Part of the 2026 IEEE World Congress on SERVICES

Profiling Neural Network Partitioning Strategies for Inference across the Computing Continuum

Nikolaos Papadakis, Alexandros Angourakis, Kostas Magoutis, Georgios Bouloukakis

2026 IEEE International Conference on Edge Computing and Communications (IEEE EDGE 2026)

13-18 July 2026, Sydney, Australia

AI Inference Is Moving into the Computing Continuum

Why is inference moving towards the Edge?

Latency
Real-time decisions cannot wait

Privacy
Keep sensitive data on-device

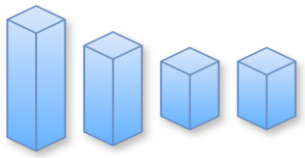
Bandwidth & Costs
Streaming to the cloud is expensive and often infeasible

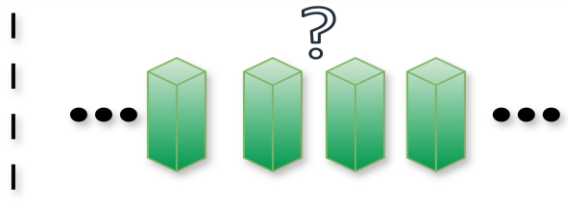
DEVICE (Constrained) ⟷ **EDGE (Nearby)** ⟷ **CLOUD (ELASTIC)**

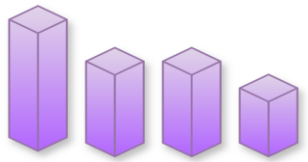
Low Latency
High Bandwidth
(local)

Higher Latency
Lower Bandwidth
(wide-area)

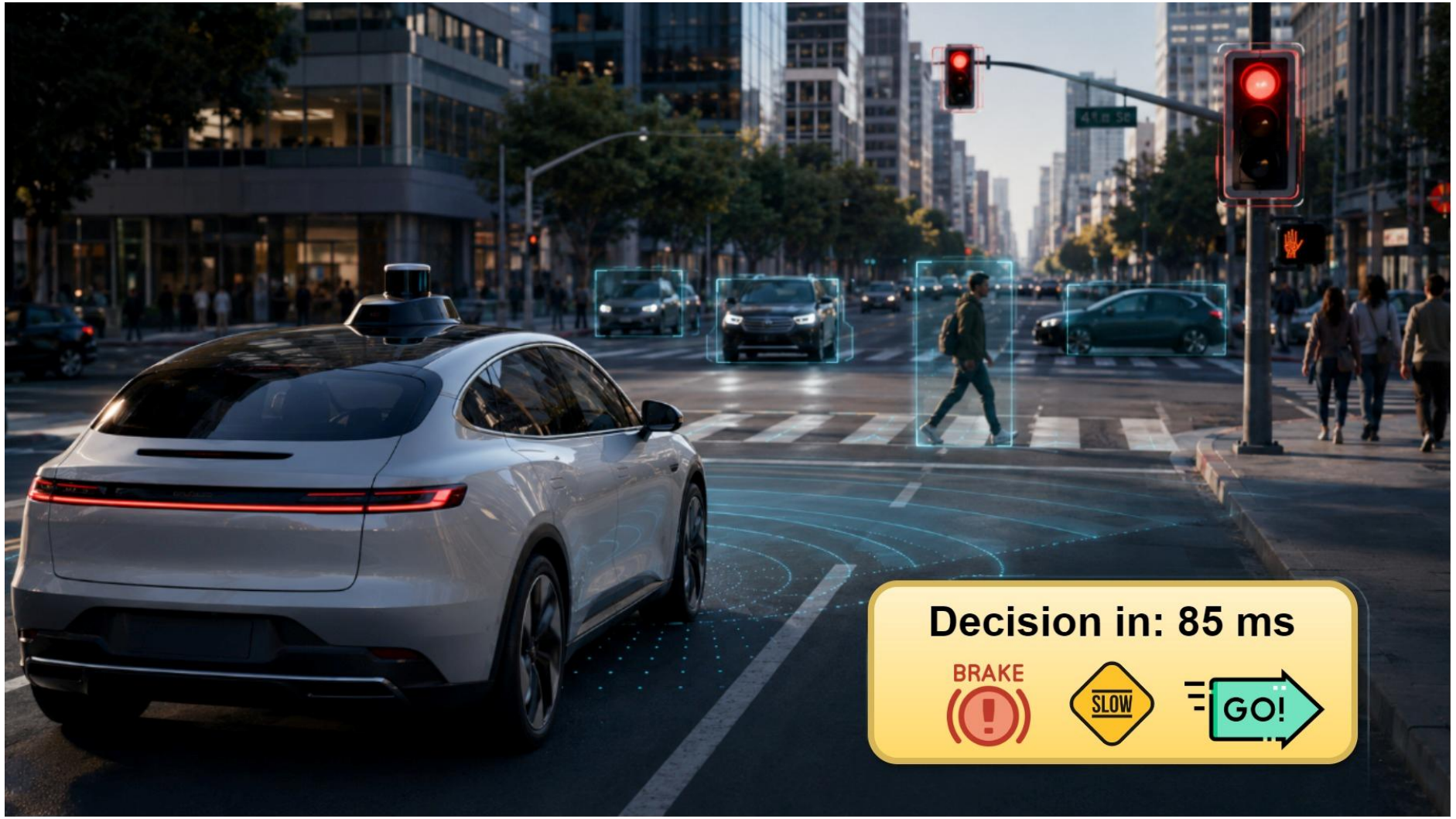
How should we partition the model across the continuum?



