University of Amsterdam
Informatics Institute

Parallel Computing Systems

HPC for AI

Andy D. Pimentel

Parallel Computing Systems (PCS) group
University of Amsterdam
https://pcs-research.nl

University of Amsterdam
Informatics Institute

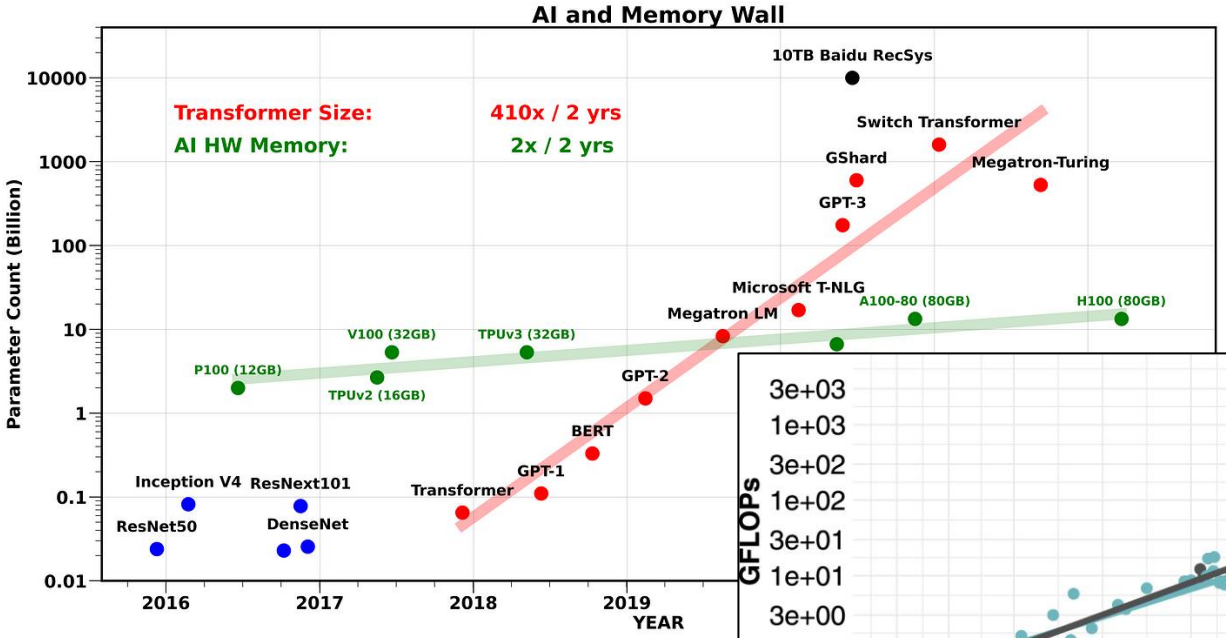
Parallel Computing Systems

HPC for AI?
Well, not always!

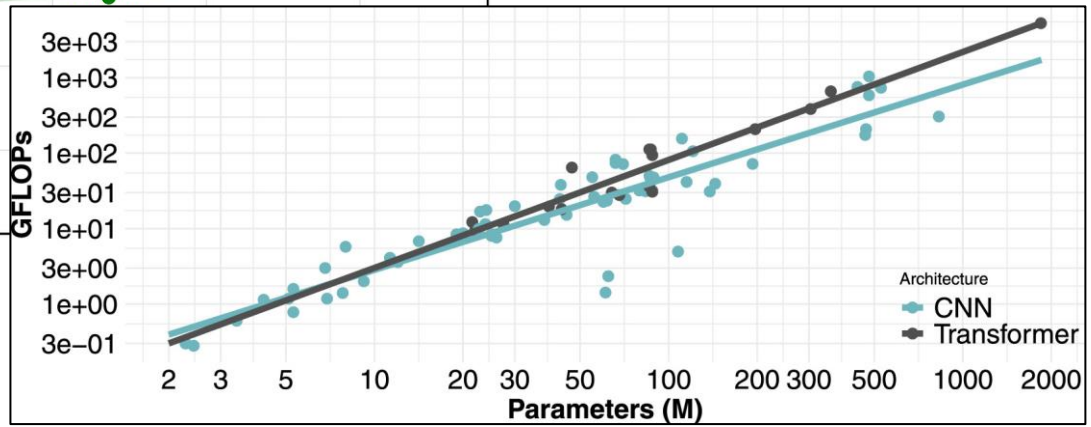
Andy D. Pimentel

Parallel Computing Systems (PCS) group
University of Amsterdam
https://pcs-research.nl

Modern DNNs: big, bigger, humongous



[Gholami et al., "AI and Memory Wall," *IEEE Micro*, 2024]

