Optimizing Edge AI: Performance Engineering in Resource-Constrained Environments

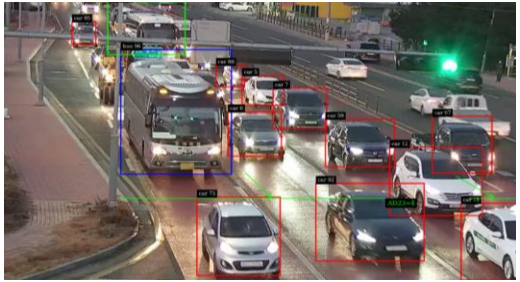
Giuliano Casale

Quality of Service Research Lab (QORE)
Department of Computing - Imperial College London

ACM/SPEC ICPE, 10-May-2024

Edge AI Systems

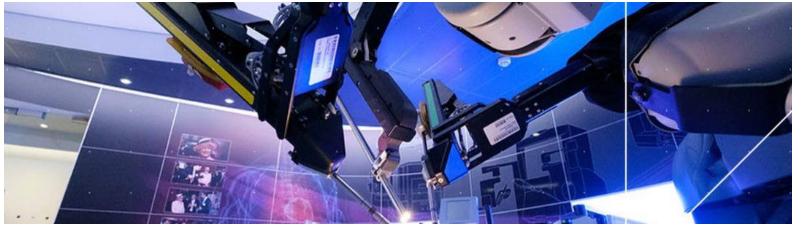
Edge AI focuses on operating AI systems with edge computing


Self-driving cars & AI traffic monitoring

Augmented reality for hospitality

Edge computing goals:

- Ultra-low latency / real-time
- Increased data privacy
- Lower energy footprint


AI-enabled telerobotic surgery

The trend keeps growing:

- Cloud-Edge-IoT continuum
- Edge accelerators
- 5G / 6G
- Tiny ML
- …

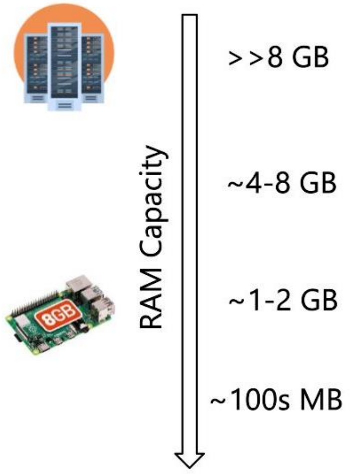
Real-time sport analytics

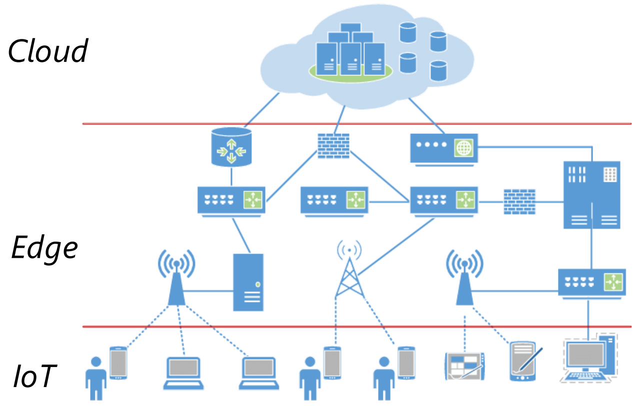

Industrial IoT

Deep Neural Network (DNN) Models at the Edge

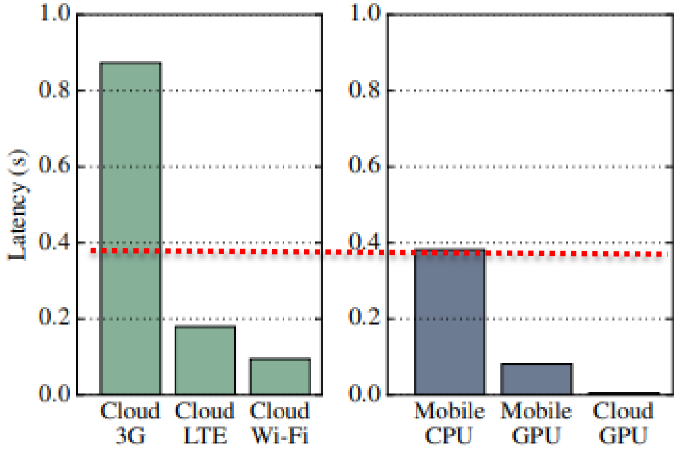
What challenges do we face in performance engineering of Edge AI systems?

- Resource constraints:
  - Task offloading and partitioning of large DNN models
  - MobileNet-v2: ~650MB vs. GPT-v3: ~350GB





Data transfer latency vs. Local processing



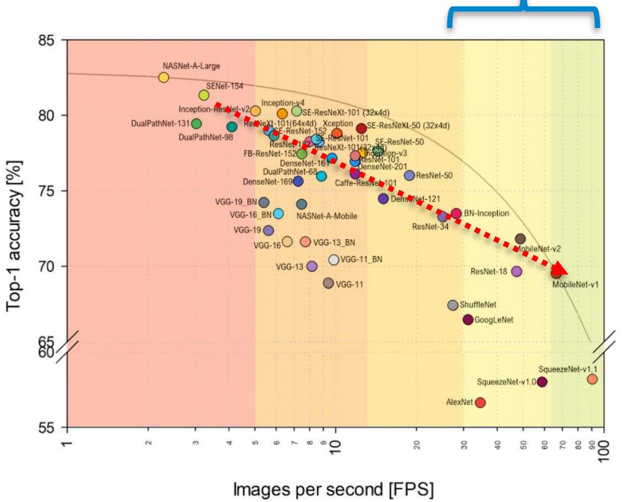
Y. Kang *et al.* Neurosurgeon: Collaborative intelligence between the cloud and mobile edge, Proceedings of ASPLOS.

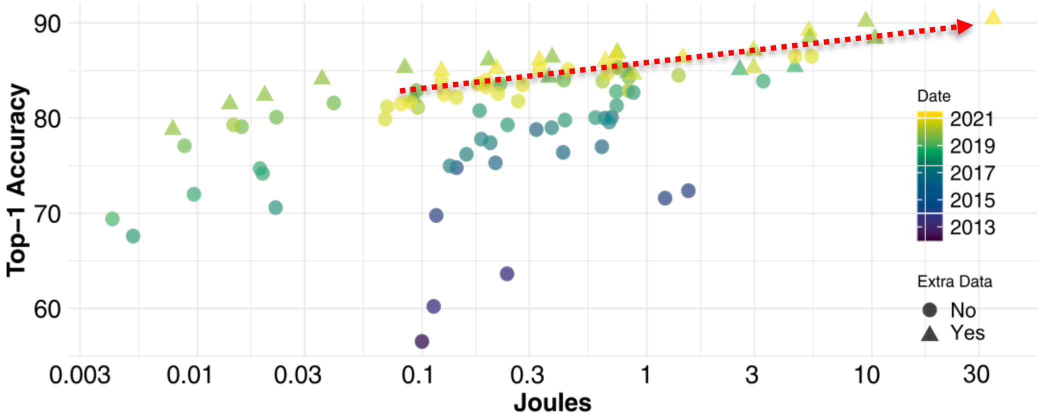
Deep Neural Network (DNN) Models at the Edge

What challenges do we face in performance engineering of Edge AI systems?

