Designing Compact CNN for Embedded Stereo Vision Systems

Mohammad Loni, Amin Majd, Abdolah Loni, Masoud Daneshtalab, Mikael Sjödin and Elena Troubitsyna

School of Innovation, Design and Engineering
Mälardalen University, Sweden
Agenda

- DPAC Project
- Autonomous Systems
- Convolutional Neural Networks (CNNs)
- Processing Challenges of DNNs
- Promising Approaches to Overcome Existing Challenges
- Proposed Framework
- Results
- Conclusion
DPAC Project

- **DPAC: Dependable Platforms for Autonomous systems and Control**
  - Establishing a leading research profile targeting dependable platforms for autonomous systems and control

- **Main Goals:**
  - Facilitating close cooperation between academia and industry to achieve a significant increase in research on advanced dependable platforms for embedded systems

- **More Info:** http://www.es.mdh.se/dpac/
Autonomous Systems

- We need to acquiring information from diverse sensors for dynamically navigating and exploring environments
  - Vision (Mono/Stereo Cameras)
  - LiDAR
  - RADAR and SONAR Detectors
  - Global Positioning System (GPS)
  - Inertial Navigation System (INS)

- Stereo-Vision Camera
  - **Multimodal** Sensing way
  - Providing three-dimensional (**3-D**) information, **Luminance**, **Color**, **Distance**
GIMME2 Stereo Vision System

<table>
<thead>
<tr>
<th><strong>Frame rate</strong></th>
<th>15fps@10MP, 60fps@1080p</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processing Unit</strong></td>
<td>Xilinx Zynq 7020</td>
</tr>
<tr>
<td><strong>Programmable Logic</strong></td>
<td>Artix-7 85K Logic Cells</td>
</tr>
<tr>
<td><strong>Processing System</strong></td>
<td>Dual ARM Cortex-A9 (766MHz)</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>4Gb DDR3, 2Gb DDR3, QSPI, SD-card</td>
</tr>
<tr>
<td><strong>Communication</strong></td>
<td>2xGBE, 1xFE, 3xUSB 2.0 (one host)</td>
</tr>
<tr>
<td><strong>Power consumption</strong></td>
<td>18W@24V</td>
</tr>
<tr>
<td><strong>Physical dimensions</strong></td>
<td>130x82mm</td>
</tr>
</tbody>
</table>

(a) Left image  (b) Right image  (c) Disparity map  (d) Harris corners (G) and edges (R)
Deep Neural Networks

- However, stereo-vision cameras cannot automatically detect and recognize the objects
- Employing Deep Learning (DL) Algorithms for **recognition**, **prediction**, **decision making**, and **search**.
  - Achieving **higher accuracy** in complex problems as the scale of data increases
  - **Flexibility**
- Convolutional Neural Network (CNN) that uses convolutional layers are very effective in areas such as **image recognition** and **classification**

![Diagram of Deep Neural Networks](image)

- Pedestrian detection (distance)
- Automatic reading road signs
Dealing with big amount of raw data for modern applications

- Stereo-Vision: 10 Megapixel, 40 frame/sec.
- By 2025, embedded data (IoT) will constitute nearly 20% of all data created.

Annual Size of the Global Datasphere [1]

Data Creation by Type [1]
Processing Challenges in Big Data Era (cont.)

- Traditional CMOS scaling no longer provides performance and efficiency gains due to the failure of Dennard scaling and Moore’s law [2].

- Increasing the complexity of DL algorithms for achieving better accuracy [3].

- By prevailing the IoT and also emerging DL algorithms, the projected growth in the data-set sizes and algorithmic complexity is expected to further increase the computational requirements [3].
Promising Approach

● Using High Performance Cloud Infrastructure

● We aim to find a near-sensor processing solution which is vital for autonomous cars since sometimes we need to keep processing close to sensors due to:
  ● Low Communication Bandwidth
  ● Battery Life
  ● Privacy Constraints
  ● Real-time Application

● Approximate Computing: basically DL applications can provide considerable energy/ performance utilization opportunity by approximate execution due to the intrinsic error-resilient nature of DNNs

● ML Accelerators: There have been proposed various accelerator based designs to be more efficient than general purpose implementations
  ● ASIC, GPU, FPGA
We aim to tackle these challenges by providing a framework which generates synthesizable accelerators for CNNs.

**Key Objective:** Trying to avoid exposing software developers to the details of hardware design.

**Front-end:** Is responsible for generating an accurate and optimized CNN from a data set or approximate code regions identified by the programmer.

**Back-end:** Efficient implementation of generated CNN on different COTS processing platforms.

---

### The Proposed Framework

**GIMME2**

- **Input:** Data from Left Camera (RGB Image)

**Frontend (Software):**

- **CGP (Network Accuracy and Network Size):** Selection, Mutation, Offspring
  - **Calculate Objectives:**
  - **User’s Criterion:** Error, Size < UR
  - **Keras:** Trained Near-Optimal DCNN
  - **Next Generation:**

**Backend (Hardware):**

- **Hardware Specifications:** Accelerator Generation
  - **GPU:**
Frontend: Designing a Near-Optimal CNN Architecture

- Using a **multi-objective** optimization strategy (MO) to optimally search the design space of DNNs
  - **Validation Accuracy**
  - **Network Complexity**: total number of trainable parameters

- The Cartesian Genetic Programming (CGP) is used for encoding the genome type by representing the CNN architecture in a directed graph that is mapped in a two-dimensional matrix.