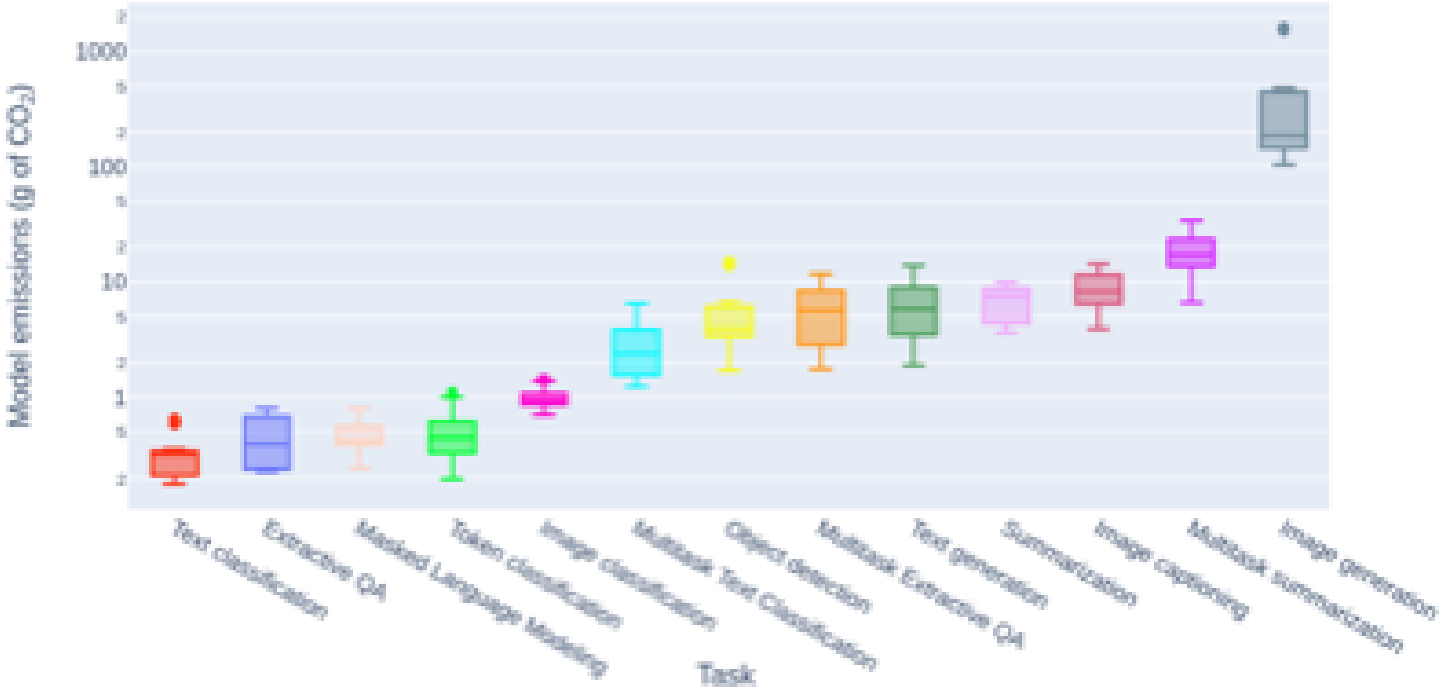


[Desislavov et al., “Trends in AI inference energy consumption”, Sustainable Computing: Informatics and Systems, 2023]

Energy consumption of deep learning

[Yuzhuo Li et al., "The Unseen AI Disruptions for Power Grids: LLM-Induced Transients", 2024]

And Generative AI doesn't help…



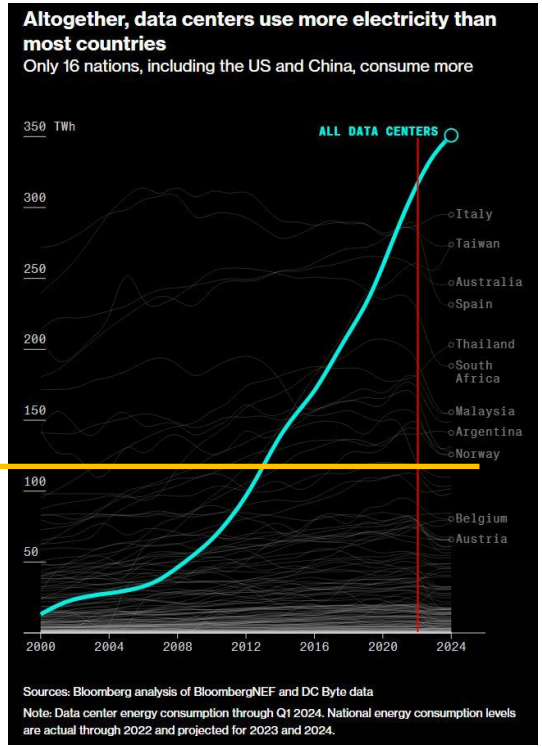
[A.S. Luccioni et al., "Power Hungry Processing: Watts Driving the Cost of AI Deployment?", 2024]

AI Energy usage: inference is dominant!

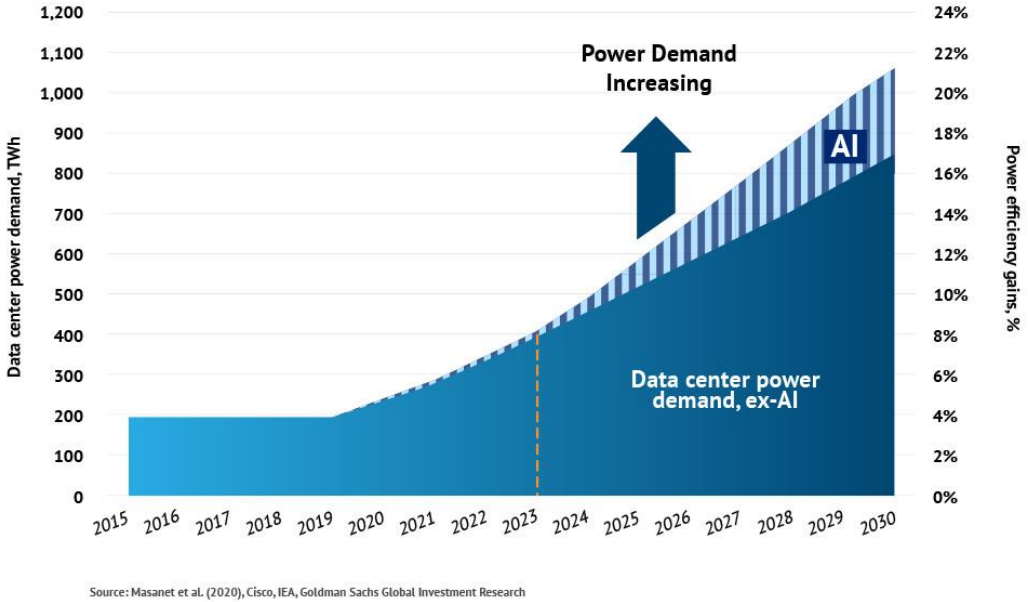
- Let's take ChatGPT as an example and do the math:
  - ✓ The training of GPT-4 consumed approximately 60 GigaWatt-hours
  - ✓ ChatGPT receives more than 1 billion (inference) queries per day
  - ✓ A single ChatGPT query takes roughly 3 watt-hours
- After 20 days of usage, inference has consumed more energy than training

