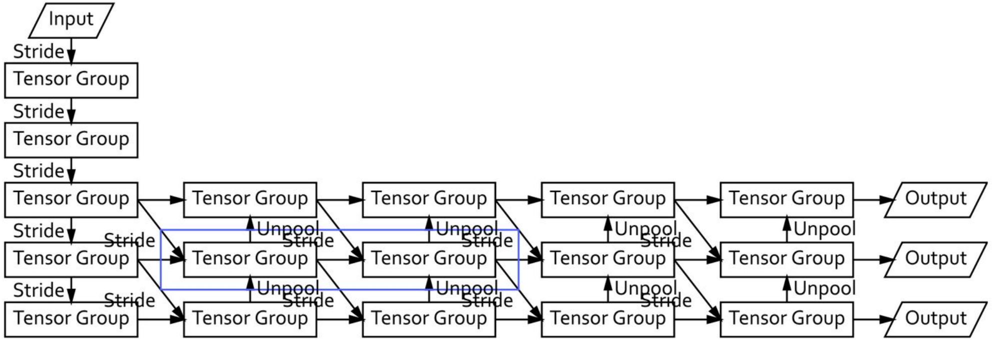


GeneGAN



EAST

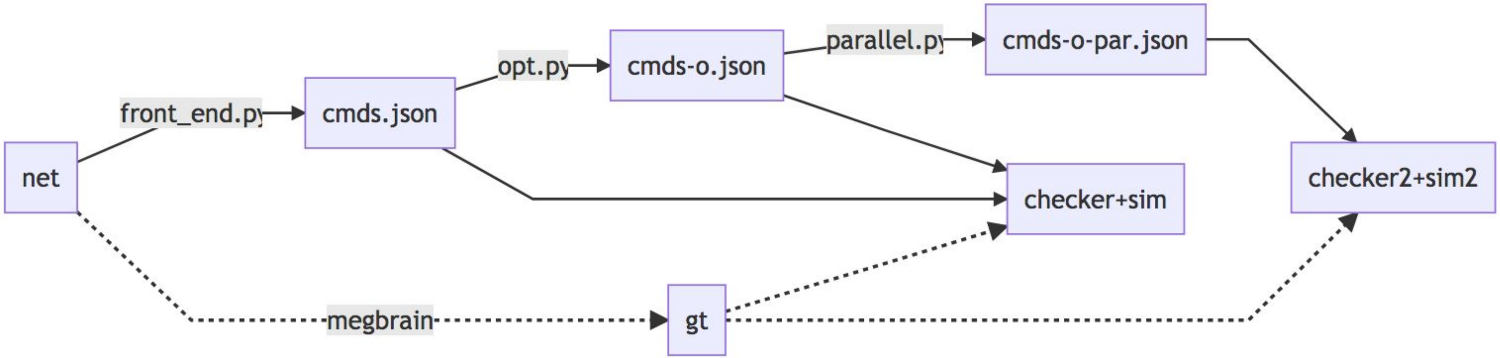
Learning2Paint



Ъ-net

Computer Architect

"X Compiler": Optimizing & Autopar Compiler

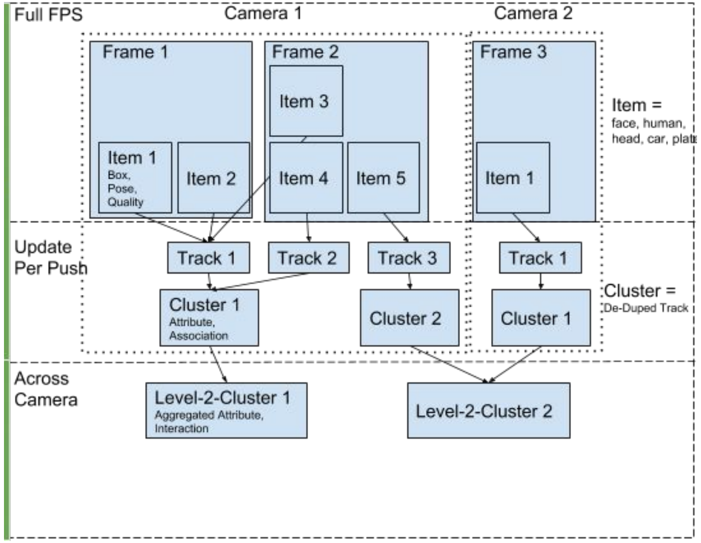


Static Scheduling

- Neural Networks are almost static
  - No branching
  - (almost always) Fixed length data: fixed input/output/intermediate size
  - Regular computation
- But there are "clouds"
  - DDR latency / bandwidth
  - Compression
  - Cache
  - Interrupts
- Dynamic Scheduling inevitable?

AI-product Programmers

S-platform: Edge-computing platform



Full FPS

Camera 1

Camera 2

Frame 1

Frame 2

Frame 3

Item 3

Item 1
Box,
Pose,
Quality

Item 2

Item 4

Item 5

Item 1

Item =
face, human,
head, car, plat

Update
Per Push

Track 1

Track 2

Track 3

Track 1

Cluster 1
Attribute,
Association

Cluster 2

Cluster 1

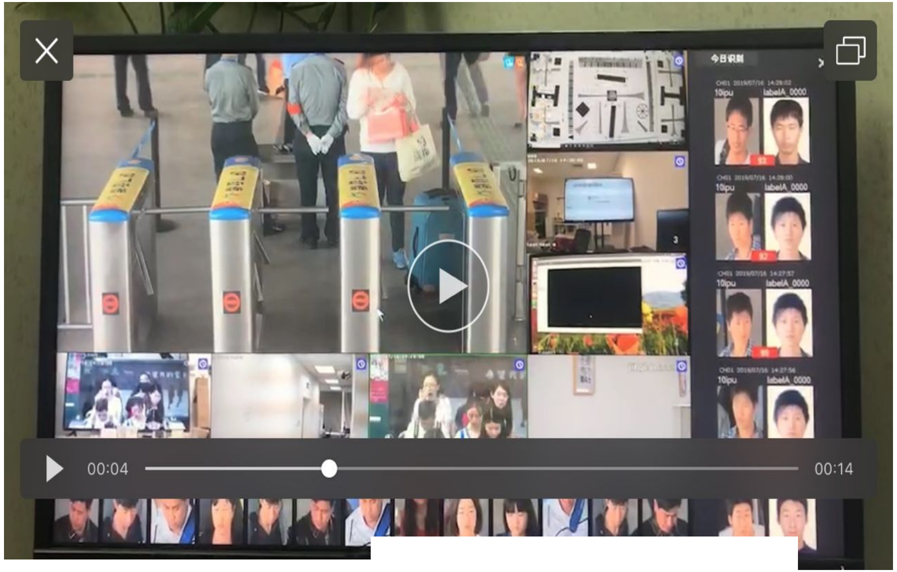
Cluster =
De-Duped Track

Across
Camera

Level-2-Cluster 1
Aggregated Attribute,
Interaction

Level-2-Cluster 2

A-firmware: Edge Device Firmware



About me

https://zsc.github.io/

● Source-to-source transformation ● Cache simulation	● Natural Language Question & Answer ● Indoor Navigation with INS ● Group Orbit Optimization	● OCR ● Quantized Neural Network ● Smart Camera ● Reinforcement Learning
Compiler Optimization	Machine Learning	Neural Network

2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019

CHINESE ACADEMY OF SCIENCES

Google

megvii