- Performance-Accuracy tradeoffs
  - Accuracy not a first-class citizen in classic performance engineering
  - Visible correlations arise with throughput and energy consumption

Real-time



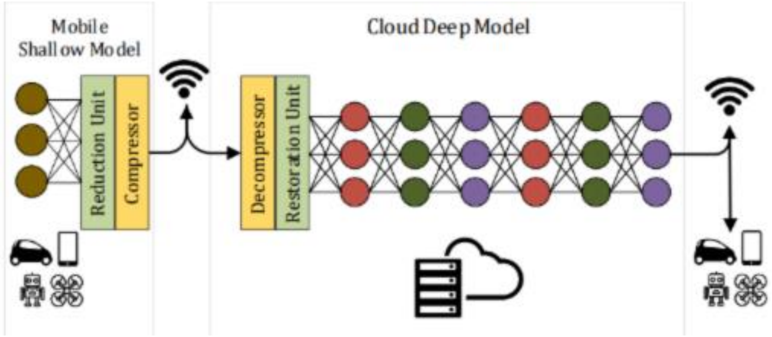


S. Bianco, et al. Benchmark Analysis of Representative Deep Neural Network Architectures, IEEE Access

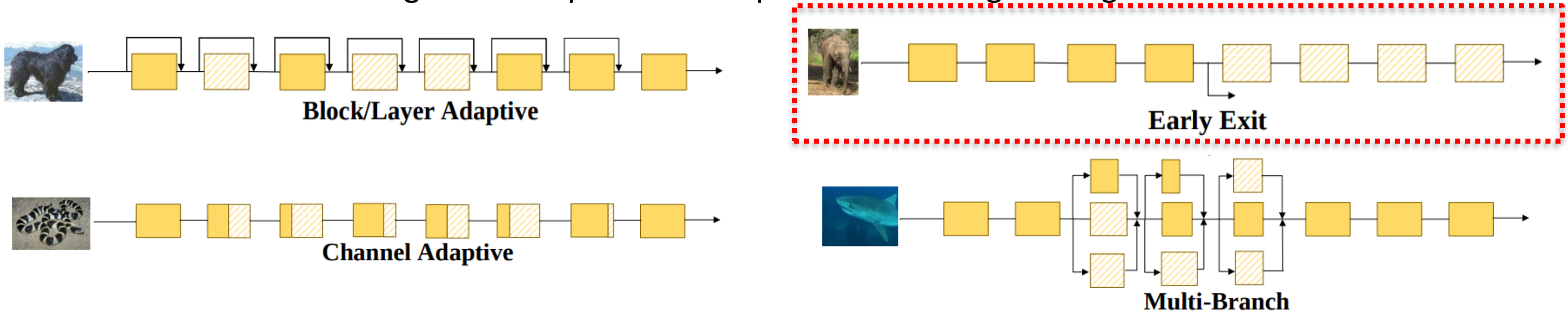
R. Desislavov, et al., Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning, Sustainable Computing.

Deep Learning Approaches to Edge AI performance

- DNN model tuning
  - Weight pruning
  - Quantization
  - Knowledge distillation
  - Lossy compression
  - Neural Architecture Search



- Adaptive DNN models
  - How shall we leverage these capabilities for performance engineering?



D Liu, H Kong, X Luo, W Liu, R Subramaniam. Bringing AI To Edge: From Deep Learning's Perspective. Neurocomputing.
Eshratifar, A. E., et al. BottleNet: A Deep Learning Architecture for Intelligent Mobile Cloud Computing Services, IEEE/ACM ISLPED.

Roadmap

1. How can we leverage early-exits for performance tuning?

   ⇨ Emerging research on scheduling early exits

   TC'23

2. How to partition and deploy DNNs?

   ⇨ Performance-aware online DNN splitting methods

   TMC'23

   ⇨ Combining classic performance evaluation with GNNs

   DSN'24

How can we leverage early-exits for performance tuning?

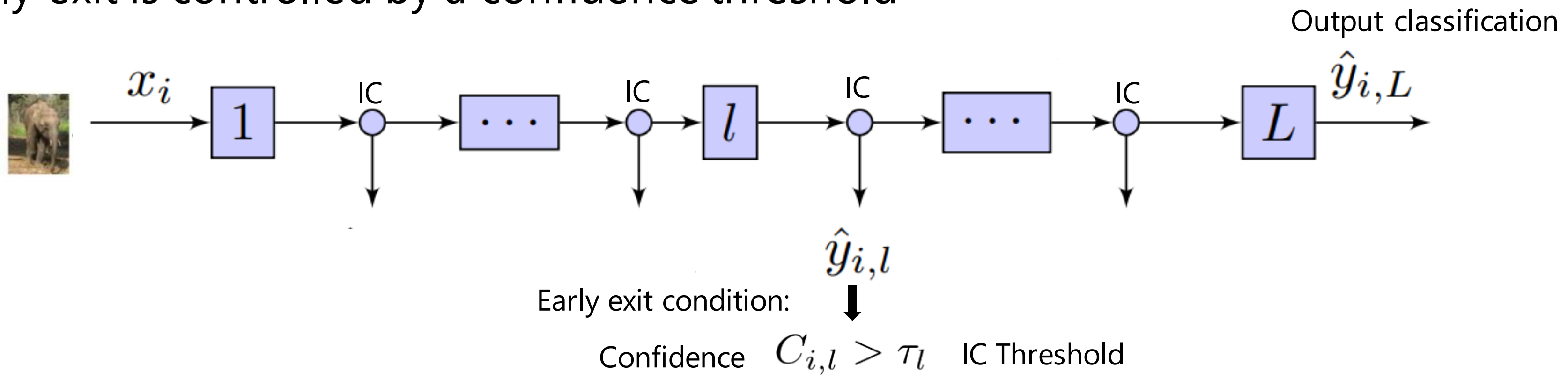
Joint work with:

Manuel Roveri
(Politecnico di
Milano, Italy)

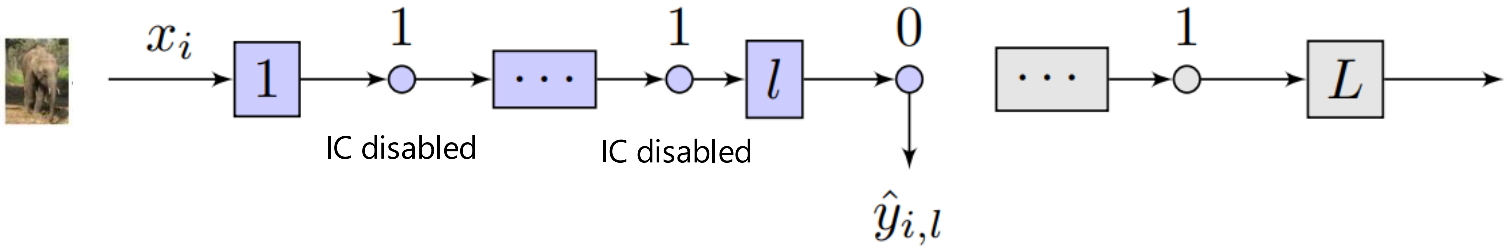
Yichong Chen
(Imperial College
London, UK)

Early exit in CNNs

- An Intermediate Classifier (IC) can produce an early classification output
- Early exit is controlled by a confidence threshold