AI is (currently) powered by the Cloud

Energy usage of data centers



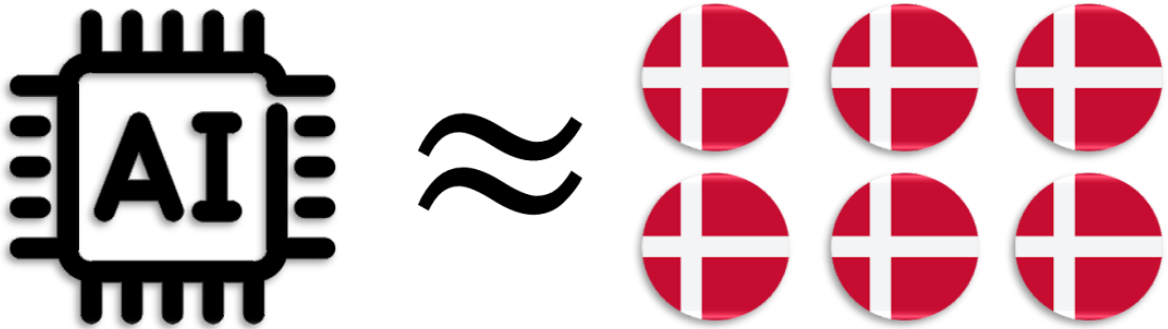
NL



And let's not forget water usage of AI

Global AI's Scope 1 & 2 Water Withdrawal in 2027

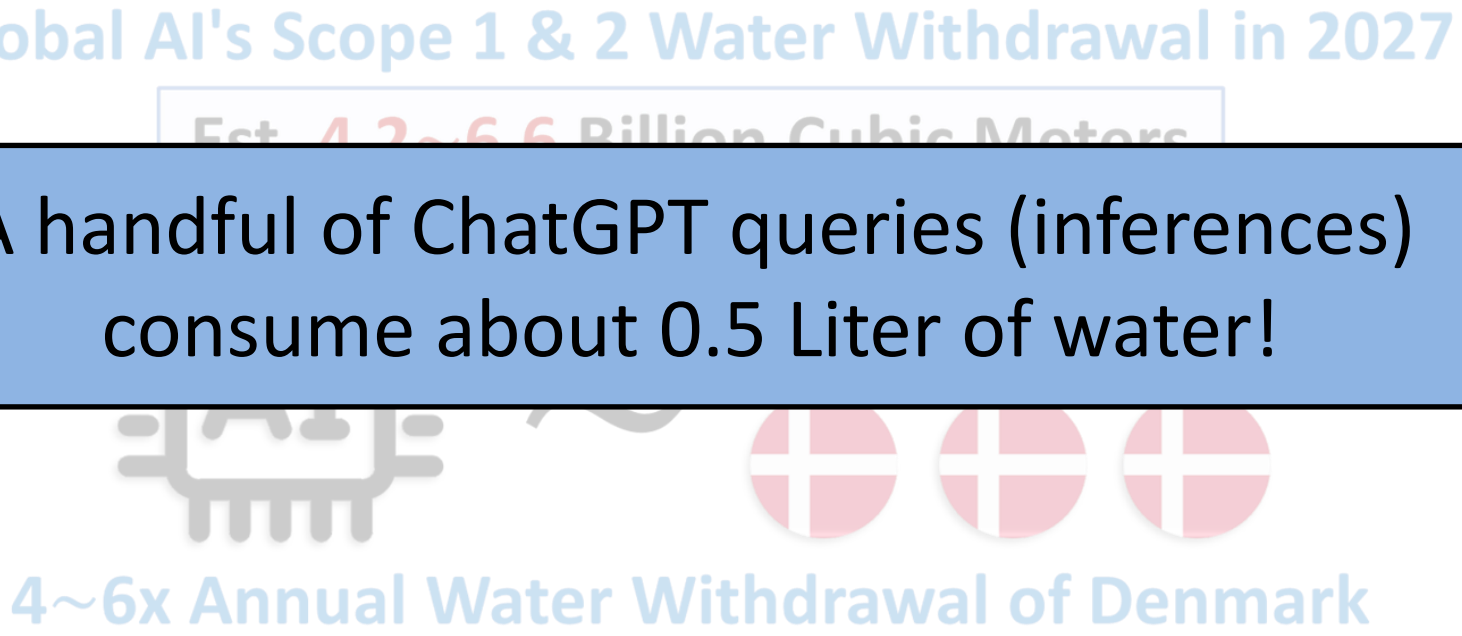
Est. 4.2~6.6 Billion Cubic Meters



4~6x Annual Water Withdrawal of Denmark

[S. Ren, "How much water does AI consume? The public deserves to know", 2023]

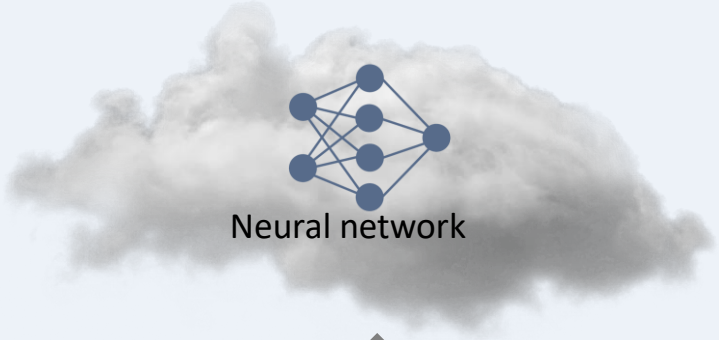
And let's not forget water usage of AI



A handful of ChatGPT queries (inferences) consume about 0.5 Liter of water!

[S. Ren, "How much water does AI consume? The public deserves to know", 2023]

