tinyML® Research Symposium

Enabling Ultra-low Power Machine Learning at the Edge

March 27, 2023

www.tinyML.org

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

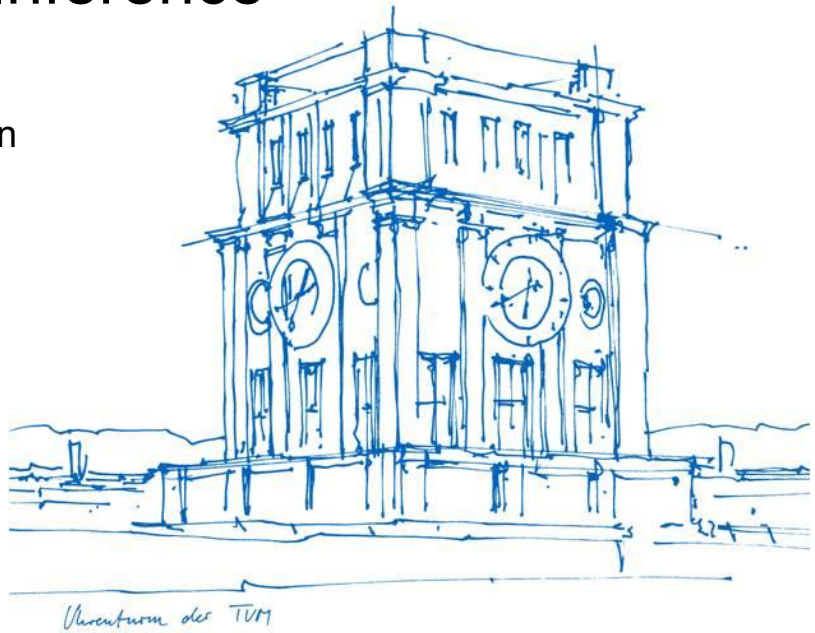
Rafael Stahl, Daniel Mueller-Gritschneder, Ulf Schlichtmann

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.

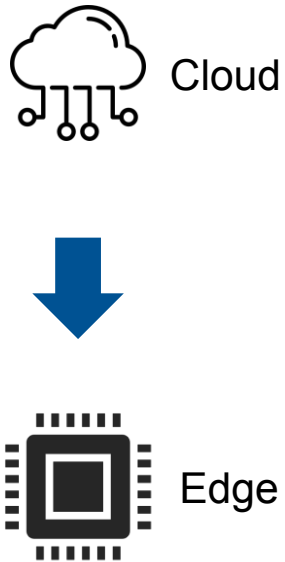
SPONSORED BY THE
Federal Ministry of Education and Research

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

Memory usage

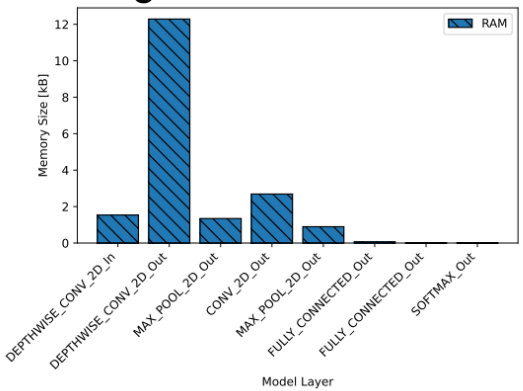
- Reducing memory with accuracy trade-off:
  - Quantization
  - Pruning
  - Network Architecture Search (NAS)

Intermediate Buffers

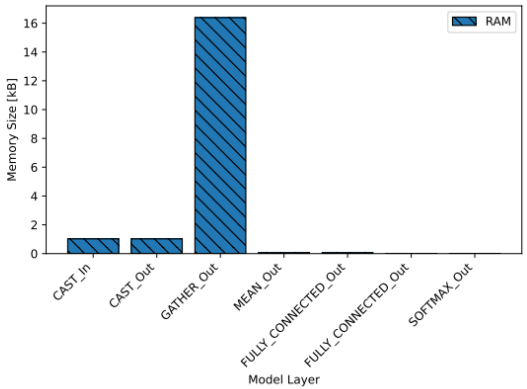
Memory requirements are dominated by few intermediate buffers