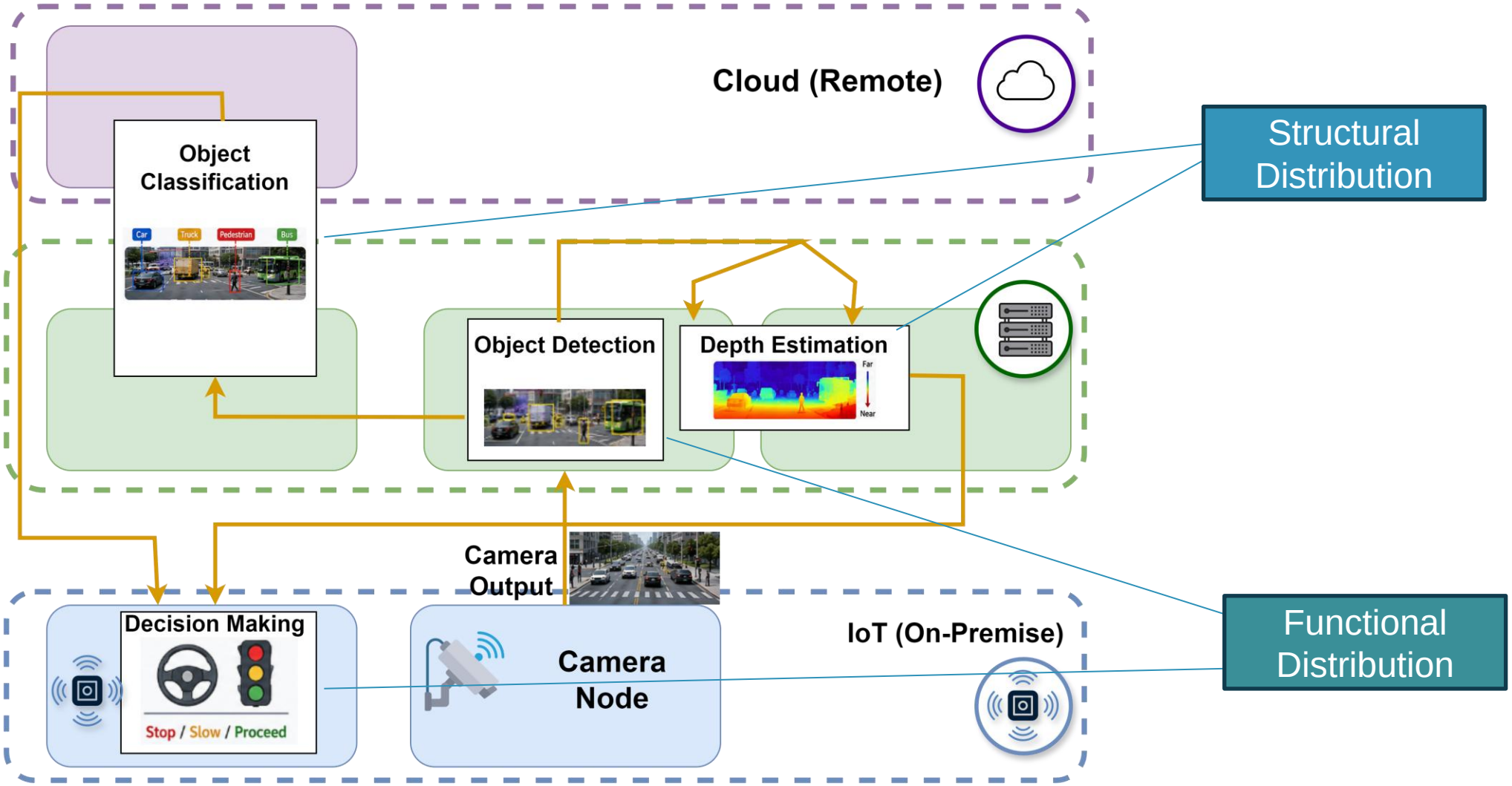




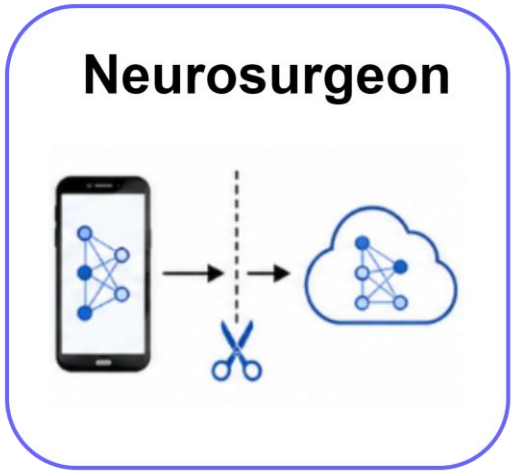
Inference in an Autonomous Vehicle

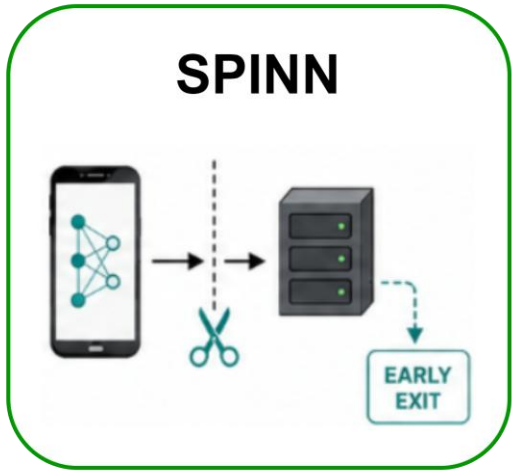


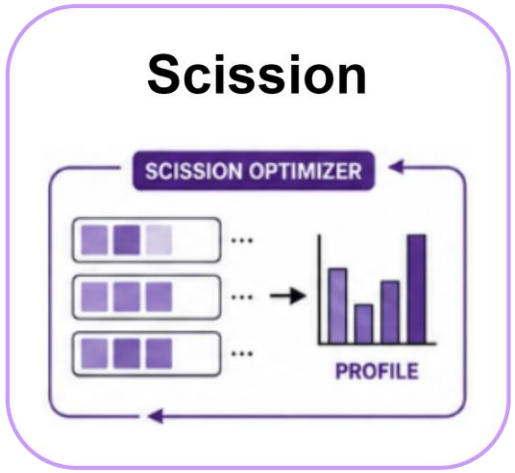
Inference in an Autonomous Vehicle

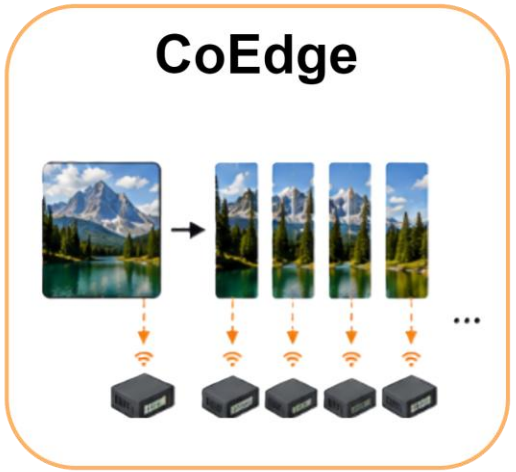


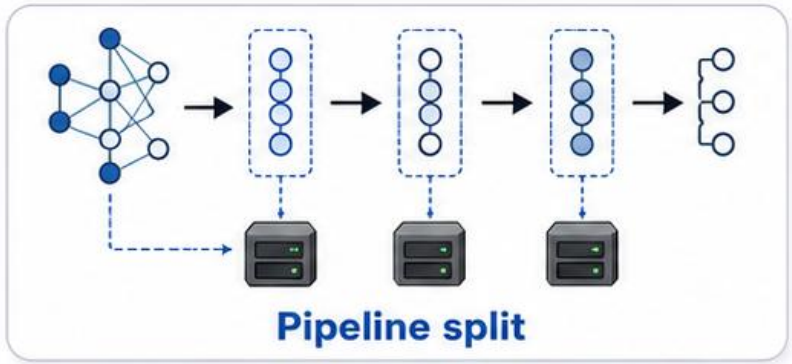
Related Work

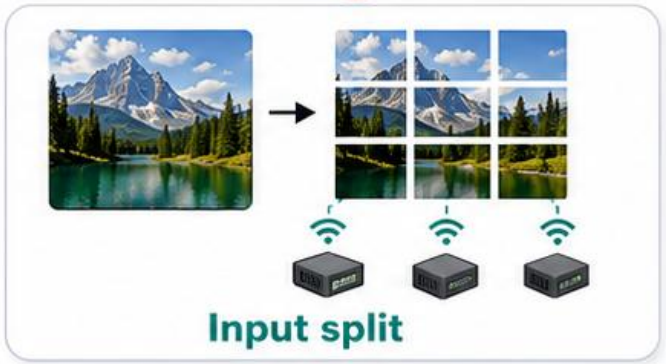


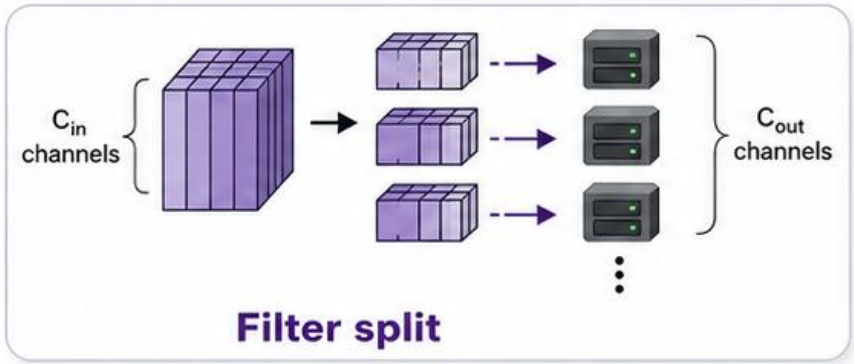






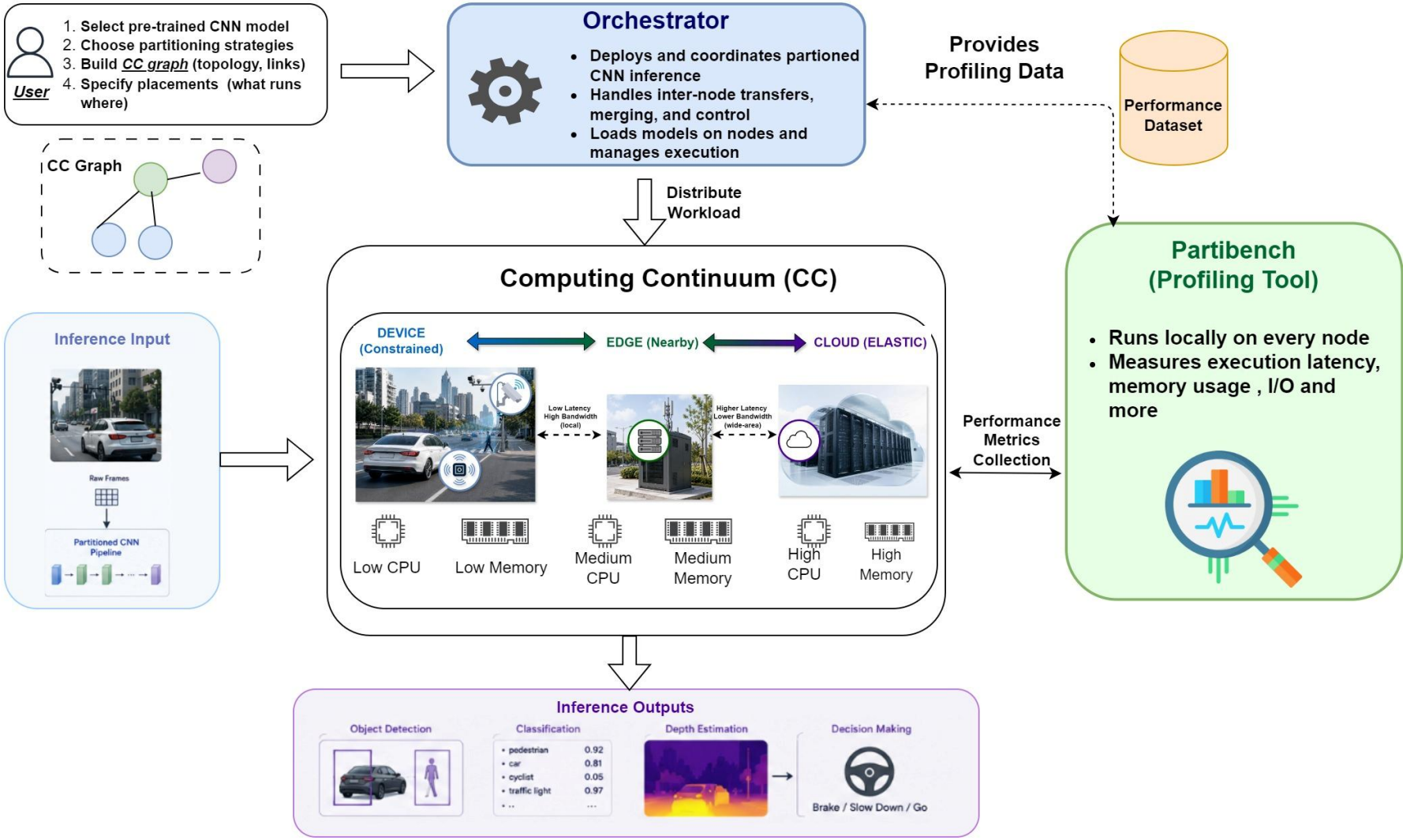




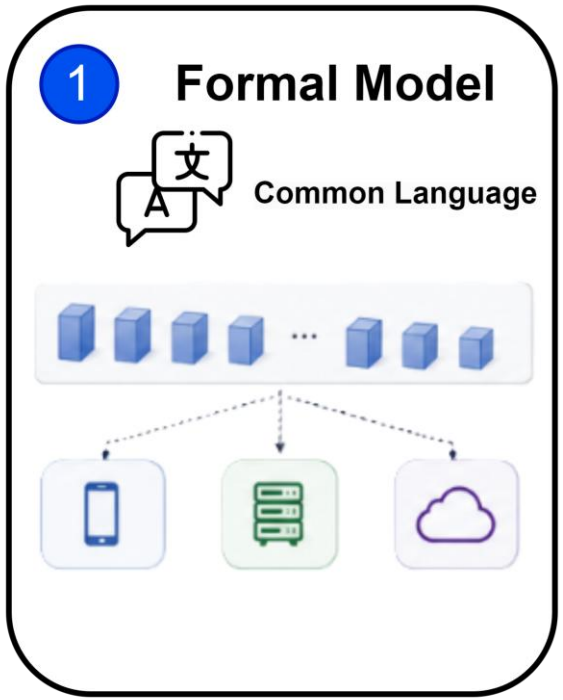


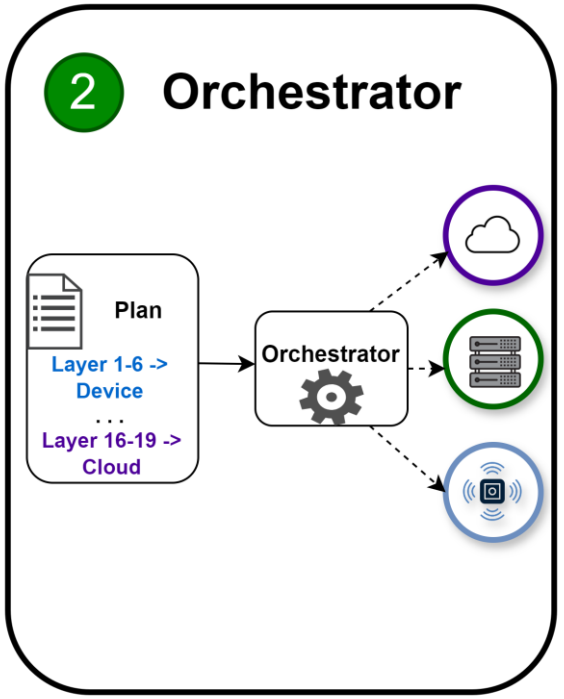