A clear trend towards Edge AI



Edge devices

Edge AI

- Lower latency
- More privacy
- Lower energy consumption

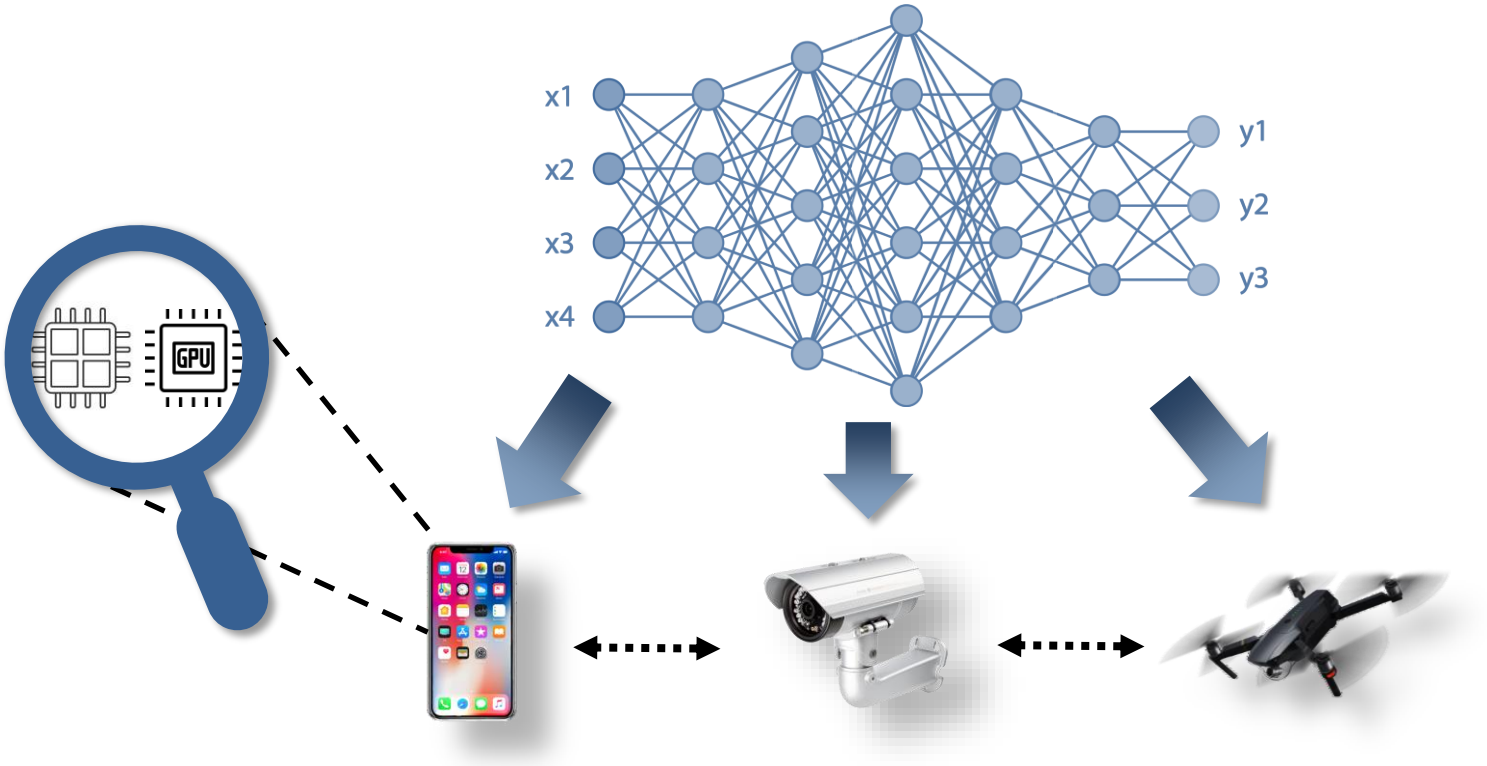
Edge devices

Edge AI is challenging

- Modern DNNs are too big to fit on edge/end devices due to limited power budget and compute/memory resources
- Solutions:
  - ✓ DNN Compression
  - ✓ Hardware-aware Network Architecture Search (NAS)
  - ✓ Distribution of DNNs across the edge-to-cloud continuum
  - ✓ Distribution of DNNs across edge devices

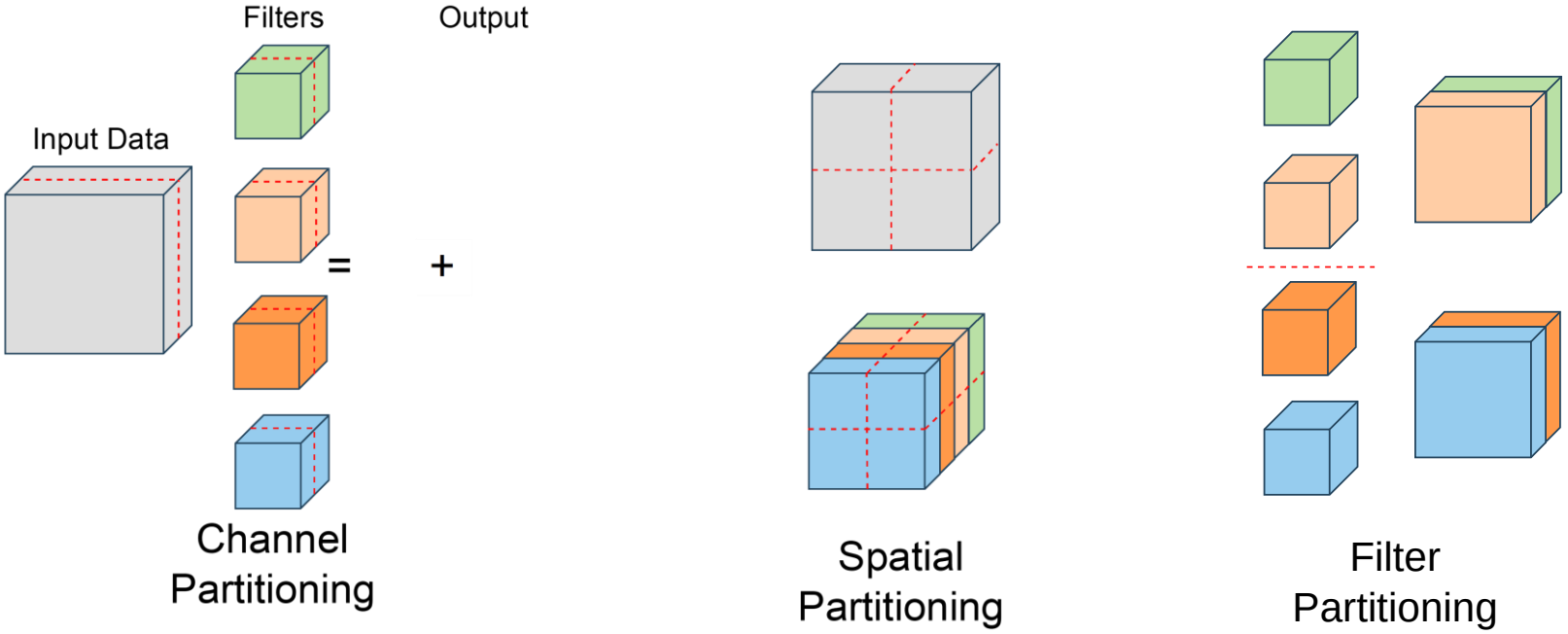
Distribution of DNNs across Edge devices

In collaboration with:
Universiteit Leiden
The Netherlands



Vertical/Pipelined partitioning

Alternative: “horizontal” partitionings



A CNN example

DNN partitioning and distributed execution:

A lot of tedious engineering and a large (parallel) programming effort!