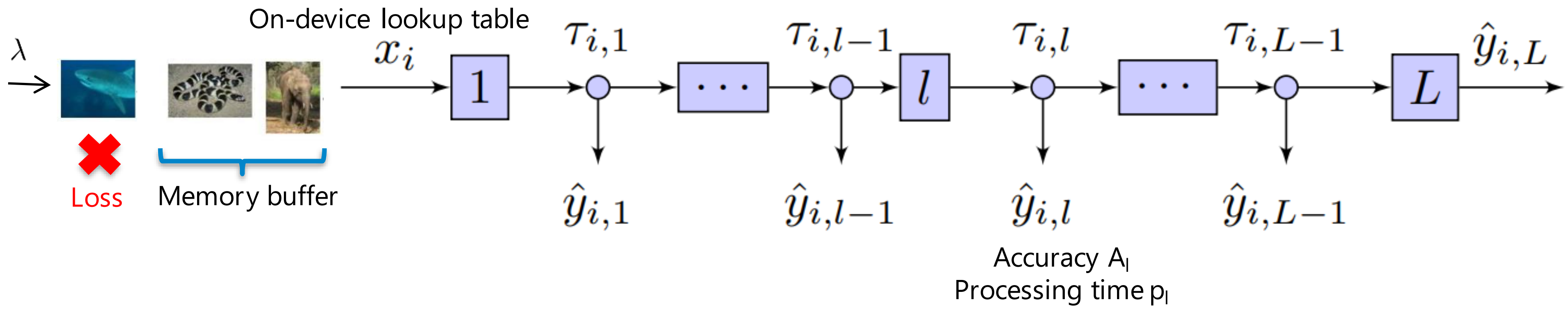
- Example – forcing exit at layer *l*:



- Thresholds trained with the CNN or afterwards.

Scheduling early exits for reliability

- How to schedule early exit online to **reduce data loss**?



- Edge AI system metrics: latency, accuracy, loss ratio.

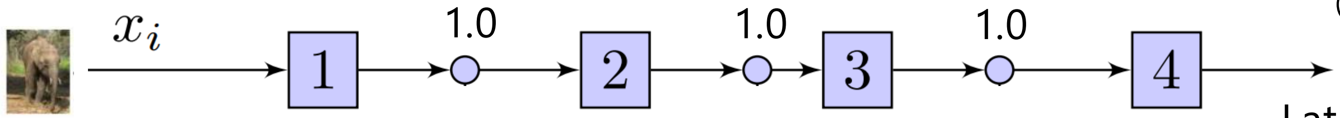
- Research questions:
  - RQ1. How to deal with accuracy and its tradeoffs with performance and reliability?
  - RQ2. What family of scheduling methods are best for early exits?

Casale, G., & Roveri, M. (2023). Scheduling Inputs in Early Exit Neural Networks. *IEEE Transactions on Computers*.

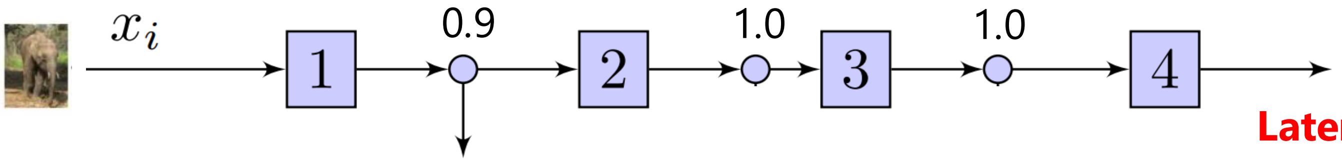
Accuracy in adaptive DNNs

- Accuracy and latency change with the data distribution!

(25-layer CNN + CIFAR10)



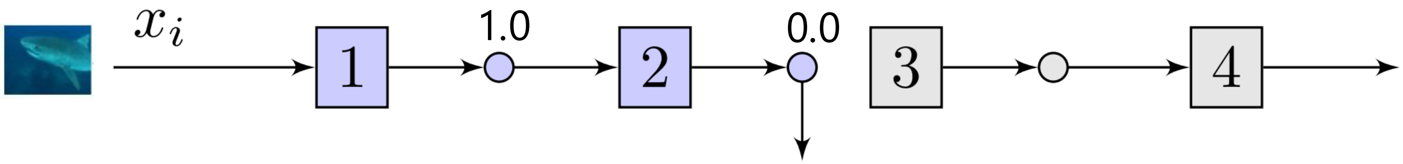
Accuracy: 84.3%
Latency: 138 × 10^6 MACs



Accuracy: 77.9%
Latency: 167 × 10^6 MACs

Accuracy: 82.2%
Latency: 56 × 10^6 MACs

- **Single-exit schedulers**: restrict feasible threshold values to {0,1}



A_2 (profilable offline)

Single-exit knapsack scheduler

- Deterministic scheduling:
  - Without arrivals, reduces to discrete scheduling with compressible resources (NP-hard)
  - A time budget *B* is assigned based on arrival rate after each completion

k spare job slots in buffer

Time budget $B \propto k/\lambda$

On-device lookup table

λ/k	*0*	*1*	*2*	...
0-0.5	$\tau_{i,l}$	...	...	...
0.5-1	...	...	...	...
⋮	⋮	⋮	⋮	⋮

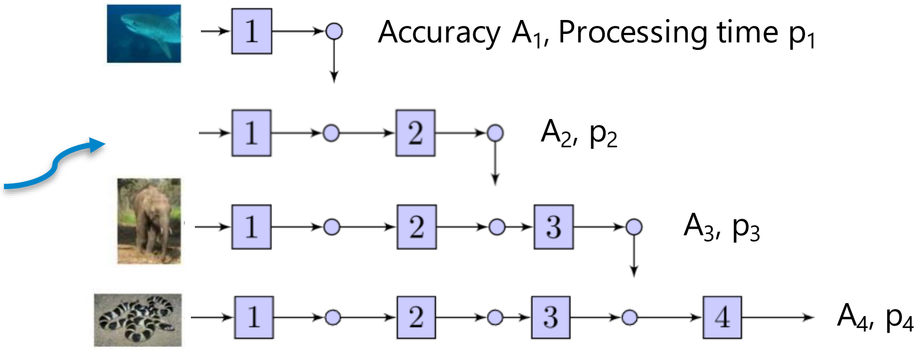
knapsack problem

$$\max \sum_{l \neq l_{min}} \sum_{j=1}^{k} A_l x_l$$ Maximize accuracy

$$\text{s.t.} \sum_{l \neq l_{min}}^{L} \sum_{j=1}^{k} p_l x_l \leq B - k p_{l_{min}}$$ Fit time budget