You cannot ask: Which splitting strategy is best for MY setup?

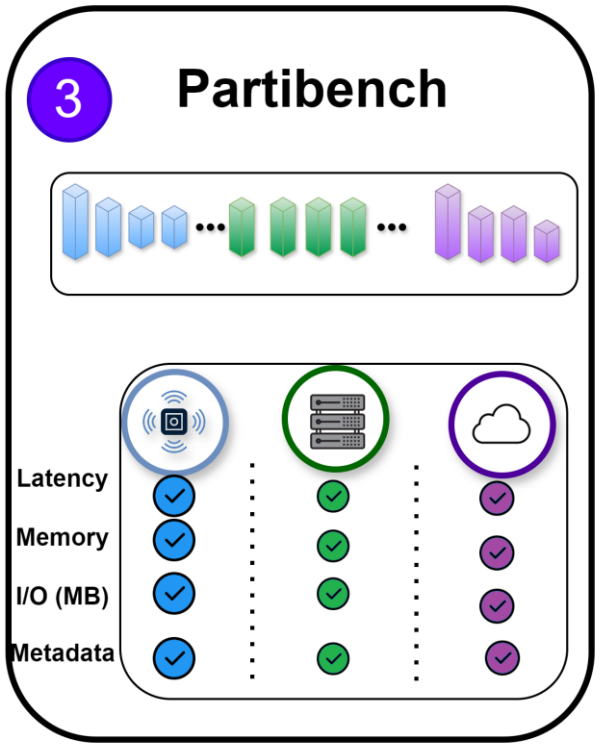
Our approach: Overview



Our Approach: A Model, a Runtime, and a Benchmarking platform





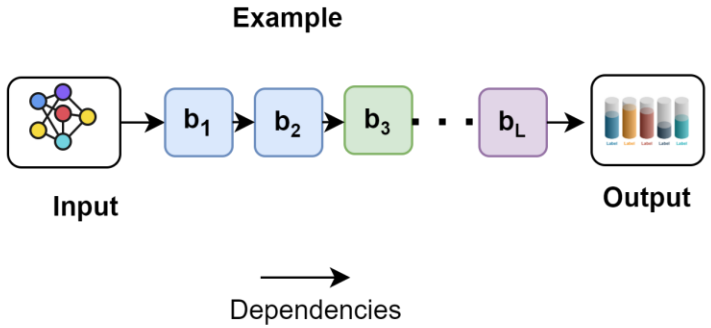


All open source

Formal Model: Blocks, Machines, and a Mapping

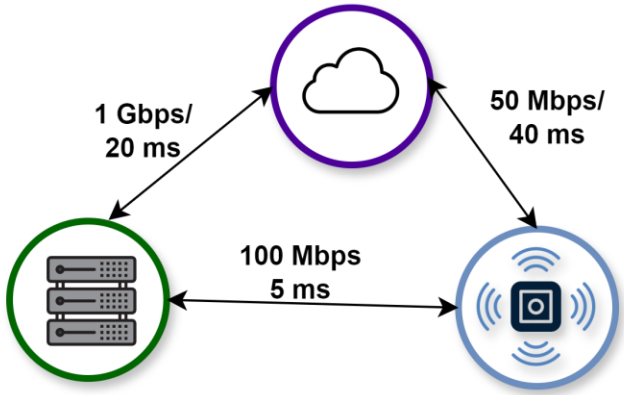
1. The NN model as blocks

A neural-network workload is represented as blocks: groups of operators connected by data dependencies. Each block is an atomic unit we can place on a machine



