Thesis Proposal

Slimmable Neural Networks for Perception aboard Nano-Drones

Politecnico di Torino

Prof. Daniele Jahier Pagliari

Prof. Alessio Burrello

Beatrice Alessandra Motetti

University of California, Irvine

Prof. Marco Levorato

Politecnico di Torino

Nano-drones

Unmanned Aerial Vehicles taxonomy by size [1]

Vehicle class	⊘ : Weight [cm:kg]	Power [W]	Onboard device
standard-size	$\sim 50 : \geq 1$	≥ 100	Desktop
micro-size	$\sim 25 : \sim 0.5$	~ 50	Embedded
nano-size	$\sim 10 : \sim 0.01$	~ 5	MCU
pico-size	$\sim 2 : \leq 0.001$	~ 0.1	ULP

Miniaturization comes at the cost of **limited available power**

~ 60 cm

Image source: https://www.dji.com/it/inspire-2

~ 10 cm

Image source: bitcraze.io/products/crazyflie-2-1/

[1] D. Palossi et al., "Fully onboard ai-powered human-drone pose estimation on ultra-low power autonomous flying nano-uavs," IEEE IOTJ, 2022

Onboard inference

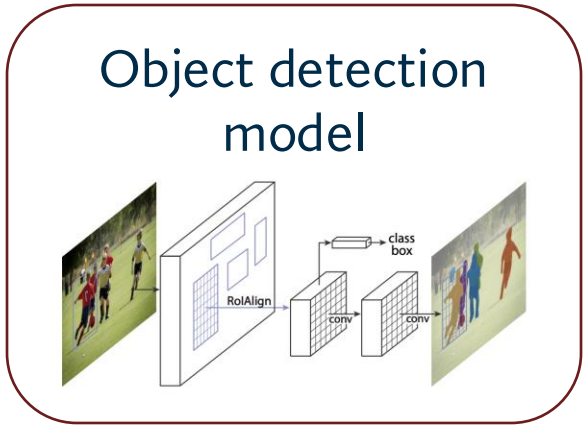
Computation on-board for a more **predictable inference latency**

- It requires to optimize the neural network

Static	Adaptive
Compress model before deployment ○ Neural Architecture Search ○ Pruning ○ Quantization	**Adapt inference complexity** at runtime ○ Dynamic slimmable networks

Task: object detection

Perception



Bounding boxes

Datasets

- COCO2017 (possible starting point) [link]
- Cityscapes [link]
- RADIATE [link]
- nuScenes [link]

Models

- Faster-RCNN
- Mask-RCNN
- …

DNN Deployment on Edge devices

After Model Optimization:

DNN topology, typically written in **Python** with common frameworks

- Pytorch
- Keras

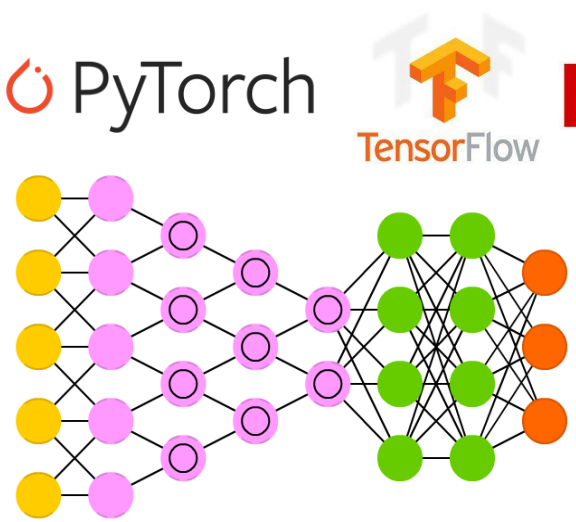
How to fit them in edge devices?

Problems:

- **No Operating System**
- **Low Memory**

	Cloud AI	Tiny AI
Memory (Activation)	16GB	320kB
Storage (Weights)	~TB/PB	1MB

DNN Deployment on Edge devices



Dedicated C Templates

Optimized C kernels (conv, etc..)

Interpreter → Parser → Code Generator

C libraries.RuntimeModule

AOT C runtime C Code | Kernels C Code | Accelerator Kernels C Code | Runtime Library C Code

GCC C Compiler → Binary (.ELF)

1 Phase 1: Creation of a new kernel library

2 Phase 2: Deployment on an edge platform and gather results

Milestones

1. Literature review on the topic
2. Implementation of a slimmable object detector in PyTorch and evaluation on one (or more) benchmarks
3. Deployment of the object detector

Useful material

- Yu et al., *Slimmable Neural Networks*, ICLR 2019 [link]
- Yu et al., *Universally Slimmable Networks and Improved Training Techniques*, ICCV 2019 [link]
- Available code [link]