Fused Depthwise Tiling for Memory Optimization in TinyML Deep Neural Network Inference

Rafael Stahl, Daniel Mueller-Gritschneder, Ulf Schlichtmann
Technical University of Munich
School for Computation, Information and Technology
Chair for Electronic Design Automation
Burlingame, 27th of March 2023

This work was supported in part by the German Federal Ministry of Education and Research (BMBF) within the project Scale4Edge under contract no. 16ME0131.
Machine Learning on Edge Devices

- Focus: **Inference**

- Improves:
  - Communication demand
  - Latency
  - Data privacy

- Many application suitable for extreme low-power: **tinyML**
  - Keyword Spotting
  - Visual Wake-up
  - Anomaly Detection
  - Radar Gesture Detection
Challenge: Memory

- Power usage
- Cost

- Reducing memory with accuracy trade-off:
  - Quantization
  - Pruning
  - Network Architecture Search (NAS)
Intermediate Buffers

Memory requirements are dominated by few intermediate buffers

→ Fused Tiling
Loop Tiling

- Loop transformation to exploit spatial and temporal locality
- Typically employed for performance optimization

Fused Tiling

Tiling:
- Compute large intermediate buffer in multiple tiles

Fusion:
- Operator fusion decouples their lifetime
  - Lifetimes of split large buffers do not overlap
  - Their storage buffers may overlap
  - Memory reduction
Fused Feature Map Tiling (FFMT)

- Operator fusion through spatial locality of convolution
- Introduces overlap from kernel size
- Does not support operations with large input dependencies
  - Fully connected
  - Convolutions with very large kernel sizes
Fused **Depthwise** Tiling (FDT)

- Allows fusion of two operations with large input dependencies
- Accumulates *partial sum* in second output
- Requires **Merge** operation
- New tiling opportunities
- No significant run time overheads
FDT for Convolutions

Split by feature maps instead of neurons

\[ \downarrow \text{Conv}(3x3, \ ch=8) \]

\[ \downarrow \text{Conv}(3x3, \ ch=2) \]
End-to-end Deployment Flow

- Determines where, and how to apply fused tiling
- Memory-aware scheduling
- Memory buffer layout planning
- Path discovery
Path Discovery

- Finds an optimized sequence of operations (**path**) for fused tiling
- Intermediate buffers and operations are matched according to **blocks**

Untiled

200 → conv → 100 → conv → 1000 → pool → 250

100 → conv → 50 → conv → 100 → conv
Path Discovery – FDT
Path Discovery – FFMT
Implementation

- Implemented in Apache TVM
  - Suitable for complex transformation passes

- Evaluated seven quantized models in RISC-V RV32GC
  - Memory usage from sections of compiled binary
  - Performance estimation from multiply ops of optimized graph

- To appear open source at: https://github.com/tum-ei-eda/moiopt
## Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Mem [kB]</th>
<th>[%]</th>
<th>MACs [1 million]</th>
<th>[%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Untiled</td>
<td>FFMT</td>
<td>FDT</td>
<td>FFMT</td>
</tr>
<tr>
<td>KWS</td>
<td>65.6</td>
<td>65.6</td>
<td>53.7</td>
<td>0.0</td>
</tr>
<tr>
<td>TXT</td>
<td>18.6</td>
<td>18.6</td>
<td>4.43</td>
<td>0.0</td>
</tr>
<tr>
<td>MW</td>
<td>17.6</td>
<td>7.04</td>
<td>11.3</td>
<td>60.9</td>
</tr>
<tr>
<td>POS</td>
<td>9.35k</td>
<td>5.11k</td>
<td>8.94k</td>
<td>45.3</td>
</tr>
<tr>
<td>SSD</td>
<td>14.3k</td>
<td>8.66k</td>
<td>12.2k</td>
<td>39.4</td>
</tr>
<tr>
<td>CIF</td>
<td>179</td>
<td>76.7</td>
<td>170</td>
<td>57.1</td>
</tr>
<tr>
<td>RAD</td>
<td>36.2</td>
<td>26.7</td>
<td>29.4</td>
<td>26.3</td>
</tr>
<tr>
<td><strong>Avg.</strong></td>
<td></td>
<td>32.7</td>
<td>24.7</td>
<td></td>
</tr>
</tbody>
</table>
Summary

- Applied **Fused Depthwise Tiling** to DNN graphs for memory optimization
- Built end-to-end deployment flow for evaluation

→ Reduces memory usage where previously not possible
→ Adds alternative solution where existing tiling causes too much performance overhead
Copyright Notice

This presentation in this publication was presented at the tinyML® Research Symposium (March 27, 2023). 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