tinyML® Asia

Enabling Ultra-low Power Machine Learning at the Edge

November 29 & 30, 2022

www.tinyML.org
Hardware-aware model optimization in Arm Ethos-U65 NPU

Shinkook Choi | NetsPresso Core Research Lead
November 30, 2022
Table of Contents

• Problem statement
• Arm Ethos-U65
• NetsPresso
• Results
Problem statement
Problem statement

- The number of float-point operations (FLOPs) is a widely used metric to measure computation complexity. However, FLOPs is not the same as the direct metric such as speed or latency.
- The figure below shows that networks with similar FLOPs have different speeds.

Problem statement

- On various machine learning accelerators, not all FLOPs and the number of trainable parameters have the same efficiency.
- Fused-IBN may run the same as fast as a depthwise-IBN even with 7× as many Parameters.

Problem statement

Latency Staircase pattern

• When other hyperparameters are fixed and only \( C_{out} \) varies, the computation and memory complexity of a Convolution operator is linear. Therefore, the measured latency should also have a linear relationship with \( C_{out} \).

• The latency plot with respect to convolution configurations shows a staircase pattern with certain step sizes.

![Figure 4. The latency of Conv shows a step/staircase pattern with \( C_{out} \). Configuration: \( H \times W = 28 \times 28, C_{in} = 320, K = 3, S = 1 \). The X-axis is \( C_{out} \) (different intervals to better show the pattern) while the Y-axis is the latency in milliseconds.](image)

• It is important to analyze the hardware and design the neural network accordingly.

Arm Ethos-U65 NPU
Arm Ethos-U65 NPU

Arm Virtual Hardware (AVH)

- Arm Virtual Hardware (AVH) is an evolution of Arm’s modeling technology delivering models of Arm-based processors, systems, third party hardware for application developers and SoC designers to build and test software before silicon and hardware availability.

- **Fixed Virtual Platform (FVP)**
  - digital twin of a development board with Ethos-U65 & Cortex-M55

- **Corstone-300 available as part of Arm Virtual Hardware**

- **Device setting**
  - Arm Ethos-U65 NPU
  - Optimize: Performance
  - 256 MAC units
  - Memory mode: Dedicated SRAM
  - System: High End
Arm Ethos-U65 NPU

- Device dedicated for Neural processing

- Consists of:
  - Computation Units
    - MAC Unit
    - Elementwise Engine
  - Memory
    - Internal Memory
    - External Memory: SRAM, DRAM
When memory mode is dedicated SRAM, the arena cache size is 384 KiB.

If the activation buffer size is larger than 384 KiB, DRAM is used and latency increases dramatically.

Since the memory size is 32 MiB, it cannot be executed if the sum of the model size and the activation buffer size is greater than 32 MiB.
Arm Ethos-U65 NPU

**Convolution types**

- Input → Conv → Output
- Input → PW → DW → Concat → Output
- Input → Split → Conv → Split → Conv → Concat → Output
- Input → Split → PW → DW → Output
- Input → PW → DW → Output

**Neural Network Blocks**

- **Group Conv**
  - Input → Conv → Concat → Output

- **Ghost Conv**
  - Input → PW → DW → Concat → Output

- **Blueprint Separable Conv (BS Conv)**
  - Input → Split → Conv → Split → Conv → Concat → Output

- **Inverted Bottleneck (IBN)**
  - Input → PW → DW → PW → Output

- **Fused group convolution IBN (Fused-GC-IBN)**
  - Input → Split → Conv → Split → Conv → Concat → PW → Output

- **maxpool-IBN**
  - Input → PW → maxpool → Output

**Building Blocks**

- **Input**

**Notes**

Arm Ethos-U65 NPU

Neural Network Blocks

• Latency characteristics of blocks vary on the configuration

**Convolution types**

![Latency of different convolution types](chart)

**Building Blocks**

![Latency of Various IBNs (Normalized to IBN)](chart)

* The value of the horizontal axis represents the value of $height \times width \times channels$ when the size of the input/output feature map is the same.

© 2022 Nota Inc. All Rights Reserved.
Arm Ethos-U65 NPU

Latency Staircase pattern

- **Step size of latency staircase pattern**
  - Documentation

<table>
<thead>
<tr>
<th>Operation</th>
<th>Kernel size</th>
<th>OFM height</th>
<th>OFM width</th>
<th>OFM depth</th>
<th>IFM depth</th>
<th>MACs per cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2D (depth first)</td>
<td>(kh \times kw = 1 \times k)</td>
<td>2*(h)</td>
<td>2*(w)</td>
<td>8*(d)</td>
<td>32*(n)</td>
<td>256</td>
</tr>
</tbody>
</table>

- **Experiment**
  - Latency staircase pattern with input channels
  - Latency staircase pattern with output channels
NetsPresso

Hardware-Aware AI Model Optimization

Input

Target Performance

Target Hardware

NetsPresso

Model Searcher

Model Compressor

Model Launcher

Output

Optimized AI Model
NetsPresso

Model Compressor

• Supports all CNN architectures
• Optimal compression ratio is recommended
• Eliminates months of paper implementation time
• Minimal loss of information
**NetsPresso**

**Compression Methods**

- **Structured pruning**
  - Removes entire neurons, filters, or channels and returns a model, which does not require any particular hardware or software to be accelerated.