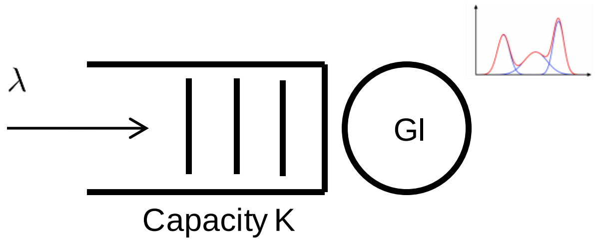
$$\sum_{l \neq l_{min}}^{L} x_l \leq k$$ Schedule at most k jobs

$$x_l \in \{0, \ldots, k\} \quad \forall l$$ Num. jobs to exit at layer l



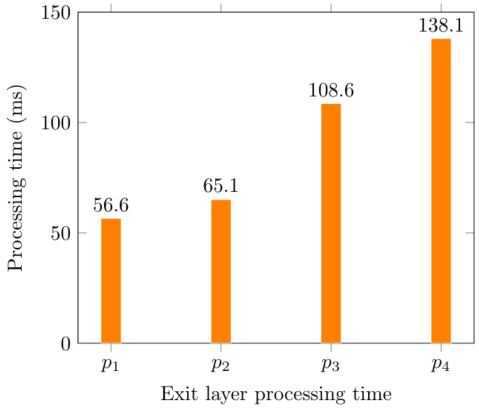
Single-exit queueing scheduler

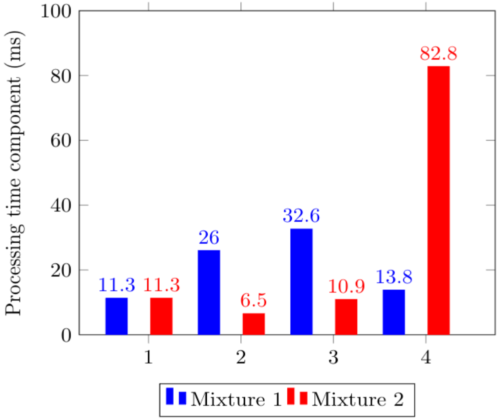
- Stochastic scheduling:
  - DNN latency from steady-state M/GI/1/K queue
- Control knob: probability of exiting after a DNN layer
  - Service becomes a mixture distribution (GI)
- Optimal schedule obtained via a Linear Program (LP)
  - Maximize target accuracy
  - Constraint on maximum acceptable loss ratio

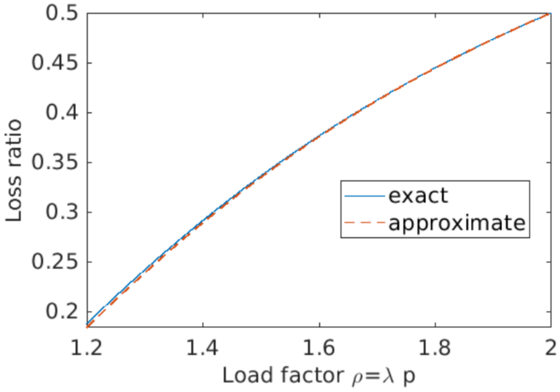


Loss ratio approximation

$$L = \frac{\rho^{(\sqrt{\rho}s^2-\sqrt{\rho}+2K)/(2+\sqrt{\rho}s^2-\sqrt{\rho})}(\rho-1)}{\rho^{2(1+\sqrt{\rho}s^2-\sqrt{\rho}+K)/(2+\sqrt{\rho}s^2-\sqrt{\rho})}-1}$$

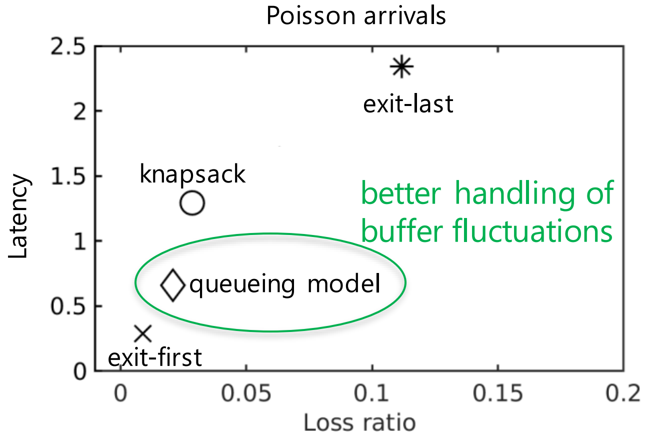


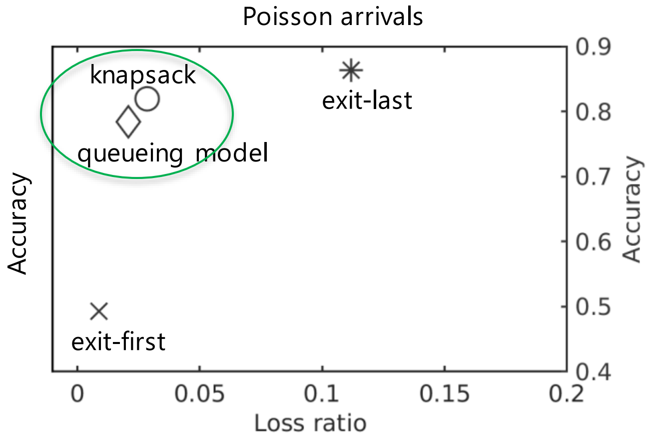


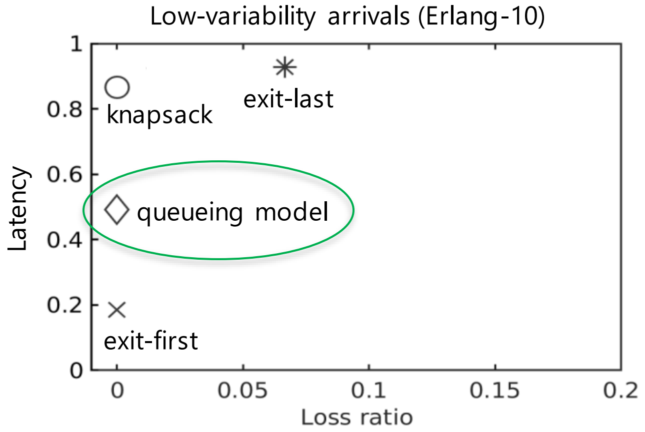


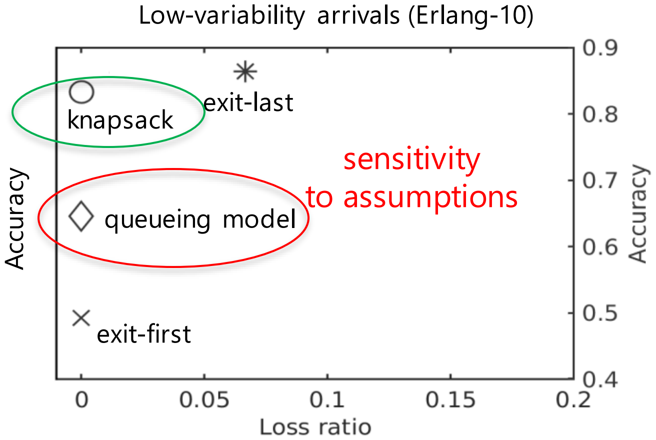
Simulation of real technological scenarios

- 6 CNNs (28-56 processing layers; 8-24 exit points; CIFAR10/100 data)



