Enabling ML on the Edge using SRAM Conserving Efficient Neural Networks Execution Approach

Bharath Sudharsan, Pankesh Patel, John G. Breslin, and Muhammad Intizar Ali

Abstract: *we propose an approach for efficient execution of already deeply compressed, large neural networks (NNs) on tiny IoT devices. After optimizing NNs using state-of-the-art deep model compression methods, when the resultant models are executed by MCUs or small CPUs using the model execution sequence produced by our approach, higher levels of SRAM conservation can be achieved*

NNs on MCUs - Challenges

SRAM overflow: To deploy max deep compressed NNs: Can exceed MCU memory just by few bytes SRAM. Cannot additionally tune as already max optimized

Proposed NN execution approach - makes such max compressed NNs consume less SRAM during execution

IoT device fail: Devices running large NNs fail - cases reported: Due to overheating, fast battery wear, run-time stalling. Prime reason is the exhaustion of device SRAM

Proposed Tensor Memory Mapping (TMM) - accurately compute + visualize tensor memory requirement of each operator in NN computation graph during execution

Efficient NN Execution Approach

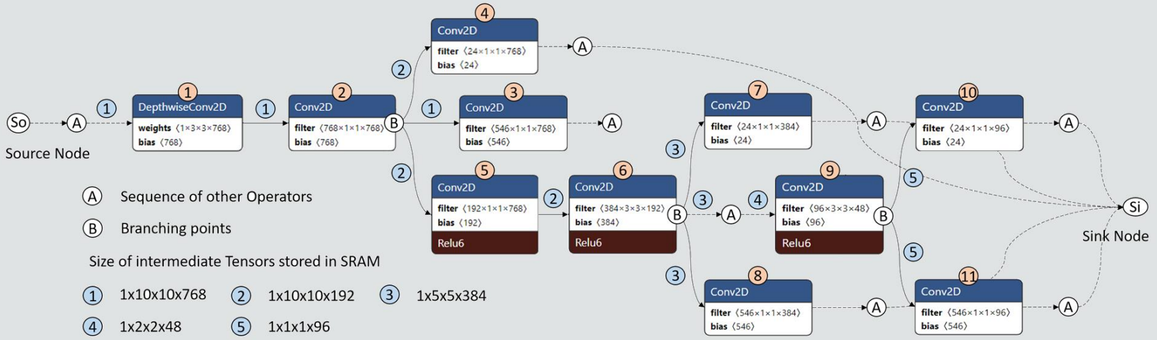


Fig. 1 *– A part of the COCO SSD MobileNet graph with its branched operators*

Tensor Memory Mapping (TMM)

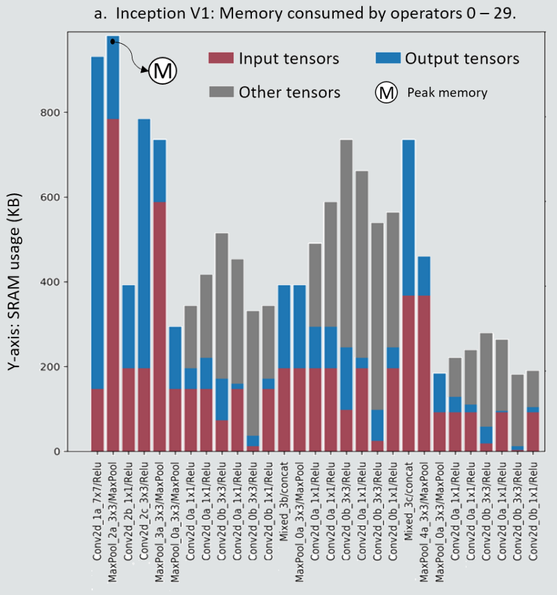
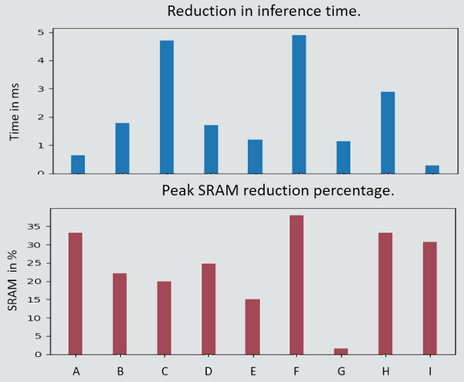


Fig. 2 *– Passing Inception V1 to TMM. Suitable for any pre-trained models like NASNet, Tiny-YOLO, SqueezeNet, etc.*

Cheapest NN graph execution sequence: We proved that in DAGs, execution of available nodes in any topological order will result in a valid execution sequence. Achieves memory conservation goal by intelligently selecting the execution branch that when executed consumes less SRAM

Load fewer tensors and tensors re-usage: Executes many operators by loading a minimum number of tensors. Also achieve SRAM conservation by tensors re-usage



Evaluation Results

A – MobileNet V1, B – Squeezenet, C – Inception V1, D – MnasNet,
E – NASNet mobile, F – DenseNet, G – DeepLabv3, H – PoseNet, I – EAST

DenseNet - max inference time reduction of 4.9 ms. Least of 0.28 ms for EAST. 4 - 84 mJ less energy to perform unit inference

Max peak SRAM reduction of 38.06% for DenseNet. Least of 1.61% for DeepLabv3

Computes total required SRAM. i.e., the space required to store input tensors + output tensors + other tensors, and then exports the detailed report in CSV format

For e.g., we feed Inception V1 that contains 84 graph operators to TMM, it produces above Fig (for brevity, we show only 0 - 29 operators) along with detailed CSV report

Contact: bharath.sudharsan@insight-centre.org