- **Pruning by Criteria**
  - L2-Norm is used to represent the importance of the corresponding filter.
  - In other words, this method prunes filters based on the magnitude of weights.

\[
C_{Tr}ch_{out} = \sqrt{\sum_{i \in ch_{in}} (|W^i|_{ch_{out}})^2}
\]

- **Recommendation**
  - The Recommendation enables a so-called global pruning, which measures the layer-wise importance for structured pruning and allocates the pruning ratio for each layer at ease.
NetsPresso

Compression Methods

• **Filter decomposition**
  - Filter decomposition is to approximate original weights into lightweight representations via low-rank approximations to reduce the computational cost.

• **Tucker decomposition**
  - Tucker decomposition decomposes the convolution with a 4D kernel tensor into two factor matrices and one small core tensor.

• **Recommendation**
  - The Recommendation allows the user to set proper compress hyperparameter of the filter decomposition, in-rank, and out-rank.
NetsPresso

Compression Methods

Structured Pruning

Filter Decomposition
NetsPresso

Latency Staircase pattern

- **Hardware-aware optimization for the latency staircase pattern**
  - If the value recommended by the Recommendation of structured pruning and filter decomposition is located in the middle of the latency staircase pattern, it should be set to a multiple of the step size.
  - The neural network capacity can be increased by increasing the number of operations and parameters for the same latency.
Results
Results

Image classification

- On Imagewoof
  - image size = 224 × 224

- Because the sum of model size and activation buffer size of the baseline VGG19 is larger than 32MiB, Ethos-U65 could not run the model.

- Nota's NetsPresso reduced the size of VGG19 to ~50% so that Ethos-U65 can run the model.

<table>
<thead>
<tr>
<th>Network</th>
<th>Type</th>
<th>Accuracy (%)</th>
<th>Macs (M)</th>
<th>Params (M)</th>
<th>Model size (MiB)</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG19</td>
<td>Original</td>
<td>88.39</td>
<td>19527.4</td>
<td>32.9</td>
<td>32.0</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Nota</td>
<td>87.99</td>
<td>6914.7</td>
<td>18.4</td>
<td>18.0</td>
<td>57.41</td>
</tr>
</tbody>
</table>
Results

Image classification

- **On CIFAR-100**
  - image size = 32 × 32

- Nota's NetsPresso reduced the latency of MobileNet V1 up to 5x with a 0.25% drop in accuracy.
- By applying hardware-aware optimization for the latency staircase pattern, Nota’s NetsPresso marginally improved the accuracy with the same latency.

<table>
<thead>
<tr>
<th>Network Type</th>
<th>Type</th>
<th>Accuracy (%)</th>
<th>Macs (M)</th>
<th>Params (M)</th>
<th>Model size (MiB)</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobileNet V1</td>
<td>Original</td>
<td>66.36</td>
<td>46.5</td>
<td>3.33</td>
<td>3.53</td>
<td>1.43</td>
</tr>
<tr>
<td></td>
<td>Nota</td>
<td>66.11</td>
<td>8.94</td>
<td>0.36</td>
<td>0.53</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>Nota_HO</td>
<td>66.45</td>
<td>9.69</td>
<td>0.37</td>
<td>0.49</td>
<td>0.27</td>
</tr>
</tbody>
</table>

* HO: Hardware-aware Optimization
Where to try

netspresso.ai

The Hardware-aware AI Model Optimization Platform

Automatic search, compression, and deployment of models optimized for your target device.

Learn More

Cost-effective AI
Enhanced Model Performance
Improve development efficiency
Where to try

netspresso.ai

Arm Virtual Hardware in NetsPresso
Copyright Notice

This presentation in this publication was presented at tinyML® Asia 2022. The content reflects the opinion of the author(s) and their respective companies. The inclusion of presentations in this publication does not constitute an endorsement by tinyML Foundation or the sponsors.

There is no copyright protection claimed by this publication. However, each presentation is the work of the authors and their respective companies and may contain copyrighted material. As such, it is strongly encouraged that any use reflect proper acknowledgement to the appropriate source. Any questions regarding the use of any materials presented should be directed to the author(s) or their companies.

tinyML is a registered trademark of the tinyML Foundation.

www.tinyML.org