→ **Fused Tiling**

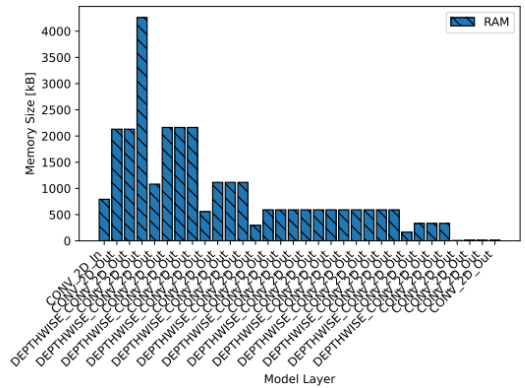
Magic Wand



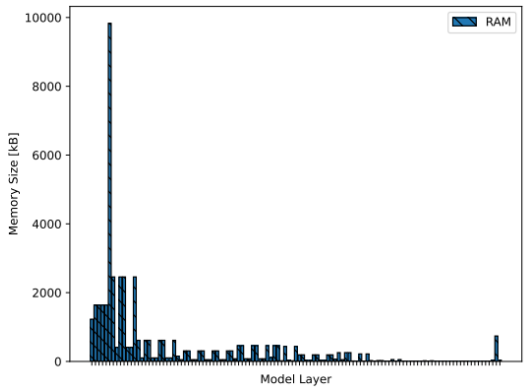
Text Classification



PoseNet



SSDLite

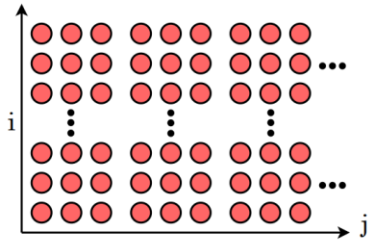


Loop Tiling

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

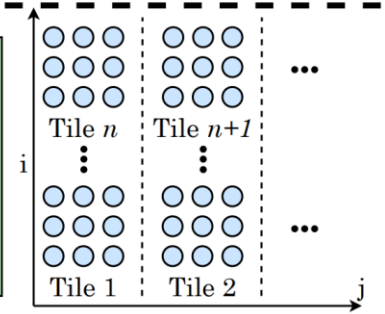
(a) Without loop tiling

```
int i,j,k=2
  for(i=1;i<=30;i+=1)
    for(j=1;j<=30;j+=1)
      a[i][j] += a[i-1][j-1]/k;
```



(b) With loop tiling

```
int i, j, m, n, k=2;
  for(n=1;n<=30;n+=3)
    for(m=1;m<=30;m+=3)
      for(i=n;i<n+3;i+=1)
        for(j=m;j<m+3;j+=1)
          a[i][j] += a[i-1][j-1]/k;
```



Umesh, Sumanth, and Sparsh Mittal. "A survey of techniques for intermittent computing." *Journal of Systems Architecture* 112 (2021): 101859.

Fused Tiling

Tiling:

- Compute large intermediate buffer in multiple tiles

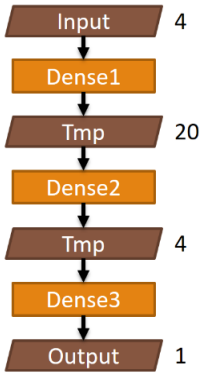
Fusion:

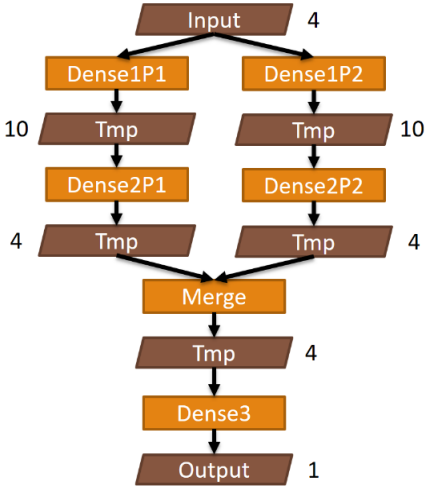
- Operator fusion decouples their lifetime