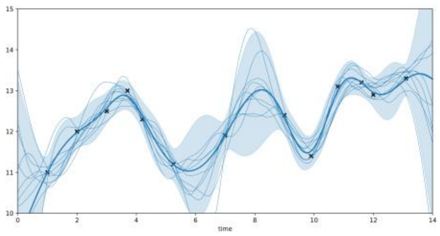


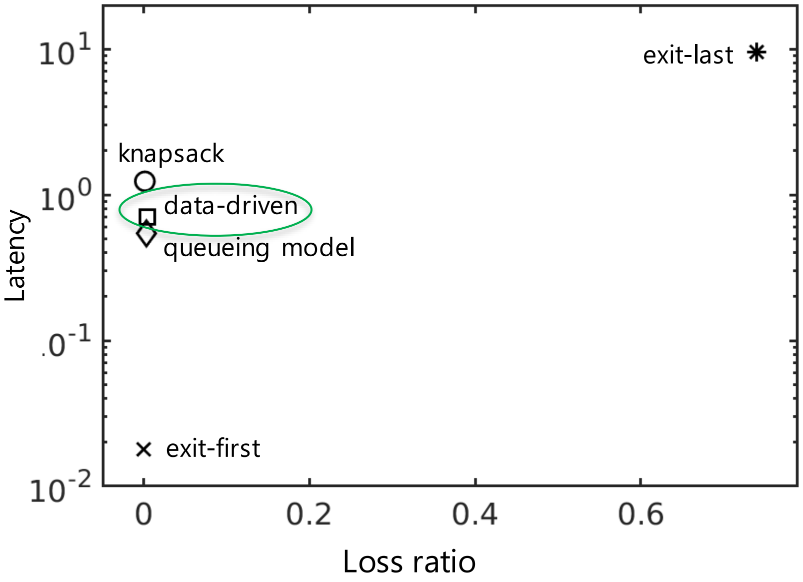


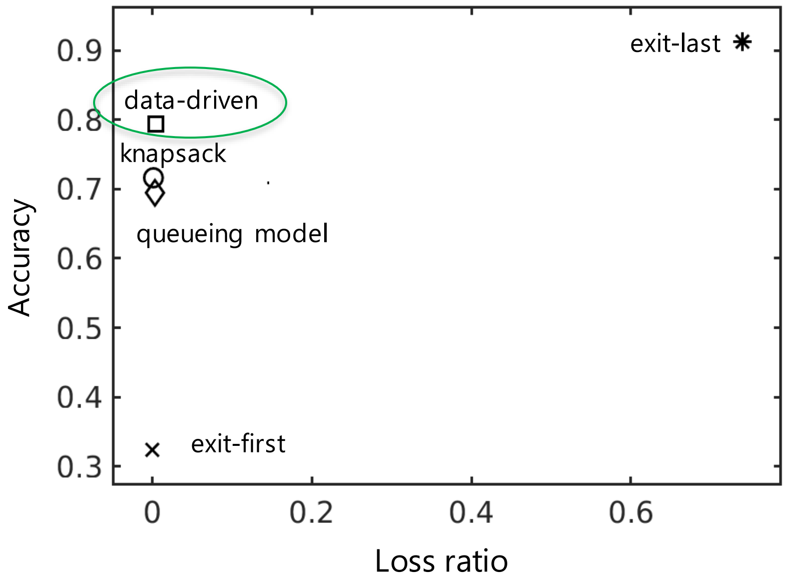


Data-driven multi-exit scheduling

- Cope with data-dependent accuracy while training
  - Thresholds now arbitrary
  - Trial-and-error via Bayesian Optimization
  - Optimize accuracy and M/GI/1/K loss
- Results from actual rPI 4B system

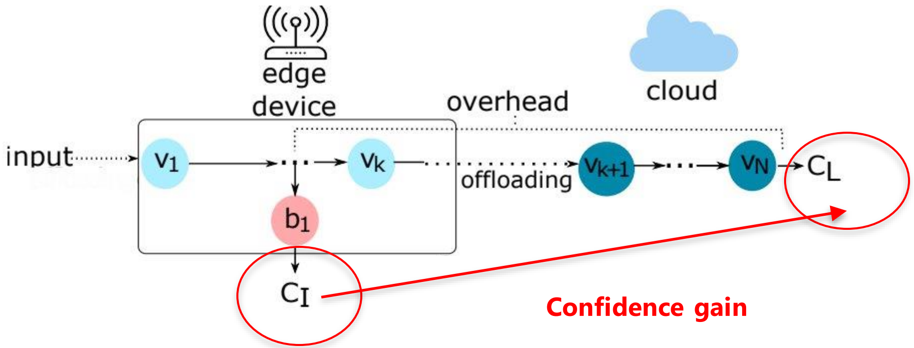






Setting confidence thresholds in evolving environments

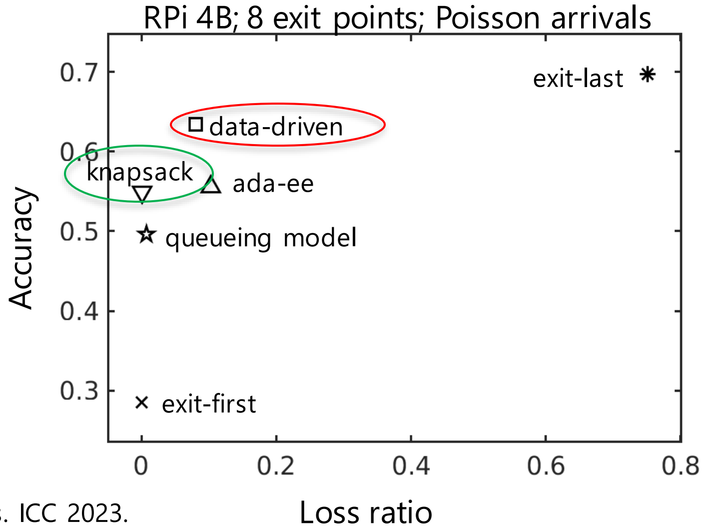
- AdaEE: multi-armed bandit (MAB) to schedule early exits
  – Reward metric: Confidence gain - Performance overhead



Upper Confidence Bound (UCB):

$$\tau_{i,l} \leftarrow \arg\max_{\tau_{i,l} \in \mathcal{A}} \left(Q_{i-1} + c\sqrt{\frac{\ln(i)}{N_{l-1}}} \right)$$

Exploitation   Exploration



- Driving early-exits with confidence gains
  – Single-exit: <1s to update the policies
  – Data-driven difficult to operate online:
    - minutes to update the policies
    - significant computational footprint

R. G. Pacheco *et al.* AdaEE: Adaptive Early-Exit DNN Inference Through Multi-Armed Bandits. ICC 2023.

Takeaways on early-exits

Metric	**Offline (accuracy-based)**	**Online (confidence-based)**
Accuracy	Data-driven	Knapsack
Latency control	Data-driven & Queueing model	Queueing model
Loss control	Data-driven & Queueing model	Queueing model
Robustness across scenarios	Data-driven & Knapsack	Knapsack

Some takeaways:

- We can tap into early-exit ICs to establish a novel performance tuning control knob
- Deterministic methods are highly robust
- Confidence gain offers a viable surrogate to accuracy
- Queueing-informed data-driven scheduling is promising for future research

Open challenge:

- How to schedule early-exits in the distributed setting?