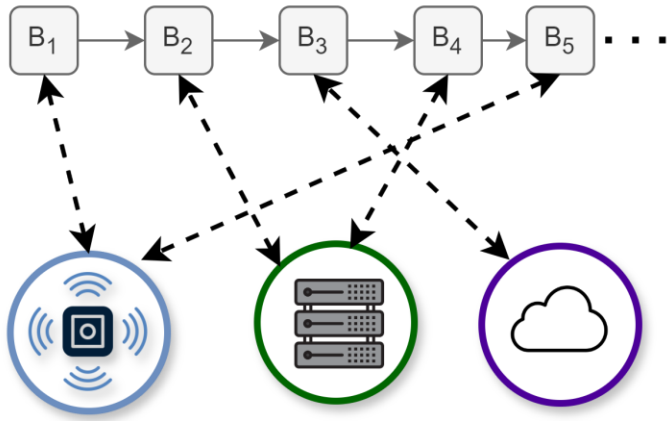
2. The Computing Continuum

The infrastructure is represented as a graph of machines connected by network links. Machines have compute, memory etc., links have bandwidth, latency etc.



3. A mapping (Deployment)

A deployment specifies which block runs on which machine. That choice determines the computation, communication and overall costs



Why this model?

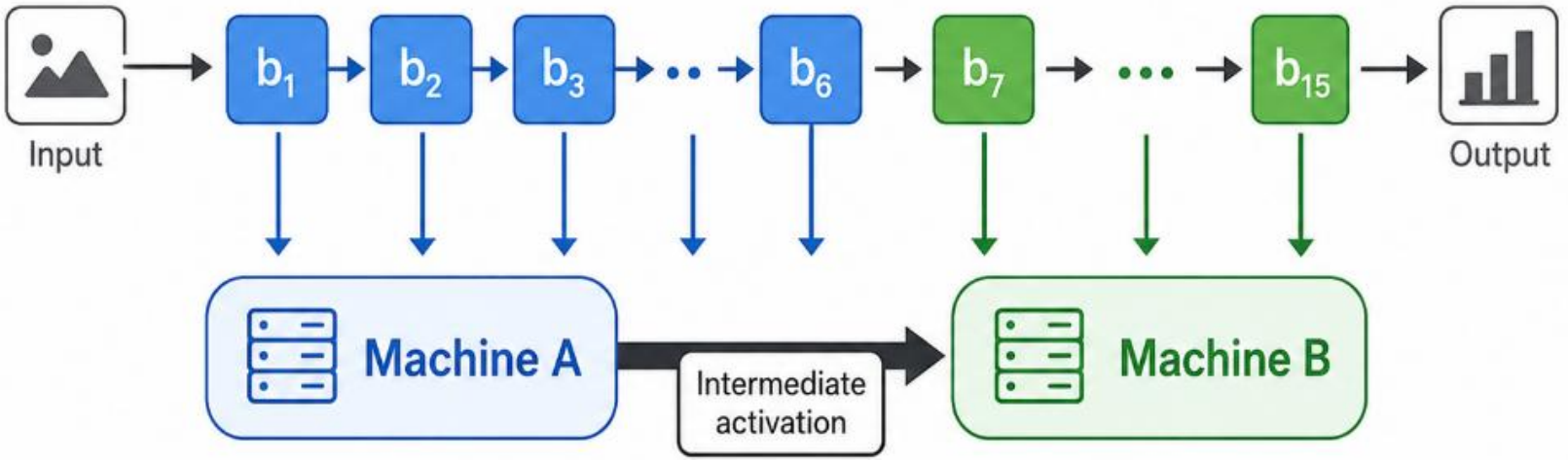
General: works for any neural network architecture

Portable: the same model can describe any setup

Compositional: build complex systems by connecting models

Formal Model: Different Partitioning Strategies (1)

Pipelined Partitioning

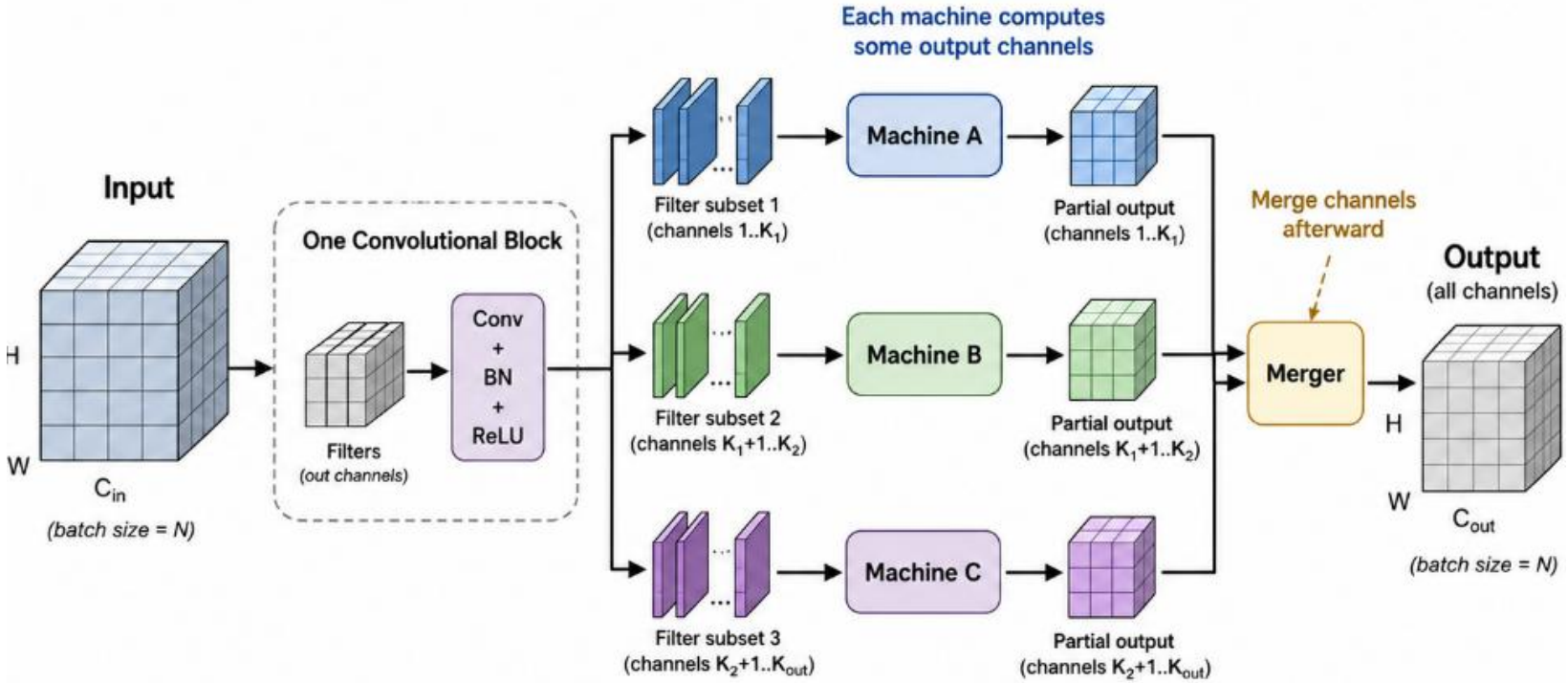


Blocks 1–6 run on A, blocks 7–15 run on B.
One transfer happens in the middle.