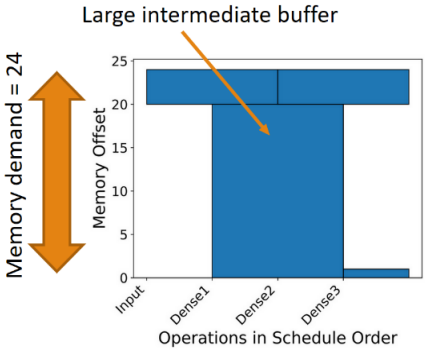
→ Lifetimes of split large buffers do not overlap

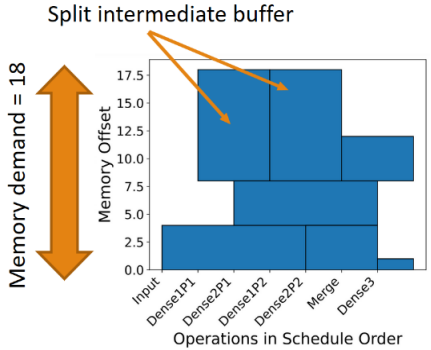
→ Their storage buffers may overlap

→ Memory reduction





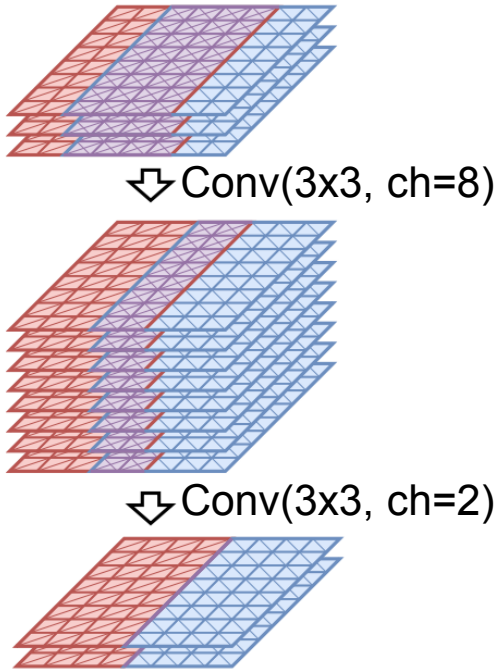




Fused **Feature Map** Tiling (FFMT)

- Operator fusion through spatial locality of convolution
- Introduces overlap from kernel size

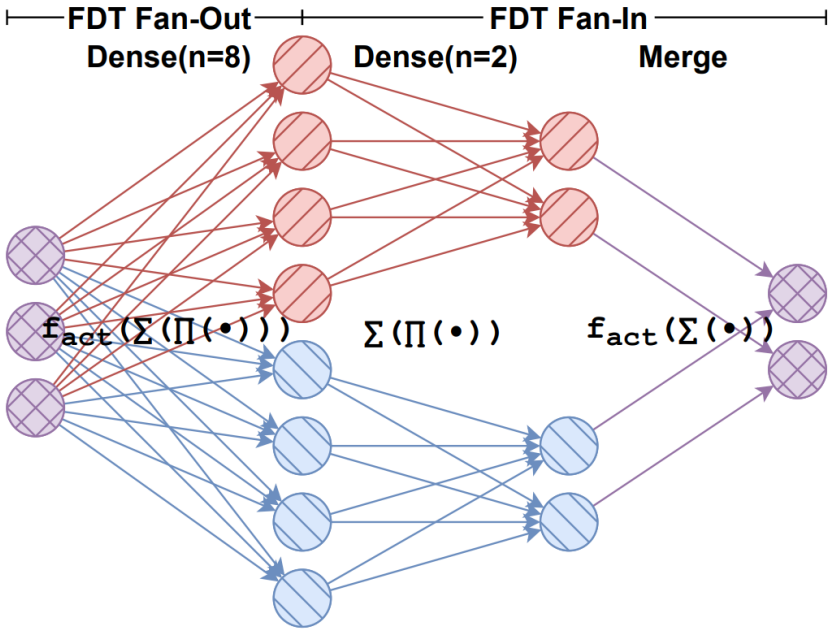
- Does not support operations with large input dependencies
  - Fully connected
  - Convolutions with very large kernel sizes