AutoDICE: automated distributed DNN inference

DNN model (.onnx)

HW platform description

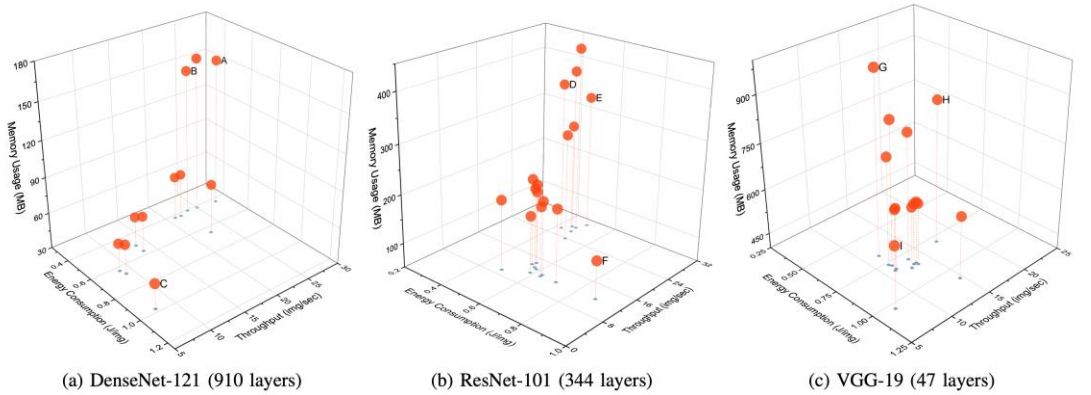
Mapping and splitting description

AutoDICE

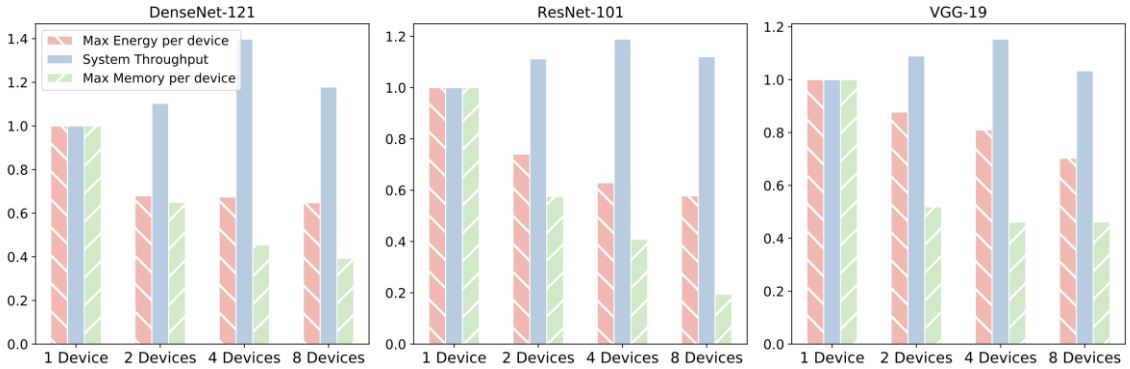
- Automated model splitting
  - ✓ Vertical/horizontal partitioning
- Automated code generation
  - ✓ MPI and OpenMP based
- Automated deployment

Automated mapping and splitting exploration

- Genetic Algorithm based search
  - ✓ Deploying analytical models + AutoDiCE measurements



(a) DenseNet-121 (910 layers) (b) ResNet-101 (344 layers) (c) VGG-19 (47 layers)

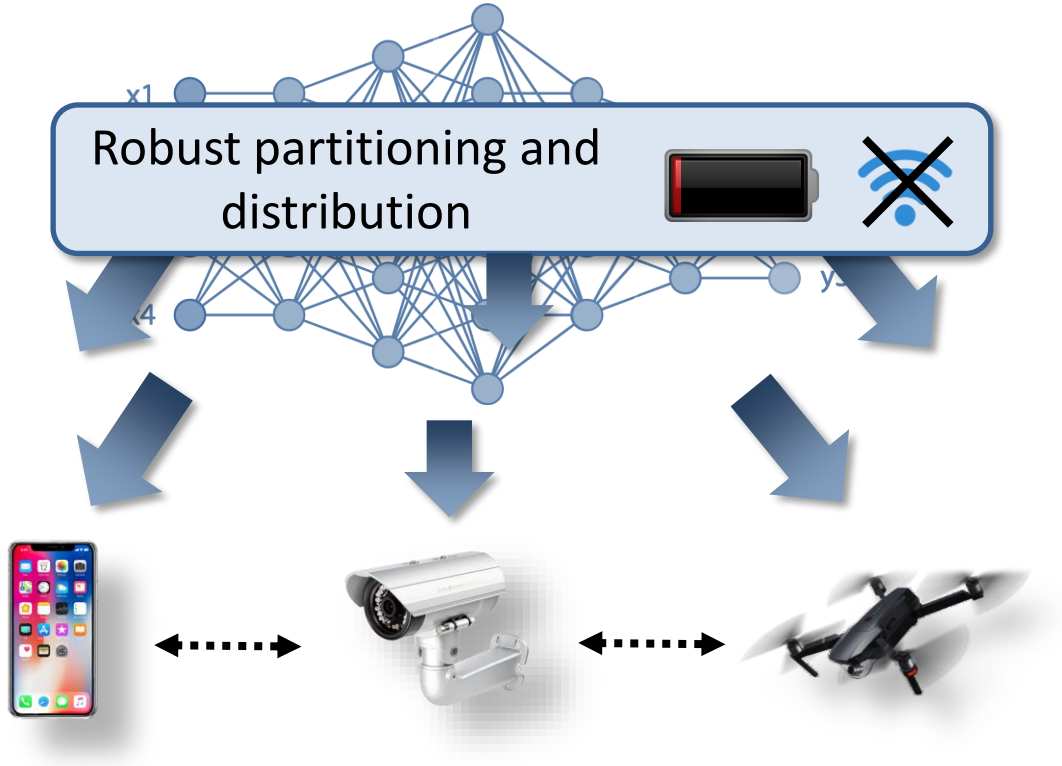


[X. Guo, A.D. Pimentel and T. Stefanov., IEEE Internet of Things Journal, Vol. 10(7), 2023]