Main cost: sending the intermediate result.

Formal Model: Different Partitioning Strategies (2)

Filter Partitioning

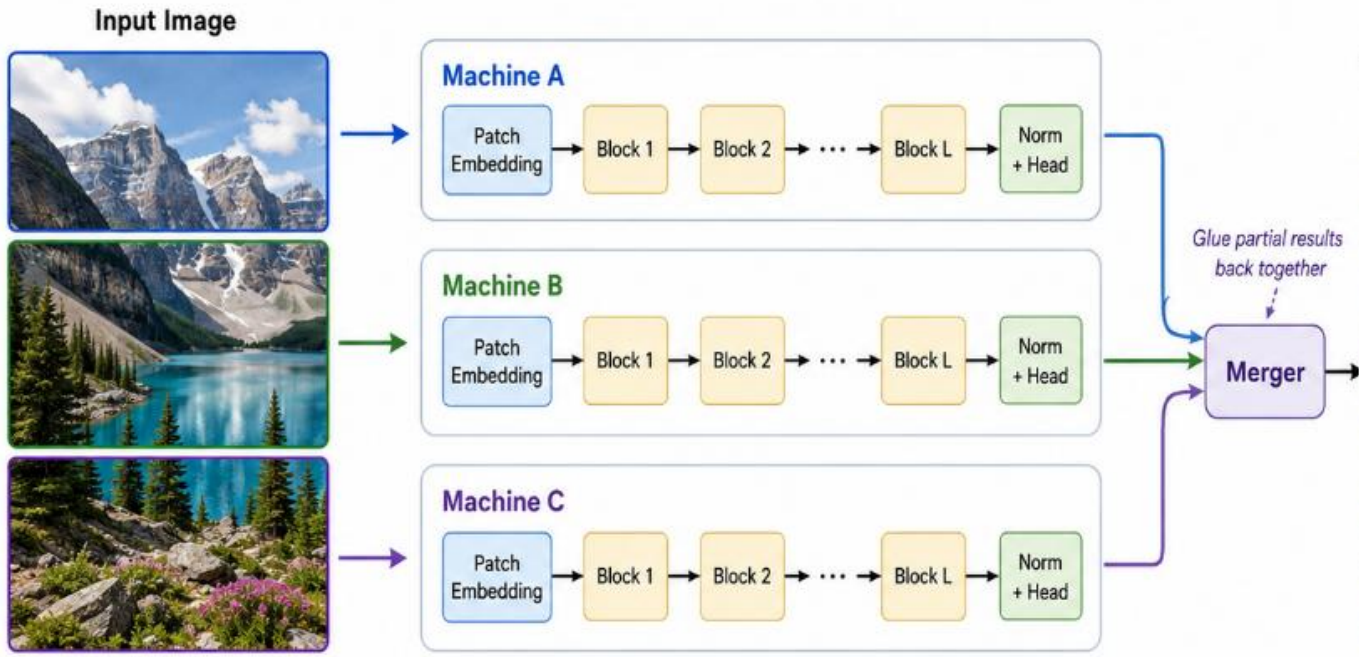


The input is shared, but each machine computes only part of the model's width (some filters / channels).

Main cost: collecting and combining partial channel outputs.

Formal Model: Different Partitioning Strategies (3)

