ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

35 Cycle

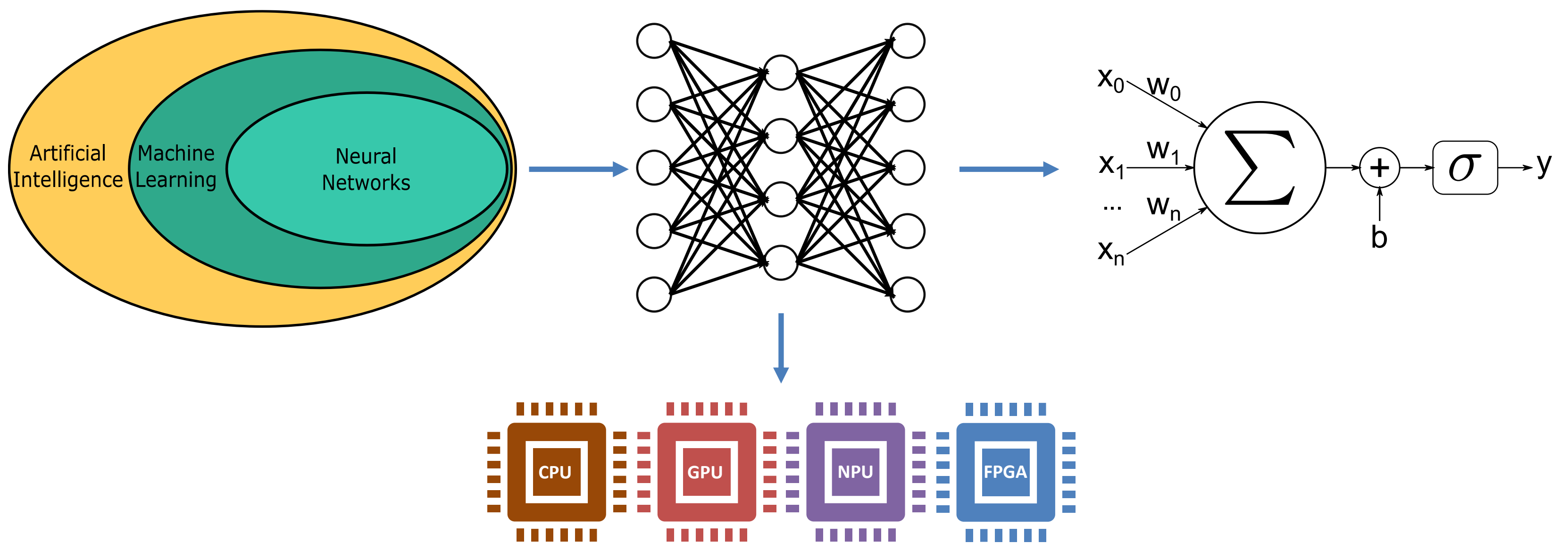
Acceleration of DNNs via Hardware Design and Mapping