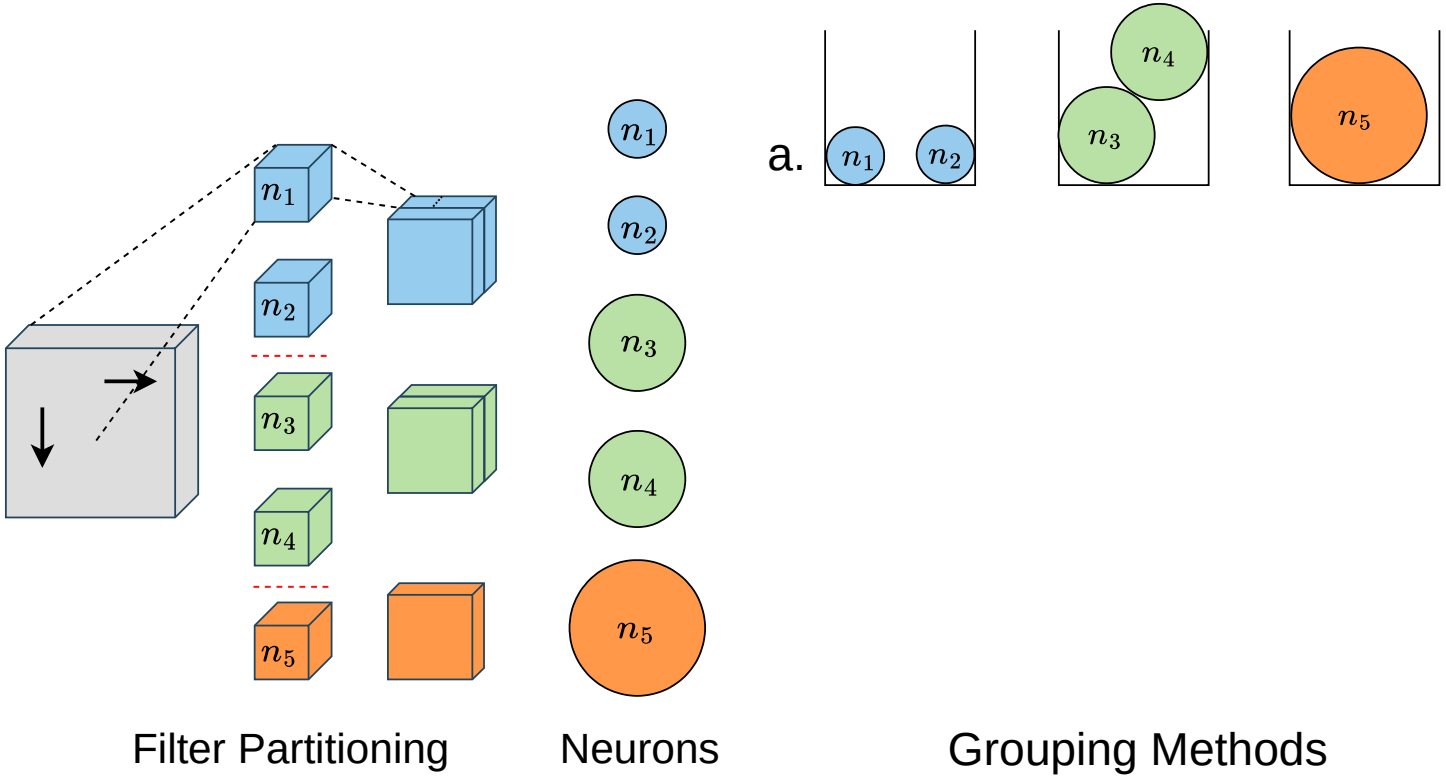
But wait a minute,
Edge devices are not reliable!
….