Performance-aware DNN splitting and placement

Joint works with:

Zifeng Niu
(Imperial College
London, UK)

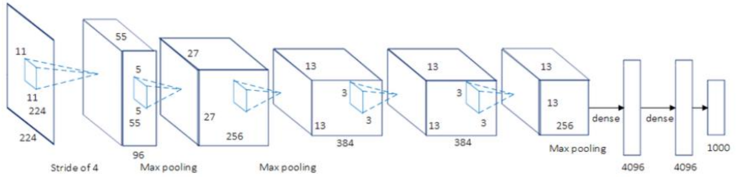
Shreshth Tuli
(Imperial College
London & Happening,
UK)

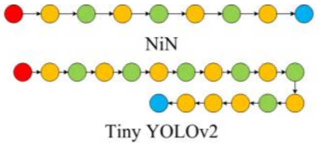
Manuel Roveri
(Politecnico di
MIlano, Italy)

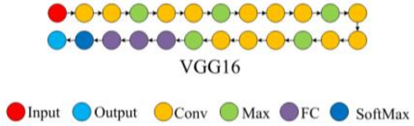
Nick Jennings
(Loughborough U,
UK)

State-of-the-art: DNN layer-wise partitioning

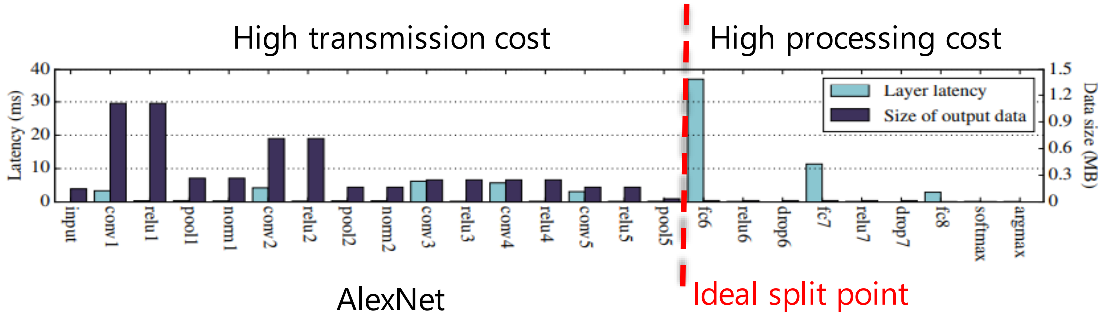
- Many popular DNNs have a linear topology (chain)







- Ideal split point determined from layer characteristics
  - Convolutional: large output data, Pooling: smaller output data; FC layers: high latency
  - Prediction on processing time on target hardware obtained via regression



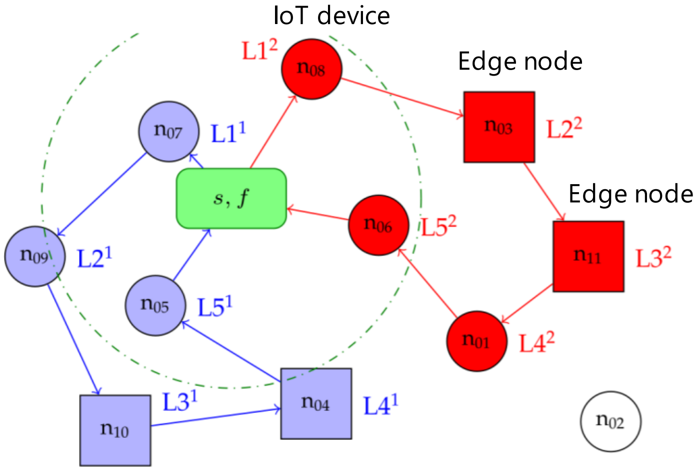
AlexNet

Y. Kang *et al.* Neurosurgeon: Collaborative intelligence between the cloud and mobile edge, Proceedings of ASPLOS.

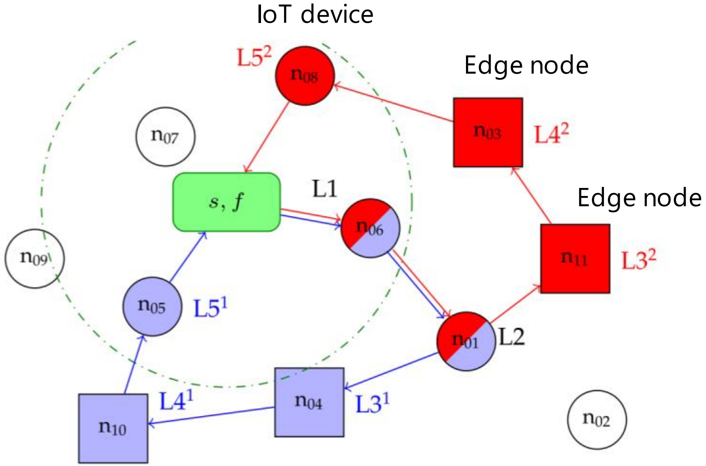
H. Liang *et al.* DNN Surgery: Accelerating DNN Inference on the Edge Through Layer Partitioning, IEEE TCC 2023.

Designing an Edge AI placement

- DNN placement is critical, e.g. IoT devices without on-air update
- State-of-the-art mainly relies on integer-linear programming (ILP)
  - Binary variables map layers to edge & IoT nodes
  - Constraints on memory, processing time, DNN layer dependencies, network range, …



No shared DNN layers

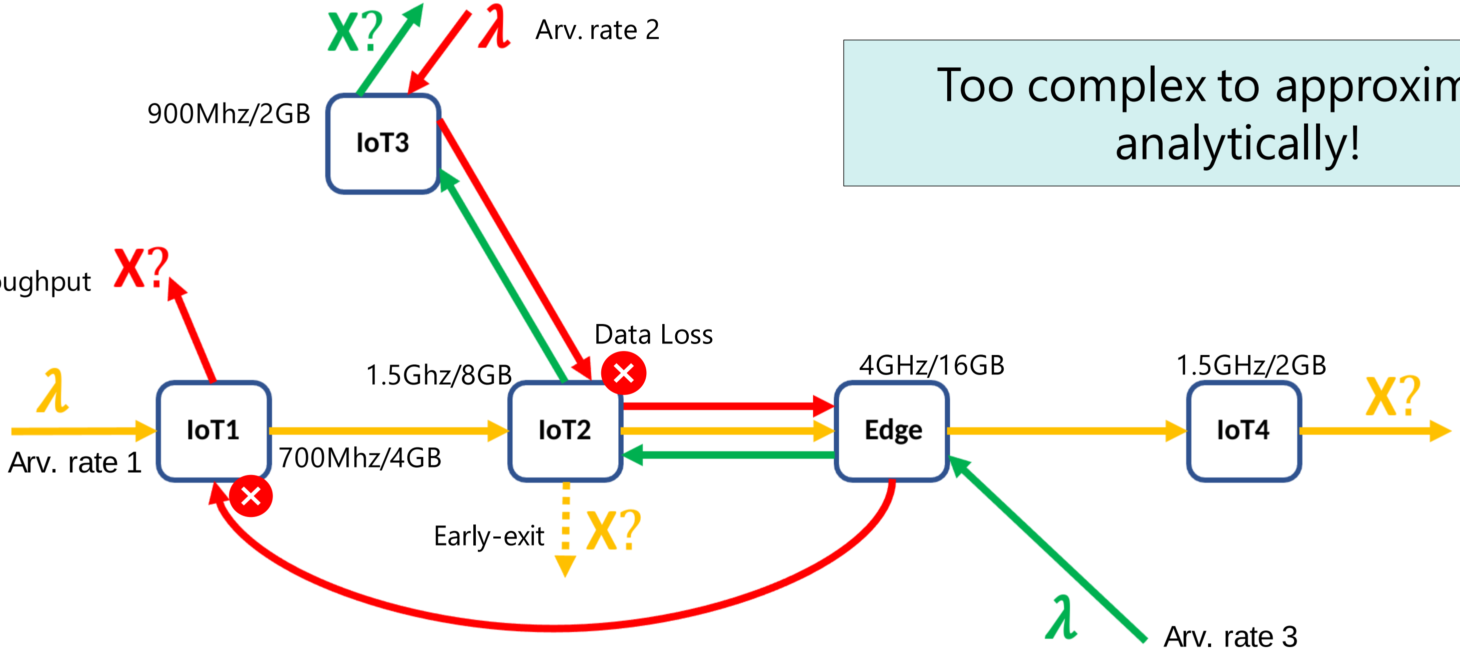


Shared DNN layers

S. Disabato, M. Roveri, C. Alippi. Distributed Deep Convolutional Neural Networks for the Internet of Things. IEEE TC, 2021.

