The Road to Affordable AI-Capable Products

Shaoshan Liu

shaoshan.liu@perceptin.io
Development of Silicon Valley

2020 ~

2015 ~ 2020

2010 ~ 2015

2000 ~ 2010

1980 ~ 2000

1960 ~ 1980
Overview

• Robotic Technologies
• Autonomous Driving
• Bringing Intelligence to Embedded Systems
• Bringing Intelligence to IoT Devices
• Innovations in Computing Hardware
Robotic Technologies

Anatomy
- Localization
- Scene Understanding
- Interaction

Physiology
- Sensing
- Perception
- Action

SLAM
CNN
LSTM

PerceptIn
SLAM Pipeline:

- Feature Extraction: extracts 3D feature points from images
- Propagation: tracks the location of the agent using IMU data
- Update: uses image data to correct propagation errors
- Mapping: extends the map using 3D feature points
CNN Pipeline:

- Convolution layer: extracts features from the input
- Activation layer: a function to determine whether the signal should be activated or not (e.g. sigmoid function).
- Pooling layer: reduces the spatial size of the representation
- Fully connected layer: neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks
Speech Recognition Pipeline:

- Feature extraction: converts audio signals into feature vectors
- Decoder: combines acoustic models, language models, and pronunciation dictionary to convert the feature vector into a list of words
Autonomous Driving

Sensing
- GPS/IMU
- LiDAR
- Camera

Perception
- Localization
- Object Recognition
- Object Tracking

Decision
- Path Planning
- Action Prediction
- Obstacle Avoidance

Cloud Platform
- HD Map
- Model Training
- Simulation
- Data Storage

Operating System
Hardware Platform
Autonomous Driving: Existing Implementations

Computing Platform:

- High-end CPU
- Multiple GPU cards
- Consumes ~ 3000 W of power at peak
Autonomous Driving: on mobile SoC?

**Mobile SoC:**

- Quad-core CPU @ 2.2 GHz
- Hexagon 680 DSP
- Adreno 530 GPU
- Peak power ~ 15 W

**FIGURE 2.** Performance and energy in (a) convolution and (b) feature-extraction tasks. In (a), the GPU takes only 2 ms and uses only 4.5 millijoules (mJ) to complete convolution tasks. In (b), the digital signal processor (DSP) is the most efficient unit for feature extraction, taking 4 ms and consuming only 6 mJ to complete a task.
- Localization: 25 FPS
- Object recognition: 3 FPS
- Power consumption: 11 W
- Capable of driving the vehicle at 5 mph
Autonomous Driving
Nvidia Jetson TX1:

- CPU: Quad-core ARM A57
- GPU: Nvidia Maxwell with 256 CUDA cores
- Peak power: ~ 15 W
Bringing Intelligence to Embedded Systems

- Camera (60 Hz)
- IMU (200 Hz)

**SLAM**
- <map, positions>

**Object Recognition**
- <label>

**Speech Recognition**
- <command>

**Navigation Unit**

**Reaction Unit**

**Command Unit**
Bringing Intelligence to Embedded Systems

- Localization mostly on CPU, but uses some GPU for feature extraction acceleration
- Speech recognition is CPU intensive
- Scene understanding mostly on GPU
- Combined they use 60% CPU and 70% GPU
Bringing Intelligence to Embedded Systems

- Localization consumes 4.5 W
- Speech recognition consumes 4 W
- Scene understanding consumes 7 W
- Combined they consume 11 W
Bringing Intelligence to Embedded Systems

- Resource utilization, when offloading, it only consumes about 10% of CPU
- High latency: ranging from 2 ~ 5 seconds
- Local area network: ~ 100 ms
Bringing Intelligence to IoT Devices

- Peak power ~ 5 W
- Bare-metal device
- Small memory size
Enabling Embedded Inference Engine with ARM Compute Library: A Case Study

Dawei Sun, Shaoshan Liu, Jean-Luc Gaudiot
(Submitted on 12 Apr 2017 (v1), last revised 14 Apr 2017 (this version, v3))

When you need to enable deep learning on low-cost embedded SoCs, is it better to port an existing deep learning framework or should you build one from scratch? In this paper, we share our practical experiences of building an embedded inference engine using ARM Compute Library (ACL). The results show that, contradictory to conventional wisdoms, for simple models, it takes much less development time to build an inference engine from scratch compared to porting existing frameworks. In addition, by utilizing ACL, we managed to build an inference engine that outperforms TensorFlow by 25%. Our conclusion is that, on embedded devices, we most likely will use very simple deep learning models for inference, and with well-developed building blocks such as ACL, it may be better in both performance and development time to build the engine from scratch.
• The ARM Compute Library (ACL) is a collection of low-level software functions optimized for ARM Cortex CPU and ARM Mali GPU architectures.

• ACL is targeted at a variety of use-cases including: image processing, computer vision and machine learning.
Bringing Intelligence to IoT Devices

ACL provides better heterogeneous computing support and contains no software dependencies.

25% Performance Gain

80% Reduction on Dev Time

With ACL, it took us one day to build SqueezeNet from Scratch!
Bringing Intelligence to IoT Devices
Bringing Intelligence to IoT Devices
Bringing Intelligence to IoT Devices
Bringing Intelligence to IoT Devices

- Multitasking on IoT devices
- It requires a light-weight communication protocol for different tasks to communicate
- A compiler is needed to directly convert existing model (MXNET, Caffe, TensorFlow) to executable code on different hardware
- Innovation on the computing hardware side
Innovations in Computing Hardware

Performance and Power

- Feature Extraction: 45 ms
- Propagation: 2 ms
- Update: 30 ms
- Mapping: 15 ms
- Average Power: 8 W
Innovations in Computing Hardware

Performance and Power

- Feature Extraction: 20 ms
- Propagation: 2 ms
- Update: 30 ms
- Mapping: 15 ms
- Average Power: 9 W
Innovations in Computing Hardware
Conclusions

• AI tasks are *communication-intensive* and *computation-intensive*

• Requires a good Operating System to bind tasks together

• Utilization of heterogeneous computing

• Optimization from the hardware side
Reference Materials

• Company Information: https://www.perceptin.io/blog
• Autonomous Driving Stack: https://www.oreilly.com/ideas/creating-autonomous-vehicle-systems
• Autonomous Driving Hardware: https://arxiv.org/abs/1702.01894
• Autonomous Driving Cloud: https://arxiv.org/abs/1704.02696
• NVIDIA TX1 Case Study: https://arxiv.org/abs/1705.10945
• Bring Intelligence to IoT devices: https://arxiv.org/abs/1704.03751
• SLAM Chip: https://arxiv.org/abs/1702.01295
• Robot Cloud: https://arxiv.org/abs/1704.04712