Fused **Depthwise** Tiling (FDT)

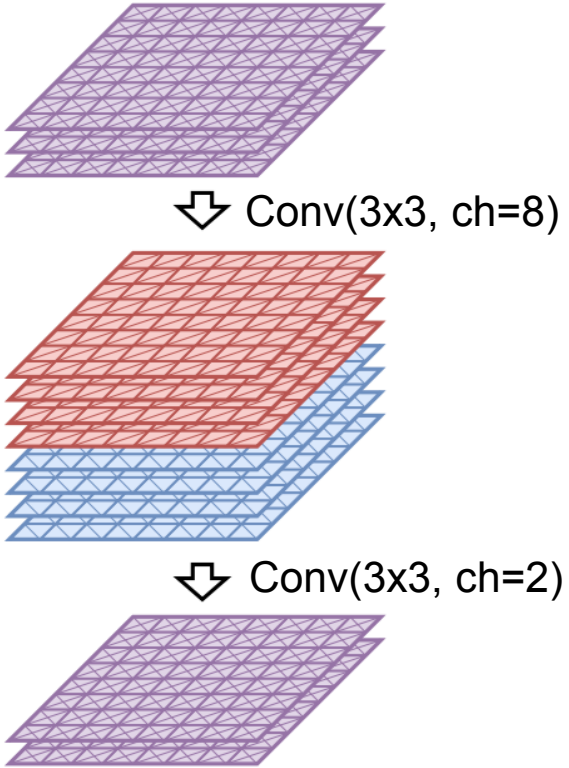
- Allows fusion of two operations with large input dependencies
- Accumulates **partial sum** in second output
- Requires **Merge** operation