Modeling data loss metrics

- Graph-based deterministic models are appropriate for periodic workloads
- The same approach cannot easily capture stochastic arrivals

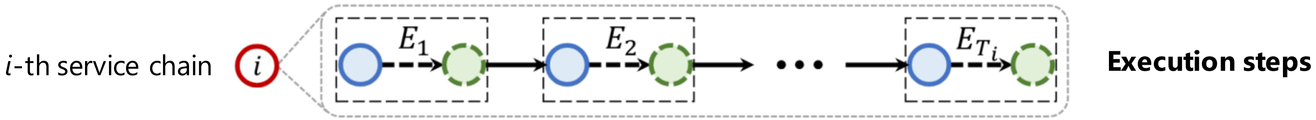


Too complex to approximate analytically!

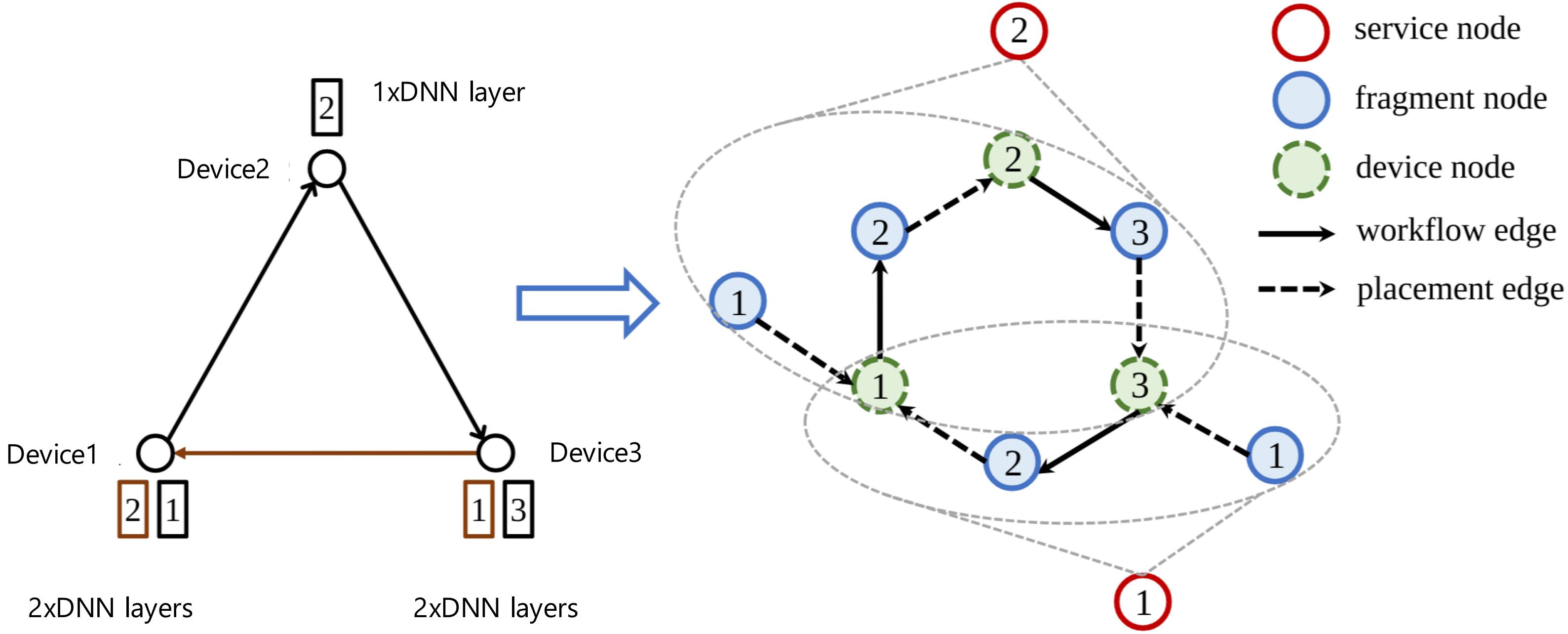
S. Disabato, M. Roveri, C. Alippi. Distributed Deep Convolutional Neural Networks for the Internet of Things. IEEE TC, 2021.
S. Suresh, W. Whitt, The heavy-traffic bottleneck phenomenon in open queueing networks, Oper. Res. Lett. 9 (6) (1990) 355–362.

Modeling an Edge AI placement as a GNN

- We focus on linear DNN pipelines (a service chain)