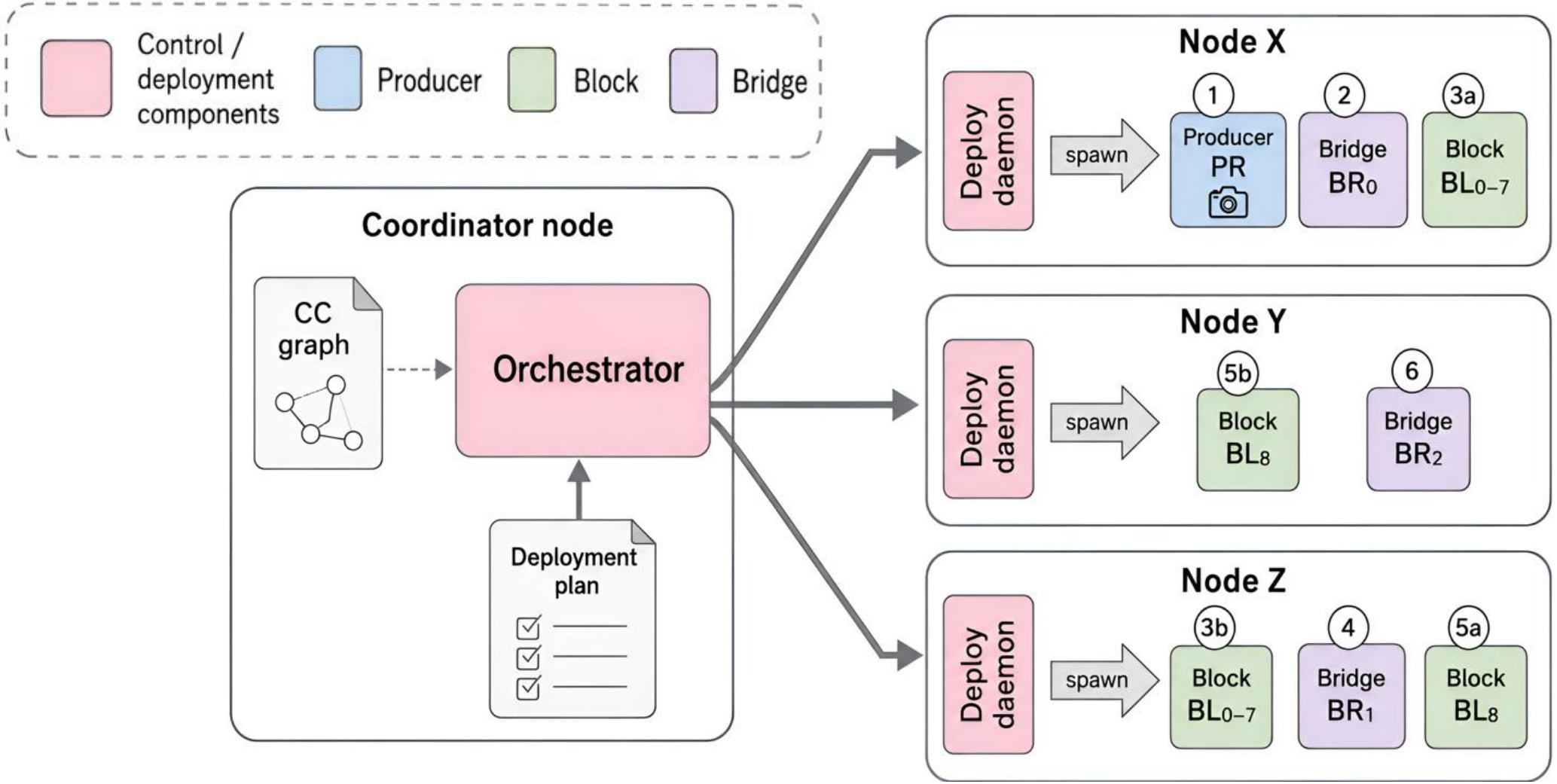
Input Partitioning



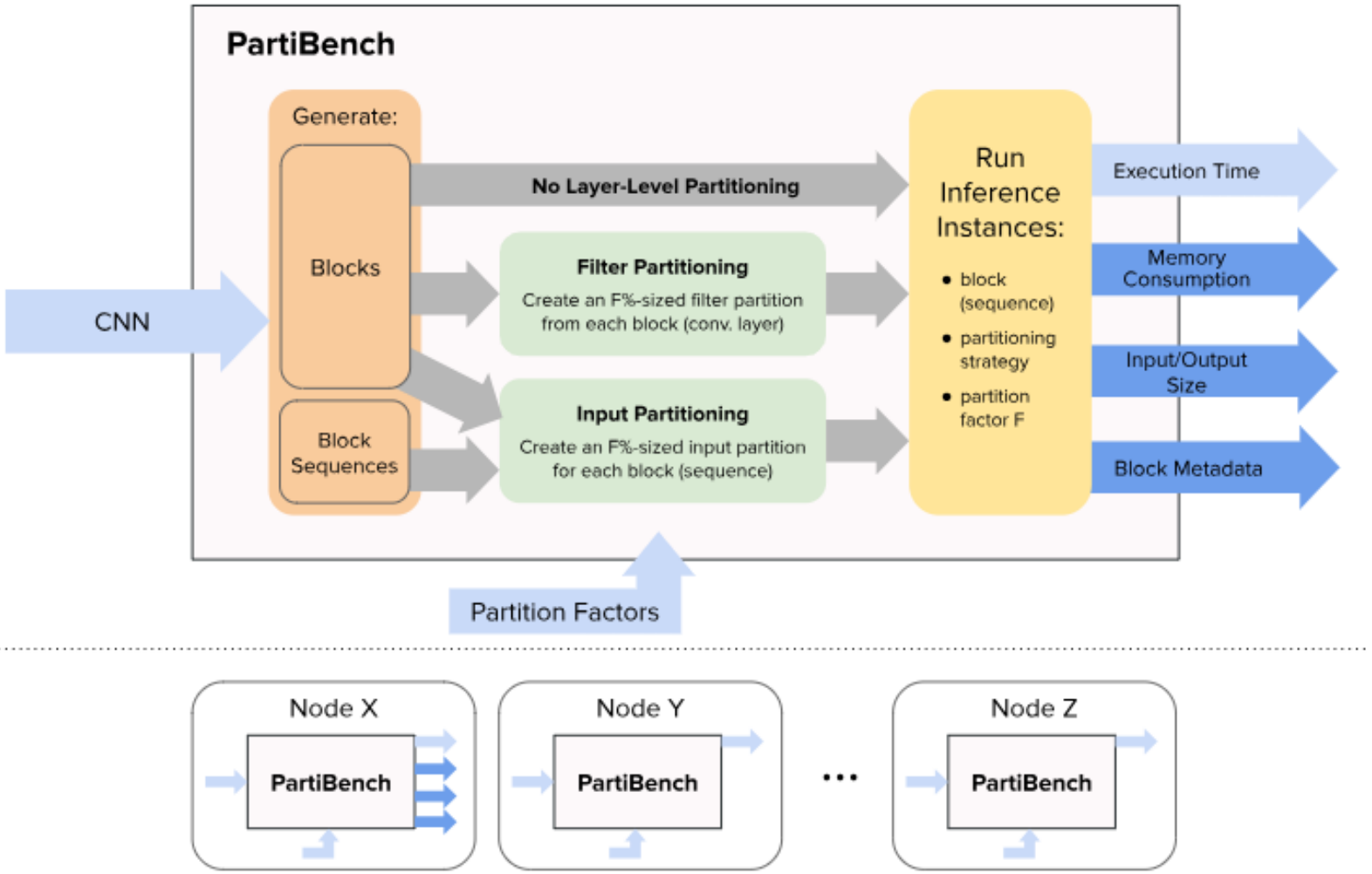
Each machine gets a slice of the input, runs the same blocks, and the merger combines the partial outputs.

Main cost: merging and communication between parallel branches.