---

![Diagram](image)

*Fig. 4. (a) The genome type example representing network architecture. (b) The architecture of ResBlock and ConvBlock.*
Selection Policy

- Employing one point mutation operator to modify genome parameters.

**Table: Parameter Range for Node Blocks and CGP**

<table>
<thead>
<tr>
<th>Training Parameters</th>
<th>Epoch</th>
<th>Learning Rate</th>
<th>Batch Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>0.0005</td>
<td>256</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CGP Parameters</th>
<th># Columns</th>
<th># offsprings</th>
<th># Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>2</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network Node Blocks</th>
<th>ConvBlock ((a,b))</th>
<th>ResBlock ((a,b))</th>
<th>Activation</th>
<th>Convolution_3x3 ((a,s))</th>
<th>Pooling (Max, Average)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(a \in 32,64,128)</td>
<td>(a \in 32,64,128)</td>
<td>relu [31]</td>
<td>(a \in 32,64)</td>
<td>2x2, 3x3</td>
</tr>
<tr>
<td></td>
<td>(b \in 3x3x5)</td>
<td>(b \in 3x3x5)</td>
<td></td>
<td>(s \in 1,2,3)</td>
<td></td>
</tr>
</tbody>
</table>

\(a\): # Output Channels; \(b\): Kernel Size; \(s\): Stride
\[ \text{net}_{-}\text{param} = \left( \frac{\text{max}_{-}\text{param} - \text{net}_{-}\text{param}}{\text{max}_{-}\text{param}} \right) \quad (1) \]

\[ \text{Score} = \alpha \times (\text{net}_{-}\text{param}) + \beta \times (\text{val}_{-}\text{acc}) \quad (2) \]

• **Output:** a set of Pareto frontiers including improved architectures
Neural Architecture Search Algorithm

Algorithm 1: Pseudo Code of Multi Objective CGP

Input:  \( G \): Max. Number of Iterations, \( H \): Possible Genome parameters
Output: An Optimal Network Architectures

Function Search \((G, H)\):

\[ P_0 = \text{Random.Parent} (H); \] //Creating initial random parent
\[ t = 1; \]
\[ \text{while } (t < G) \text{ or (achieved near-optimal net.) do} \]

Fitness Function \((P_0)\); //Calculating the fitness function of parent
\[ U_0 = \text{Force_Mutation} (P_0); \] //Generating the offspring population by doing force mutation operation
Fitness Function \((U_0)\);
\[ U_1 = \text{Force_Mutation} (P_0); \]
Fitness Function \((U_1)\);
\[ P_0 = \text{Max} (U_0, U_1, P_0); \] //Replacing the best offspring with the parent

return \( P_0 \);

- To increase the speed of search algorithm, we just partially train architectures with 30 epochs since we got roughly 90% of maximum accuracy after this number
Results

- **Training Datasets**
  - **CIFAR-10**: This is a complex colorful benchmark dataset of natural images which is used for object recognition contain 50000 and 1000 images for train and test steps, respectively.

- Getting the **total execution time** as the evaluation metric since communication time is vital for embedded implementations.

- We also did not use any network compression technique to only assess the influence of network architecture on inference time.

- **Initial Configuration**: **epoch**=30, **batch size**=128, **number of generations**=5, **initial population**=2.

- **Back-end side**:
  - Keras automatically uses cuDNN to compile a neural network for GPU.

---

<table>
<thead>
<tr>
<th>Platform</th>
<th>CPU</th>
<th>GPU</th>
<th>ARM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Frequency (GHz)</strong></td>
<td>2.9</td>
<td>1.178</td>
<td>1.9</td>
</tr>
<tr>
<td><strong>Technology (nm)</strong></td>
<td>14</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td><strong>TDP (W)</strong></td>
<td>45</td>
<td>300</td>
<td>5</td>
</tr>
<tr>
<td><strong>Cores/Total Thread</strong></td>
<td>4/8</td>
<td>4096 CUDA Cores</td>
<td>8/8</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>8MB Cache</td>
<td>16GB GDDR5</td>
<td>2.5MB Cache</td>
</tr>
<tr>
<td><strong>Approx. Price (USD)</strong></td>
<td>378$</td>
<td>7,532$</td>
<td>60$/board</td>
</tr>
</tbody>
</table>

---

CPU Core i7-78
GPU Tesla M60
ARM Cortex-A1₂
• Improvement proceeding of convergence metric (score) and network FLOPS for the best child in each iteration

• Improvement proceeding of accuracy and loss for the best child in each iteration.
Classification Results (CNN)

- **CIFAR-10** (Compare to the most accurate):
  - 47.14x compression Rate, 10% accuracy loss for DCNN-Arch.2
Implementation Results on CPU and ARM

- CPU execution time

- ARM execution time

- ODROID XU4 ARM A15 board
Deep convolutional neural networks are complex processing models which their implementation is challenging especially on embedded devices.

Stereo vision applications are one domain which suffer from the implementation barriers.

To tackle these challenges, we proposed a multi objective CGP approach which automatically design a highly optimized DCNN for multi/many core SoCs.

Evaluating the impact of framework for FPGA platform will be remain as the future work.
Thank you!