Beatrice Bussolino

Supervisor: Prof. Maurizio Martina

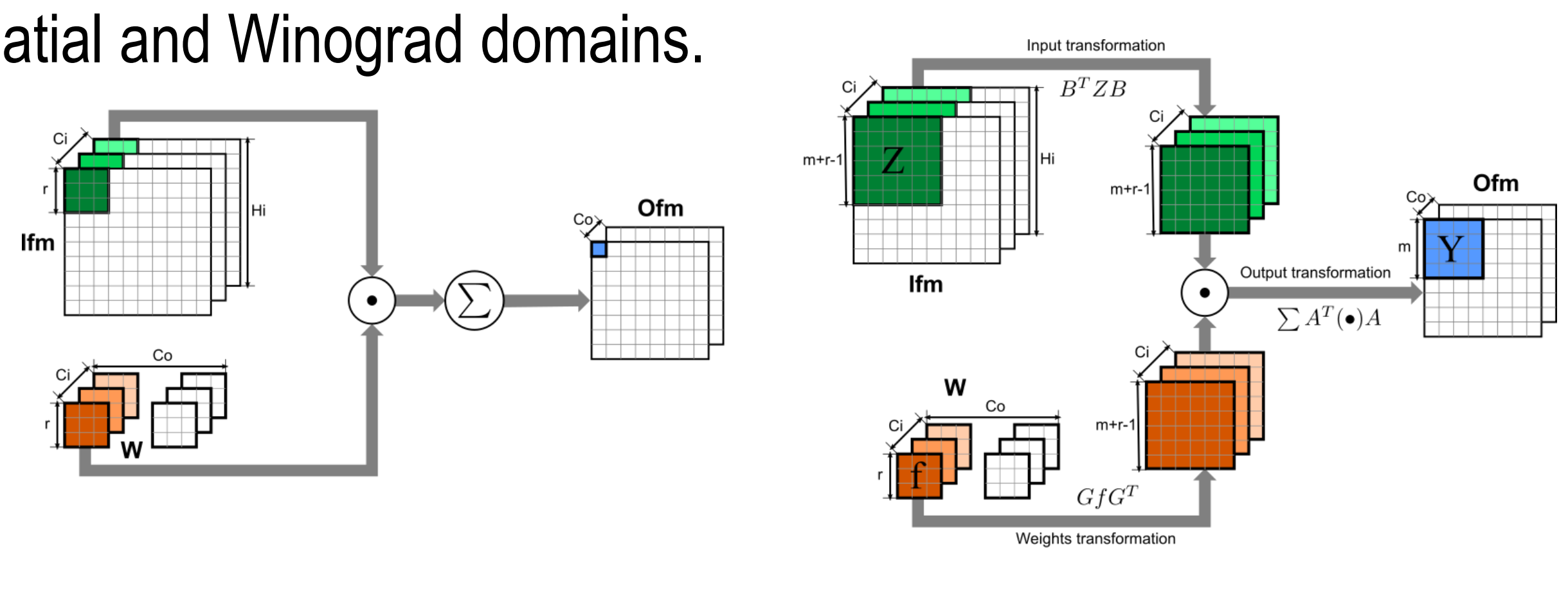
Research context and motivation

- Currently, Machine Learning (ML) is becoming ubiquitous in everyday life. Deep Learning (DL) is already present in many applications ranging from computer vision for medicine to autonomous driving of modern cars as well as other sectors in security, healthcare, and finance. However, to achieve impressive performance, these algorithms employ very deep networks, requiring a significant computational power, both during the training and inference time. A single inference of a Deep Neural Network (DNN) may require billions of multiply-and-accumulated operations, making the DNNs extremely compute- and energy-hungry. In a scenario where several sophisticated algorithms need to be executed with limited energy and low latency, optimized hardware architectures and algorithm mapping strategies are crucial.