- New tiling opportunities
- No significant run time overheads



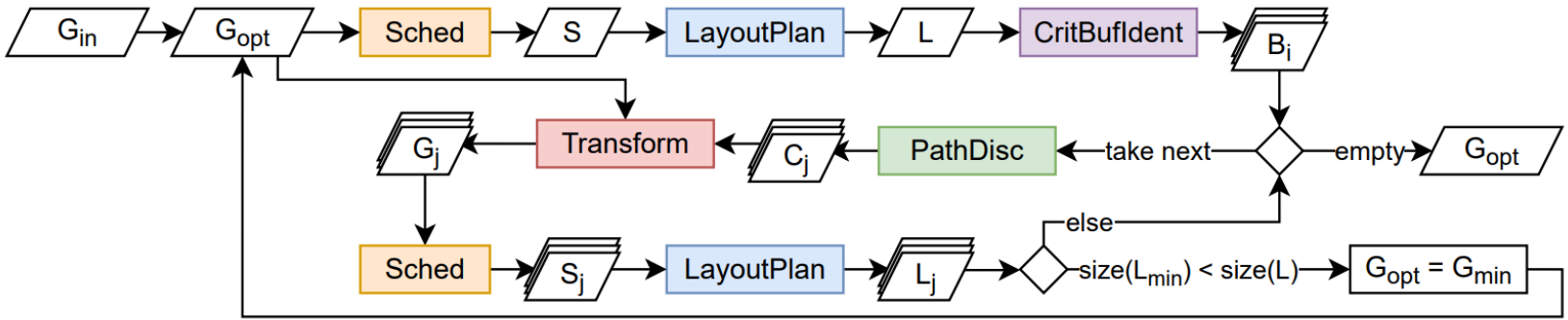
FDT for Convolutions

Split by feature maps instead of neurons

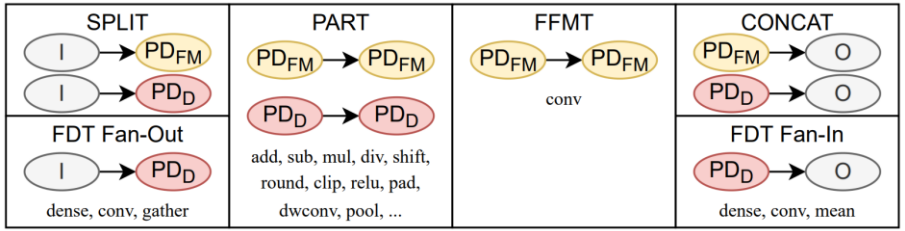


End-to-end Deployment Flow

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

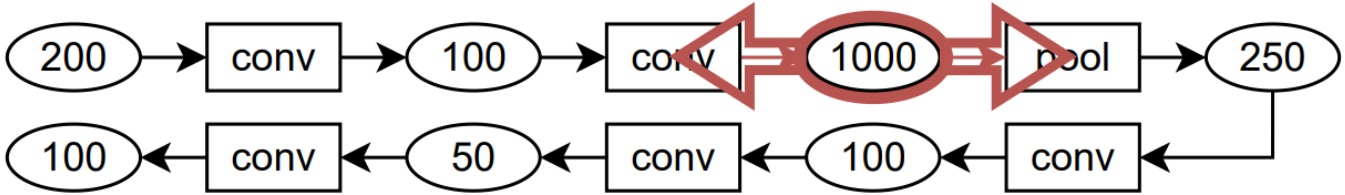


Path Discovery

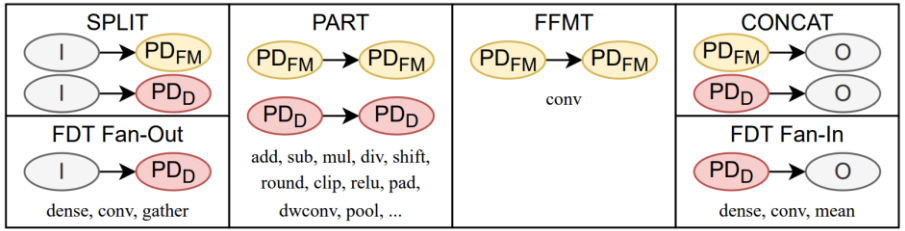


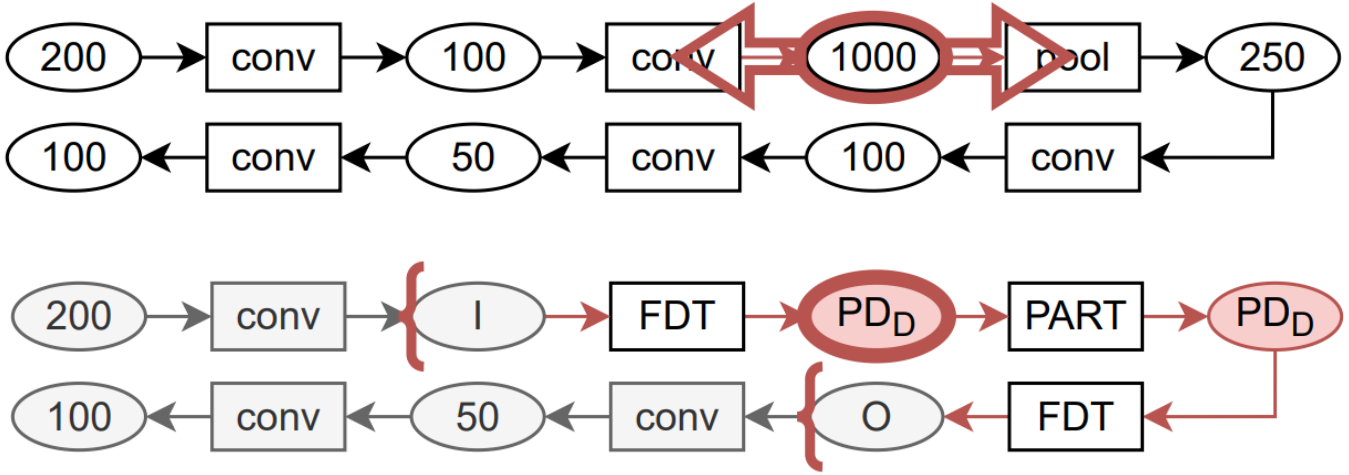
- Finds an optimized sequence of operations (**path**) for fused tiling
- Intermediate buffers and operations are matched according to **blocks**