A Strategy-Agnostic Distributed Inference Orchestrator

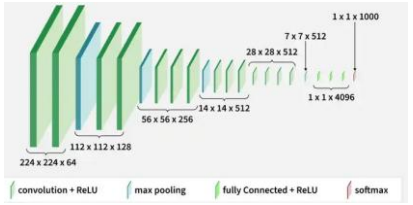


PartiBench: Profiling CNN Segments

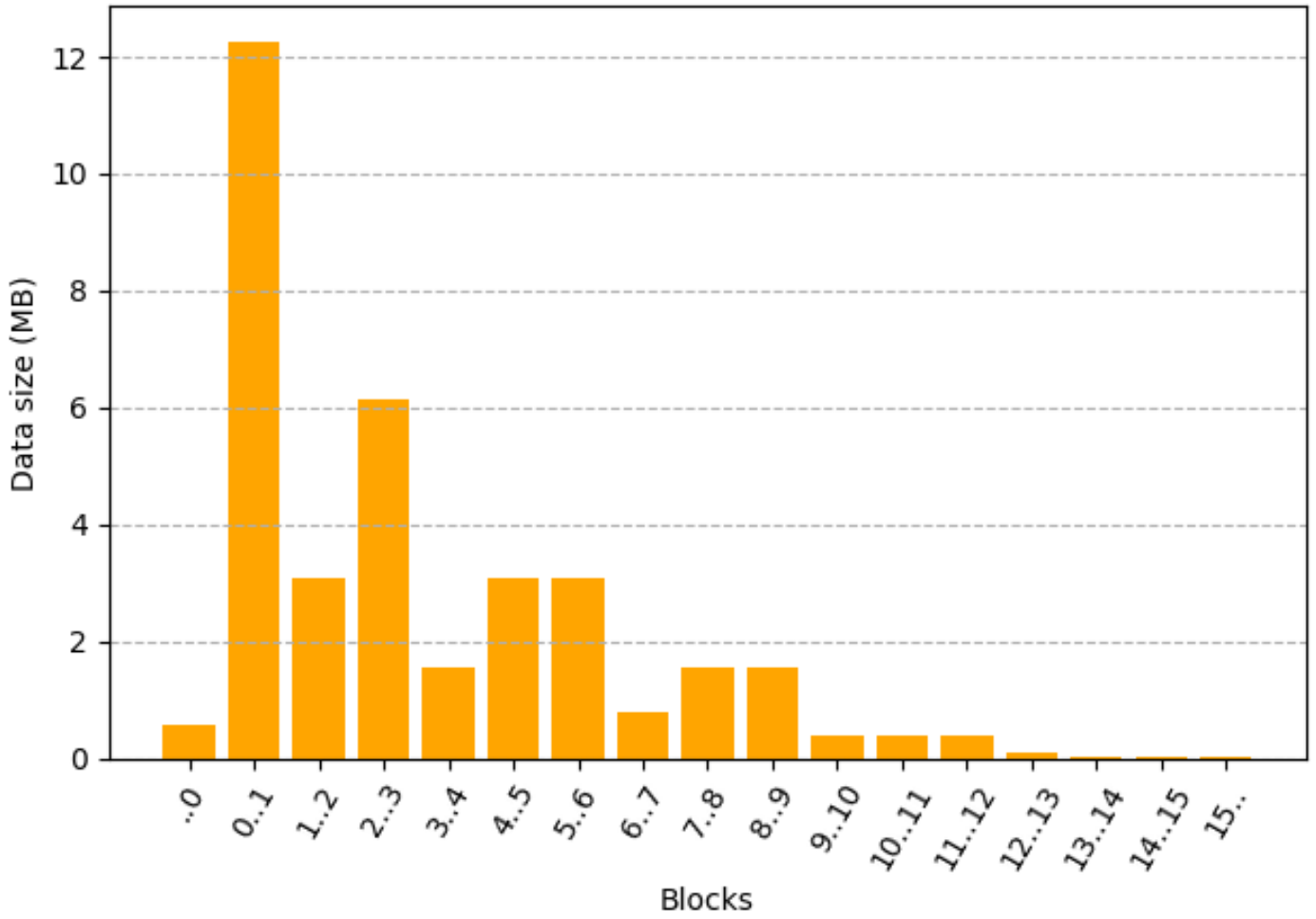


Experimental Setup: An Emulated Three-Tier Continuum

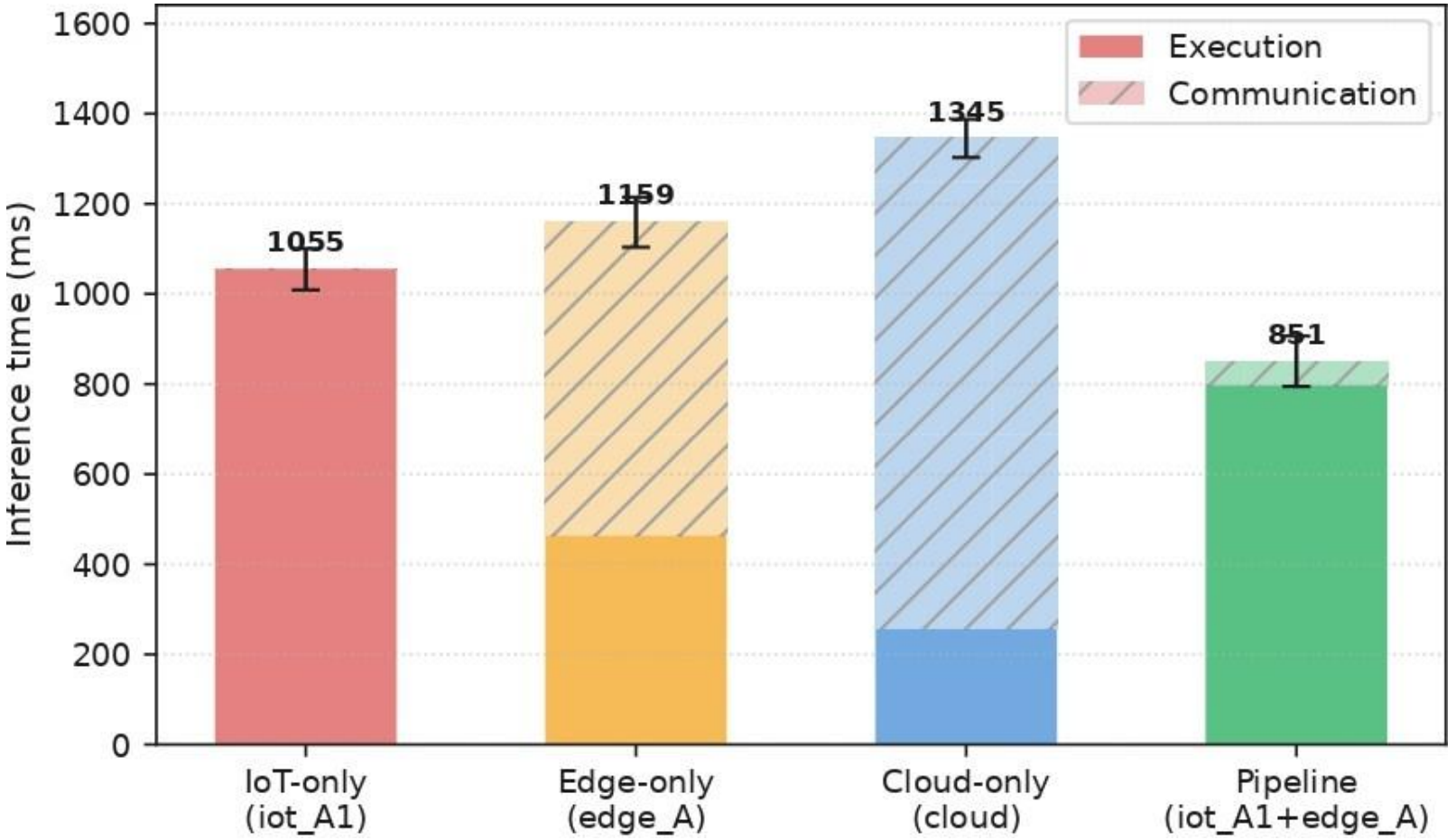
VGG16



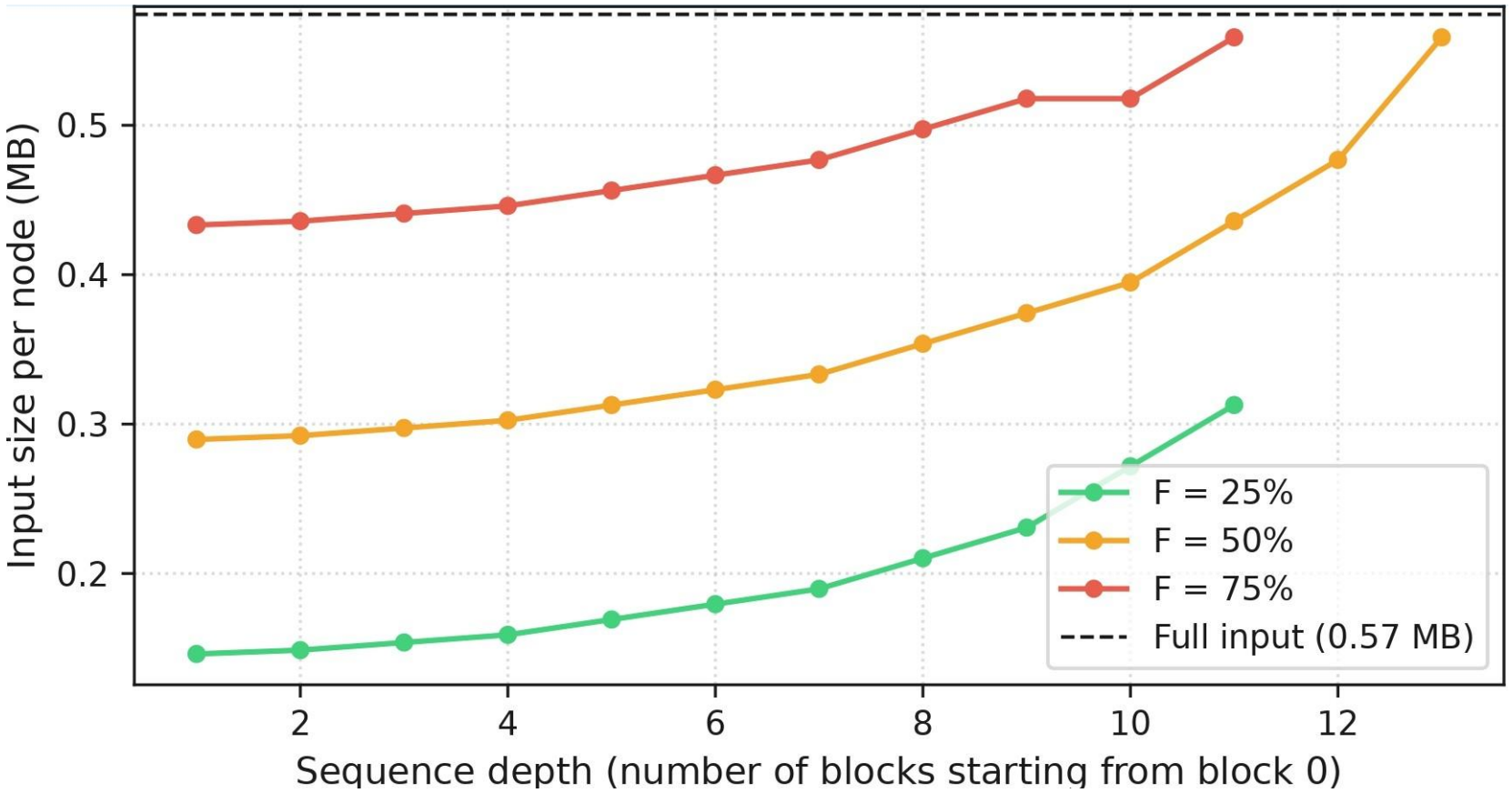
Where to Cut? Profiling Reveals the Cheap Split Points