Addressed research questions/problems

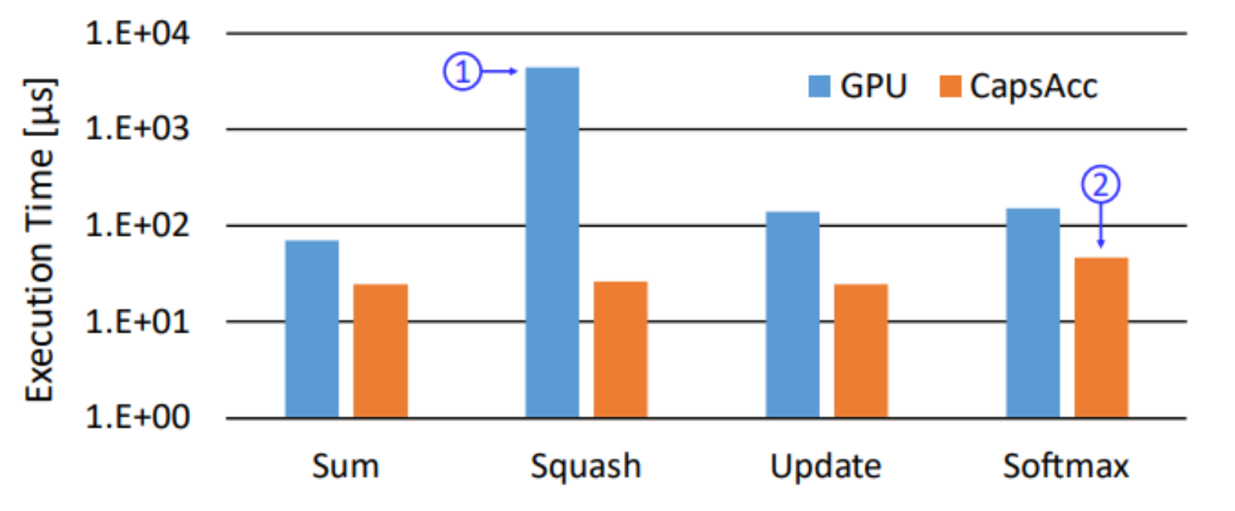
- The **Winograd convolution algorithm** computes convolutions with fewer multiply–accumulate operations (MACs) compared to the standard algorithm, reducing the operation count by a factor of 2.25× for 3×3 convolutions when using the version with 2×2-sized tiles F2. Even though the gain is significant, the Winograd algorithm with larger tile sizes, i.e., F4, offers even more potential in improving throughput and energy efficiency, as it reduces the required MACs by 4×. Unfortunately, the Winograd algorithm with larger tile sizes introduces numerical issues that prevent its use on integer domain-specific accelerators (DSAs) and higher computational overhead to transform input and output data between spatial and Winograd domains.



a) Direct Convolution

b) Winograd F(mxm, rxr) Convolution

- While for generic matrix multiplications (that are used on convolution operations) a common approach is to use approximate adders and multipliers, there are other complex operations (i.e., squash and softmax, typical of **Capsule Networks**) that need more specialized designs to be computed in approximate form. The compute-intensity these operations motivates our research on the design of approximate squash and softmax units, focusing on the tradeoffs for area, power, and delay, without reducing the inference accuracy of the overall network.



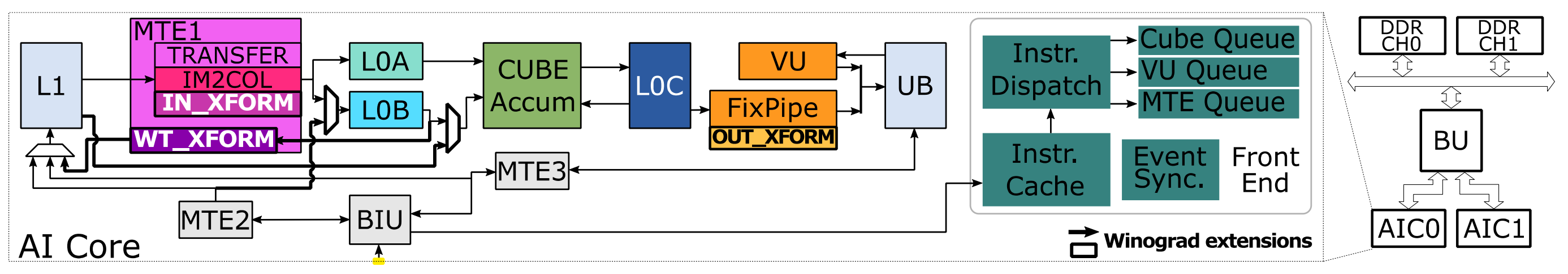
Submitted and published works

Year 2021/2022

- Andri, R., Bussolino, B., Cipolletta, A., Cavigelli, L., Wang, Z., "Going Further With Winograd Convolutions: Tap-Wise Quantization for Efficient Inference on 4x4 Tiles", 55th IEEE/ACM International Symposium on Microarchitecture (MICRO), Chicago, 2022
- Aiello, G., Bussolino, B., Valpreda, E., Ruo Roch, M., Masera, G., Martina, M., Marsi, S., "NLCMAP: A Framework for the Efficient Mapping of Non-Linear Convolutional Neural Networks on FPGA Accelerators", International Conference on Image Processing (ICIP), Bordeaux, 2022
- Marchisio, A., Bussolino, B., Salvati, E., Martina, M., Masera, G., Shafique, M., "Enabling Capsule Networks at the Edge through Approximate Softmax and Squash Operations", ACM/IEEE International Symposium on Low Power Electronics and Design (ISPLED), Boston, 2022