Robust, distributed DNN inference at the Edge

In collaboration with:
Universiteit Leiden
The Netherlands

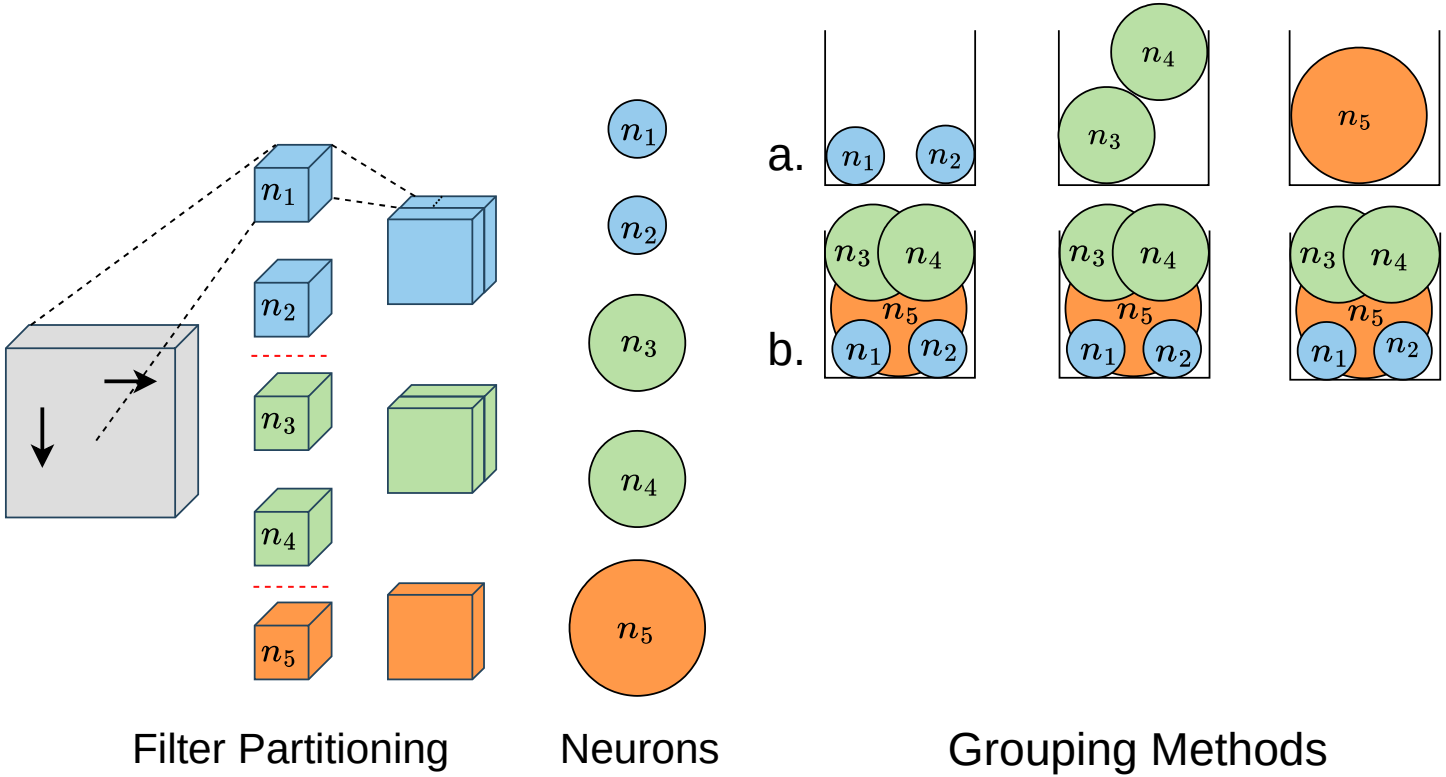


Robust, distributed DNN inference at the Edge



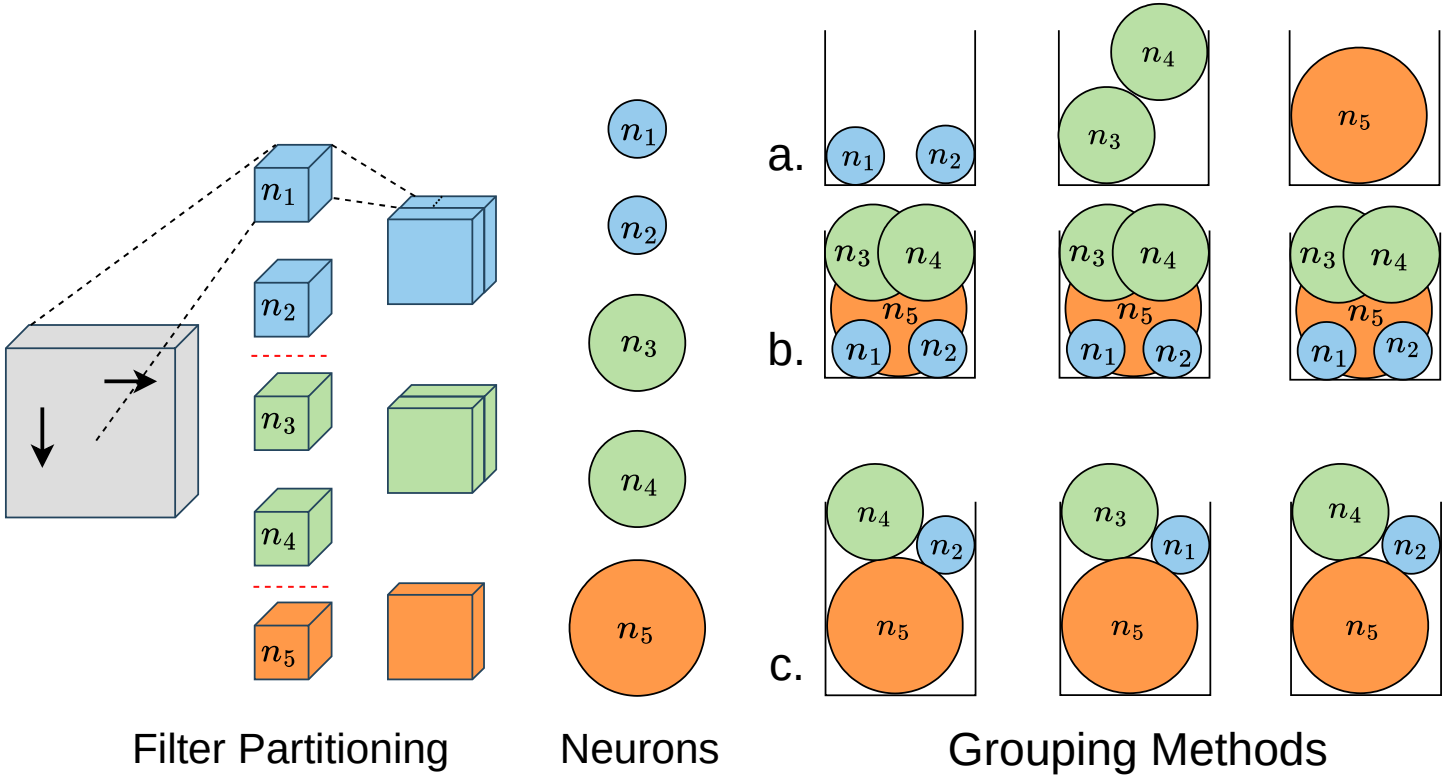
[X. Guo, A.D. Pimentel, and T. Stefanov, *ASP-DAC '24]*

Robust, distributed DNN inference at the Edge



[X. Guo, A.D. Pimentel, and T. Stefanov, *ASP-DAC '24]*

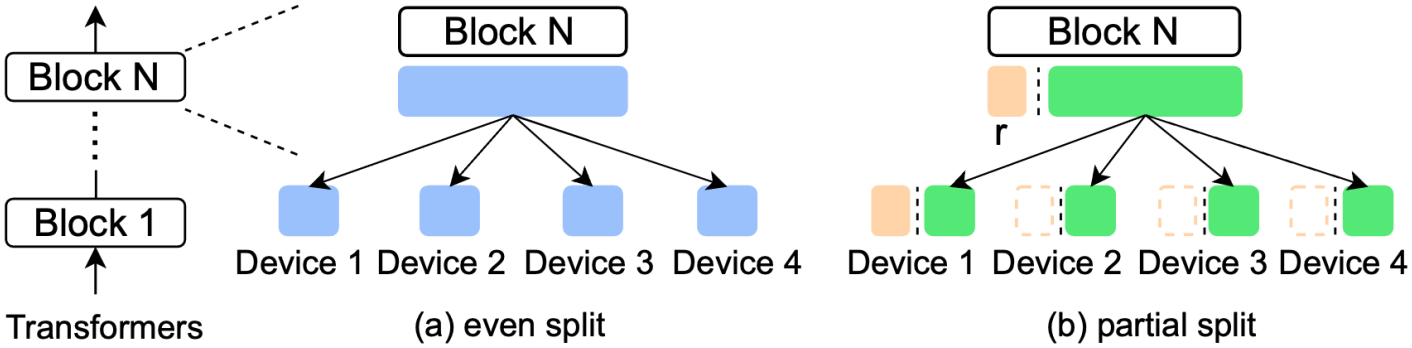
Robust, distributed DNN inference at the Edge



[X. Guo, A.D. Pimentel, and T. Stefanov, *ASP-DAC '24]*

How to split a Transformer robustly?