Untiled

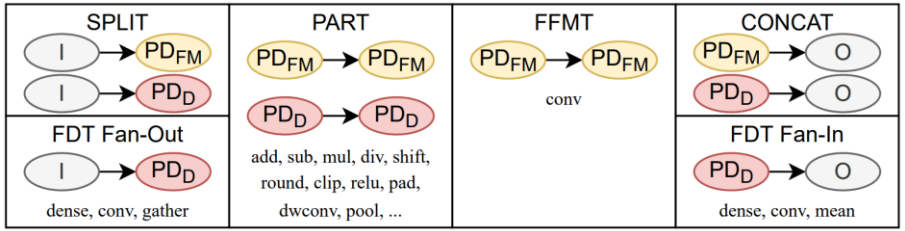


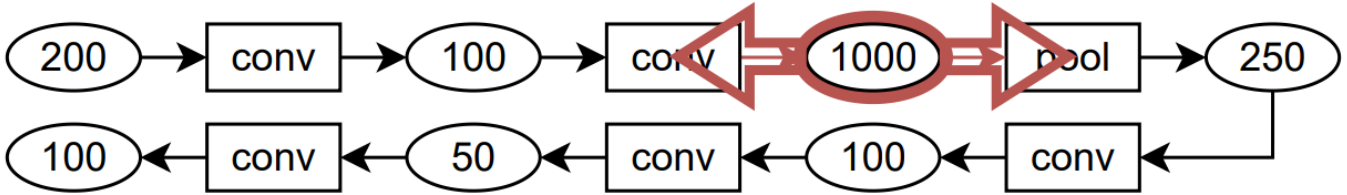
Path Discovery – FDT

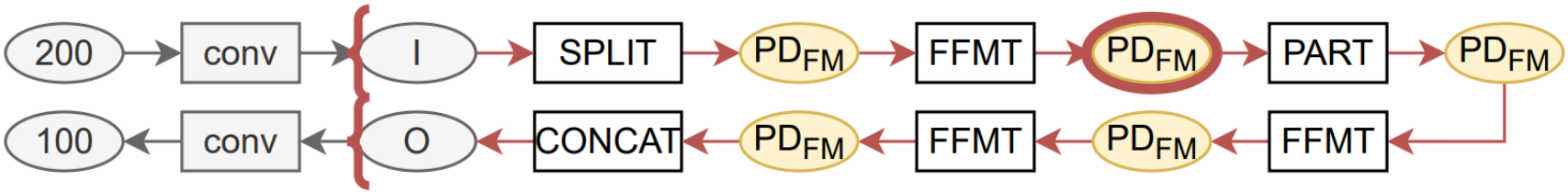




Path Discovery – FFMT







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

	Mem [kB]			[%]		MACs [1 million]			[%]	
Model	Untiled	FFMT	FDT	FFMT Savings	FDT Savings	Untiled	FFMT	FDT	FFMT Overhead	FDT Overhead
KWS	65.6	65.6	53.7	0.0	18.1	2.66	2.66	2.66	0.0	0.0
TXT	18.6	18.6	4.43	0.0	76.2	0.00	0.00	0.00	0.0	0.0
MW	17.6	7.04	11.3	60.9	35.5	0.06	0.06	0.06	0.0	0.0
POS	9.35k	5.11k	8.94k	45.3	4.4	837	1215	837	45.1	0.0
SSD	14.3k	8.66k	12.2k	39.4	14.6	313	314	313	0.2	0.0
CIF	179	76.7	170	57.1	5.0	5.52	6.02	5.52	9.0	0.0
RAD	36.2	26.7	29.4	26.3	18.8	0.09	0.09	0.09	0.0	0.0
Avg.				32.7	24.7				7.8	0.0

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

www.tinyml.org