Novel contributions

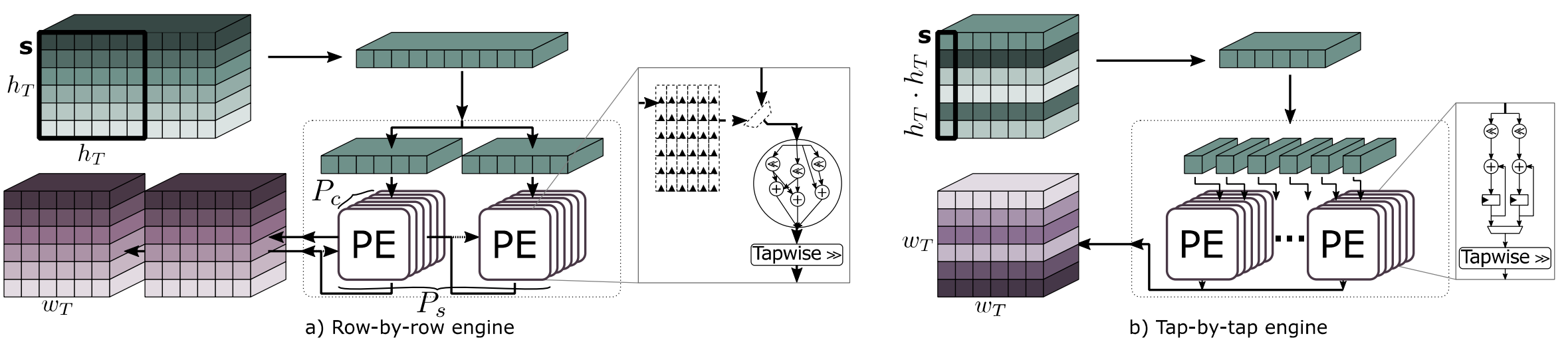
- (**Winograd**) A novel tap-wise quantization algorithm to overcome the numerical issue of Winograd F4 and an architectural and micro-architectural design space exploration for efficient hardware implementation.



- (**Capsule Networks**) Specialized approximate softmax units using domain transformations; approximate squash units with piecewise approximations; integration in the open-source Q-CapsNets framework.