Partial Split Method



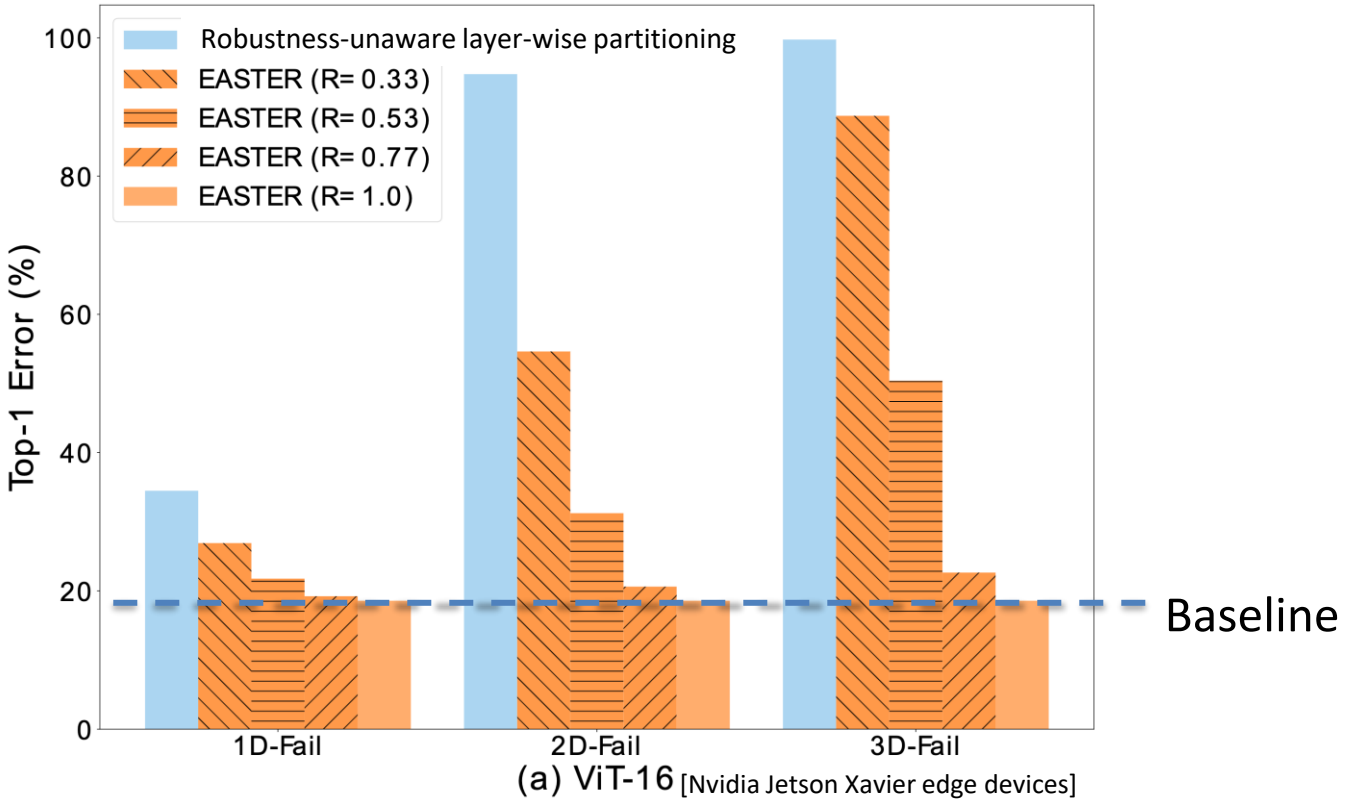
Target: Search $(r_1, r_2, \ldots, r_N)$ for N blocks

Tunable fraction ***r*** of replicated layer weights provides *tradeoff between robustness and memory usage*:

r = 0.0	⟷	**r** = 1.0
Less Robustness		Maximum Robustness
Less Memory		More Memory
Less Computation		More Computation

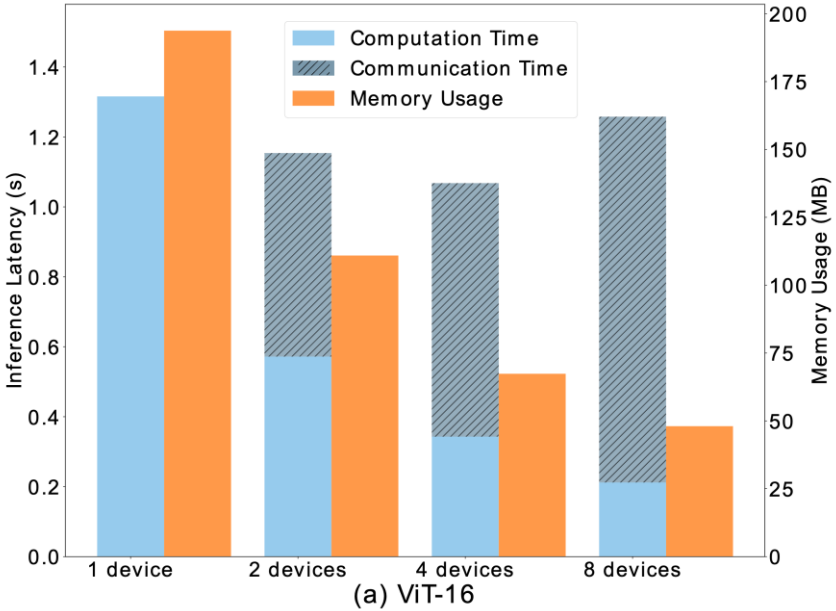
[X. Guo, A.D. Pimentel, and T. Stefanov, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol 43 (nr. 11), 2024]

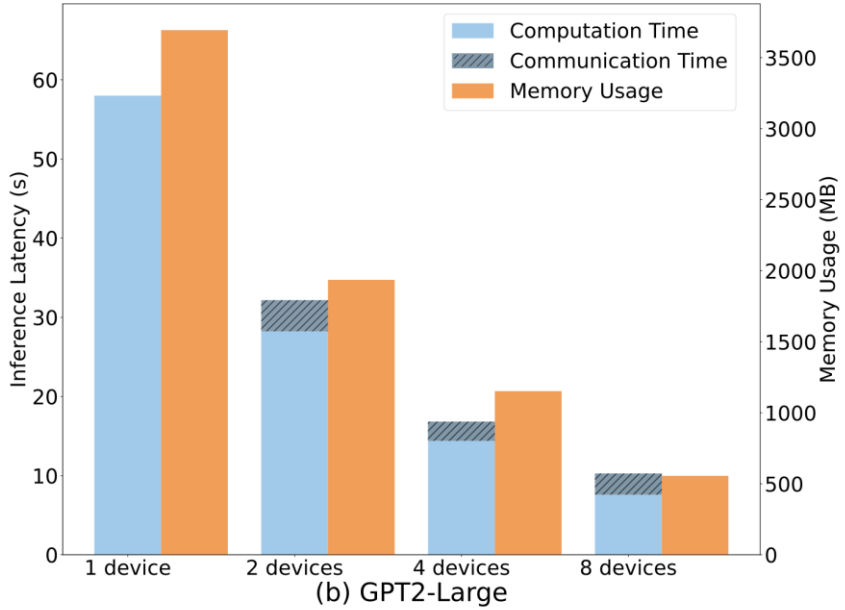
Experimental results: robustness evaluation



[X. Guo, A.D. Pimentel, and T. Stefanov, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol 43 (nr. 11), 2024]

Experimental results: scalability





Distributed inference with **r = 0**

[X. Guo, A.D. Pimentel, and T. Stefanov, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol 43 (nr. 11), 2024]

Thank you!