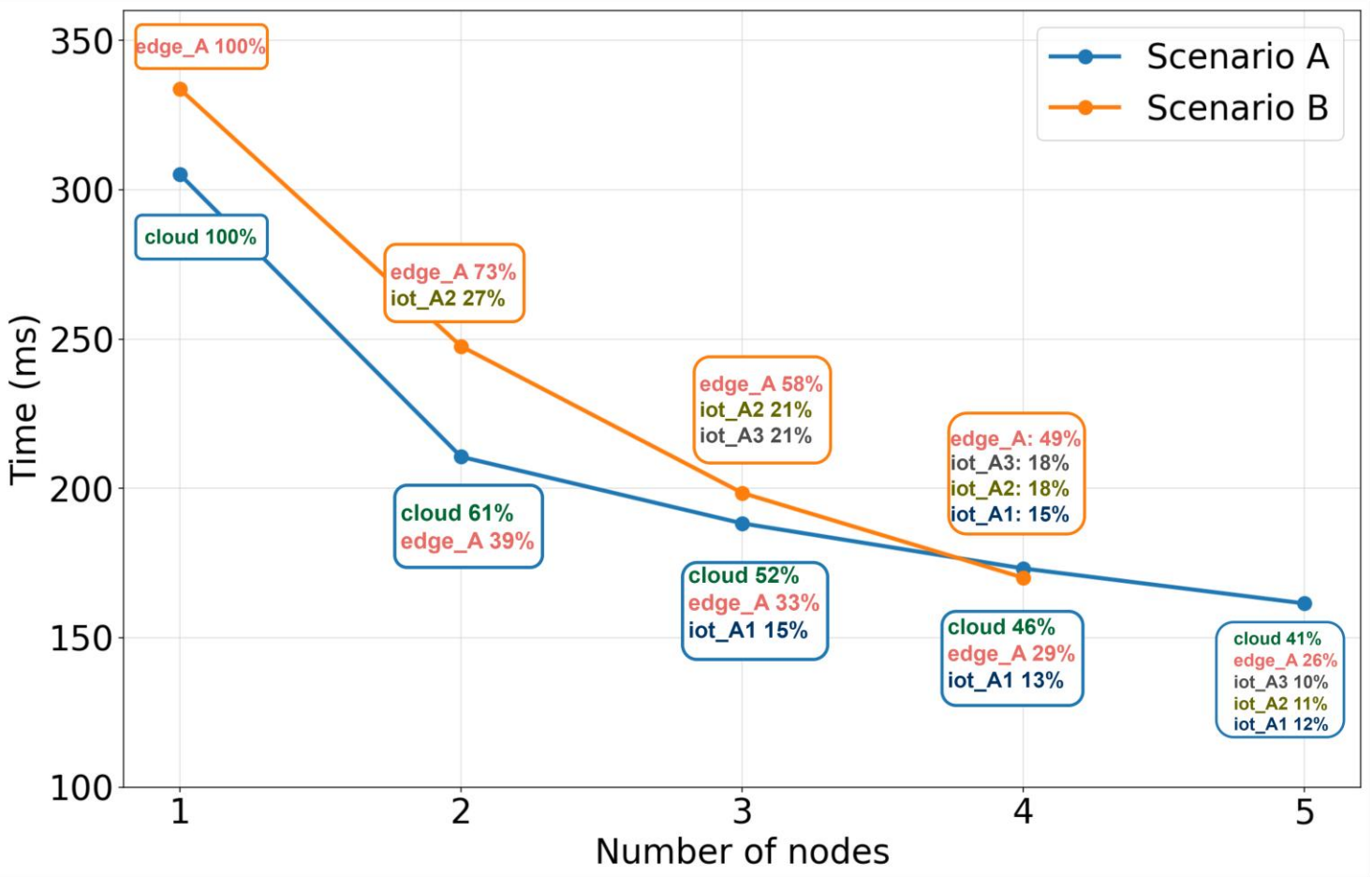
Two-Machine Pipeline vs Every Single-Machine Option



Input Partitioning: Beware the Overlap Trap



When Conditions Change, the Best Deployment Changes Completely



Normal day

5 machines, 161 ms

Cloud takes the biggest share.

Cloud alone:
305 ms

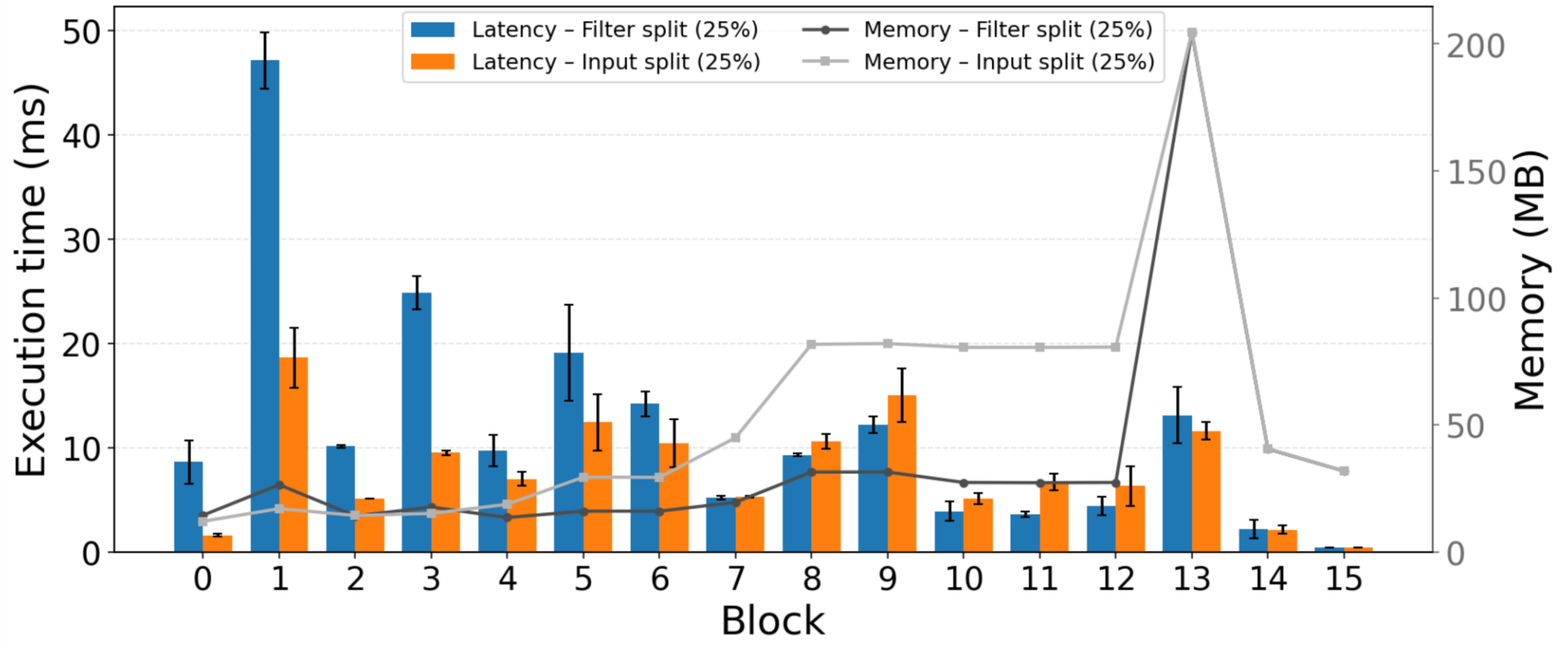
Bad day

Cloud link +80 ms,
source device slowed +200 ms

Best plan uses 4 machines,
cloud kicked out entirely.
170 ms

Edge alone:
334 ms

Filter vs. Input Partitioning



Takeaways & Future Work

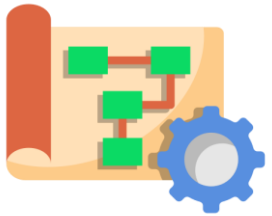
Key takeaways

1. **Placement is context-dependent**
   The best split changes with the model, hardware, network, and current load.

2. **One neutral runtime enables fair comparison**
   Stage, input, and filter partitioning can run on the same infrastructure.

3. **Profiling exposes hidden costs**
   PartiBench measures pieces, not just the whole model: time, memory, data in, and data out.

Future work



1. **Automated placement**
   Use the PartiBench dataset as input to heuristics, search, or learning-based optimizers

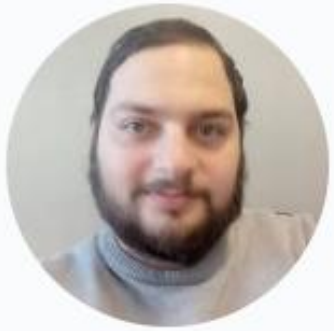
2. **More model families**
   Extend beyond CNNs to Transformers, GNNs, and multi-model inference pipelines.

3. **Dynamic conditions**
   Adapt to load, jitter, failures, and time-varying bandwidth.

4. **Richer resources**
   Support GPUs, accelerators, and energy-aware deployment.

Thank You! Questions?

Thank you!



Nikolaos Papadakis
PhD Student
Télécom SudParis / IP Paris

Feel free to contact me at:

nikolaos.papadakis@telecom-sudparis.eu

PANDORA

Try out Partibench

This Presentation has been designed using resources from Flaticon.com