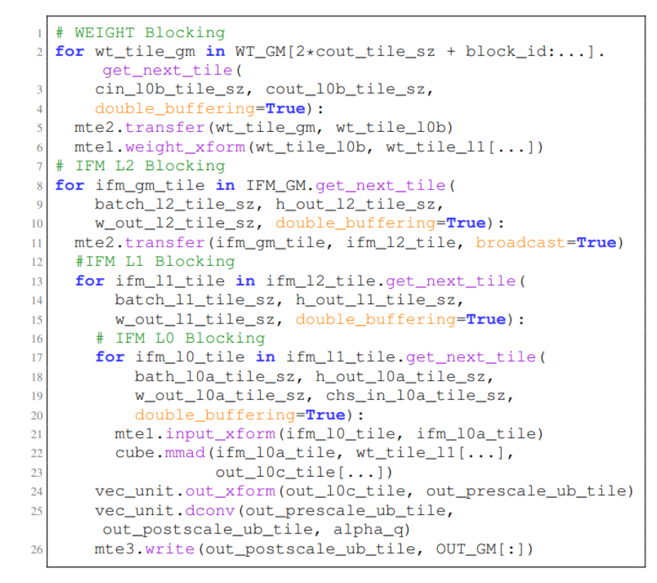
Adopted methodologies

- (**Winograd**) To enable int8 inference, we develop a *tap-wise* quantization scheme, with a different scaling factor for different elementes of the Winograd tiles, with a hardware-friendly training method that constrains the scaling factors to be powers-of-two.
- We do a design space exploration of custom hardwired modules that implement the Winograd transformation operations in an area- and power-efficient way.
- We integrate the Winograd transformation engines in an industrial grade, programmable AI accelerator and how to tune the microarchitecture of such blocks to match the throughput of data movement, Winograd transformation, and compute operations, maximizing the overall compute efficiency



a) Row-by-row engine

b) Tap-by-tap engine



```
# WEIGHT Blocking
for wt_tile_gm in WT_GM[2*cout_tile_sz + block_id:...].
    get_next_tile(
    cin_l0b_tile_sz, cout_l0b_tile_sz,
    double_buffering=True):
  mte2.transfer(wt_tile_gm, wt_tile_l0b)
  mte1.weight_xform(wt_tile_l0b, wt_tile_l1[...])
# IFM L2 Blocking
for ifm_gm_tile in IFM_GM.get_next_tile(
    batch_l2_tile_sz, h_out_l2_tile_sz,
    w_out_l2_tile_sz, double_buffering=True):
  mte2.transfer(ifm_gm_tile, ifm_l2_tile, broadcast=True)
  #IFM L1 Blocking
  for ifm_l1_tile in ifm_l2_tile.get_next_tile(
      batch_l1_tile_sz, h_out_l1_tile_sz,
      w_out_l1_tile_sz, double_buffering=True):
    # IFM L0 Blocking
    for ifm_l0_tile in ifm_l1_tile.get_next_tile(
        bath_l0a_tile_sz, h_out_l0a_tile_sz,
        w_out_l0a_tile_sz, chs_in_l0a_tile_sz,
        double_buffering=True):
      mte1.input_xform(ifm_l0_tile, ifm_l0a_tile)
      cube.mmad(ifm_l0a_tile, wt_tile_l1[...],
          out_l0c_tile[...])
    vec_unit.out_xform(out_l0c_tile, out_prescale_ub_tile)
    vec_unit.dconv(out_prescale_ub_tile,
     out_postscale_ub_tile, alpha_q)
  mte3.write(out_postscale_ub_tile, OUT_GM[:])
```

- (**Capsule Networks**) We implement the approximate softmax and squash architectures in VHDL, synthesize them in a 45nm technology node with the ASIC design flow, and perform gate-level simulations to evaluate the area, power consumption, and critical path delay. By integrating the functional approximations into the Q-CapsNets framework, we evaluate the inference accuracy of state-of-the-art CapsNet models using the proposed approximate units

Results

- (**Winograd**) The tap-wise quantization algorithm makes the quantized Winograd F4 network almost as accurate as the FP32 baseline. The Winograd-enhanced DSA achieves up to 1.85× gain in energy efficiency and up to 1.83× end-toend speed-up for state-of-the-art segmentation and detection networks.
- (**Capsule Networks**) The approximate softmax-b2 design outperforms the related works, having −11% area, −8% power, and −19% critical path delay, and comparable accuracy results. The approximate squash-exp and squash-pow2 have up to −6% power consumption and up to −36% critical path delay compared to the state-of-the-art, while showing similar accuracy as having the exact squash function.

Future work

- Study of architectural changes to generic/industrial AI cores, particularly to data movers, to make data transfer more flexible, with the possibility of performing light on-the-fly transformations, such as im2col, or Winograd.

List of attended classes

- 01DNZRV –Emerging Ultra-low Voltage, Ultra-low Power Analog and Mixed-Signal Integrated Circuits for the IoT (07/09/2022, 4)
- 01DNHRV – System level low power techniques for IoT (15/07/2022, 4)

Politecnico di Torino
1859

PhD program in
Electrical, Electronics and Communications Engineering