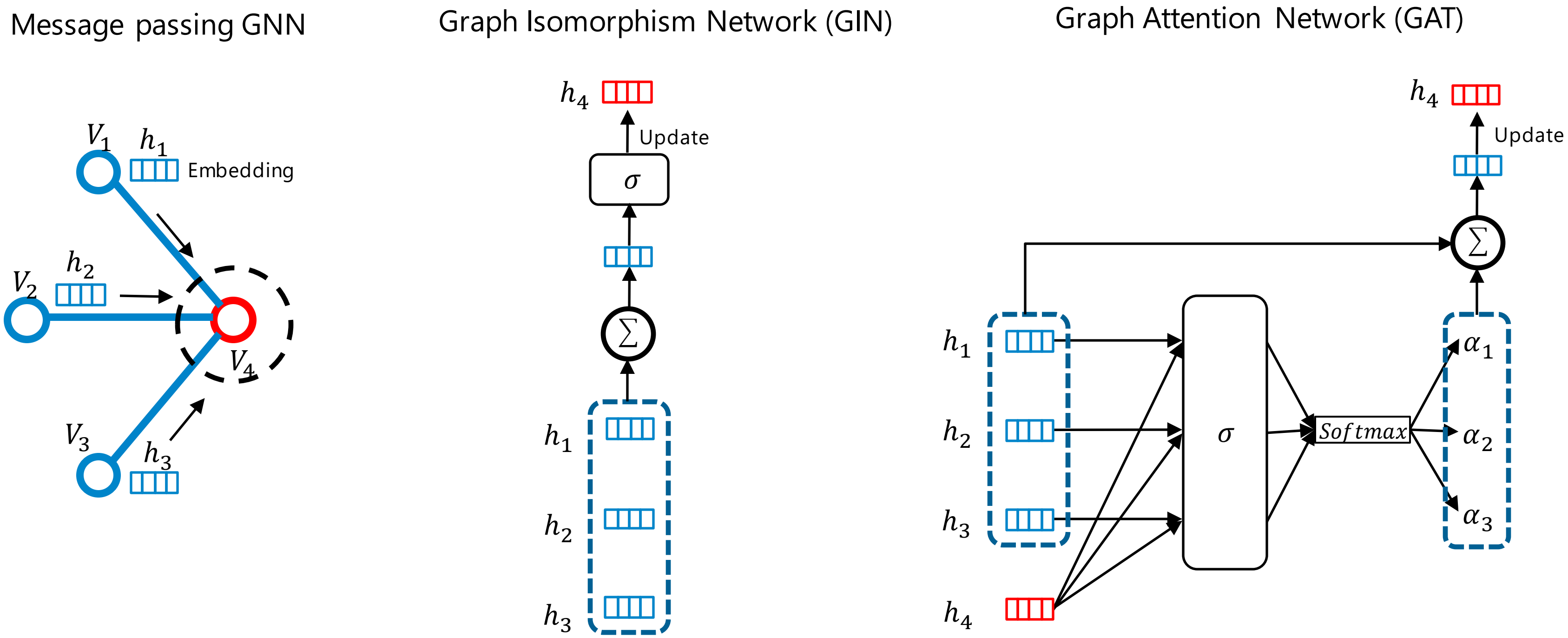


- DNN placement seen as a heterogeneous graphs



Graph neural networks for Edge AI performance prediction

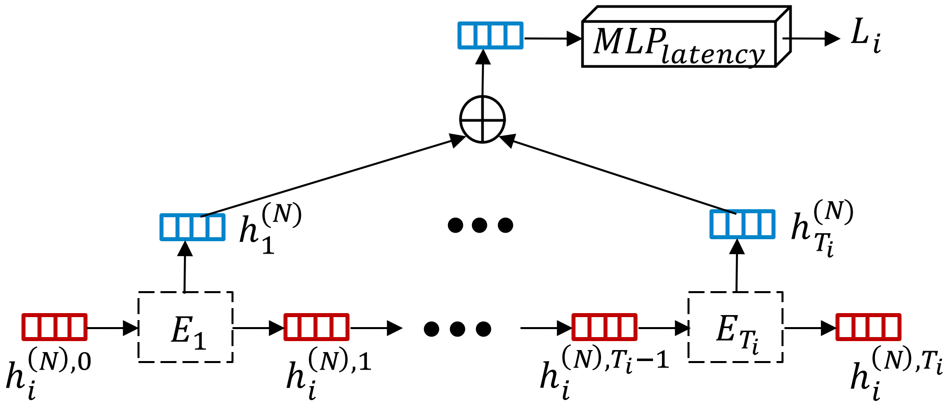
- GNN surrogate trained on simulation and/or system data
  - Input features: system and workload parameters: arrival rates, RAM size, CPU GHz, ...
  - Output features performance metrics: throughputs, latencies, loss ratio, ...



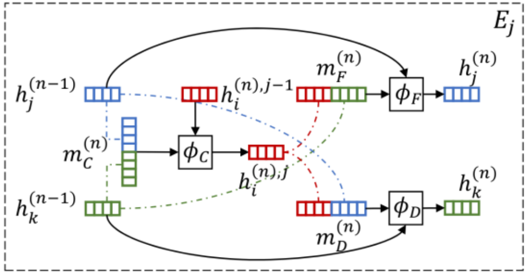
ChainNet: a customized GNN for performance prediction

- Off-the-shelf GNNs need to implicitly learn performance laws from scratch
- ChainNet: a customized GNN tailored to system performance metrics
  - End-to-end service flow embedding:
  - DNN fragment embedding:
  - Device embedding:

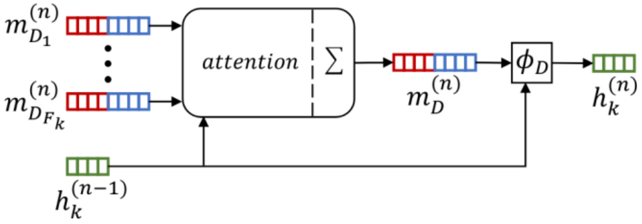
Message passing across execution steps



Message passing within execution steps

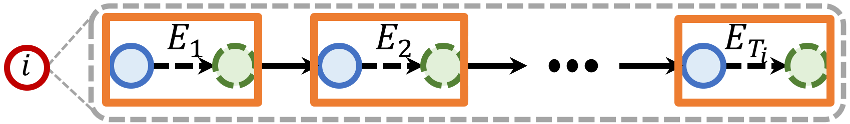


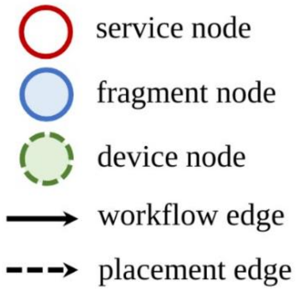
Attention to model "multiclass" interactions

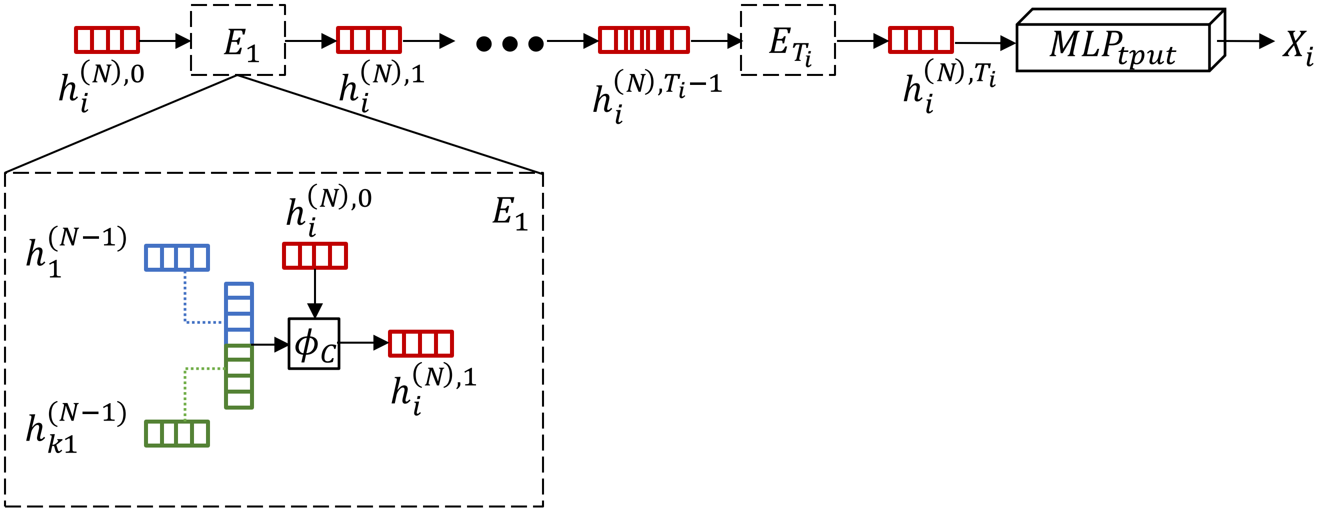


ChainNet GNN: custom message passing

- Modelling throughput in ChainNet

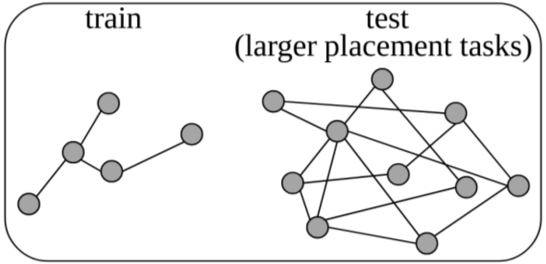


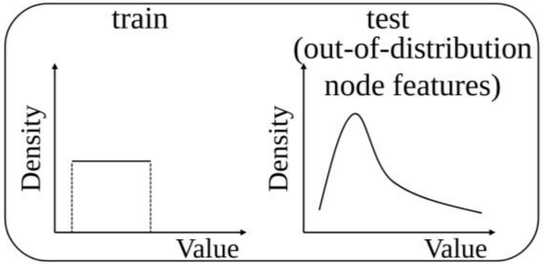


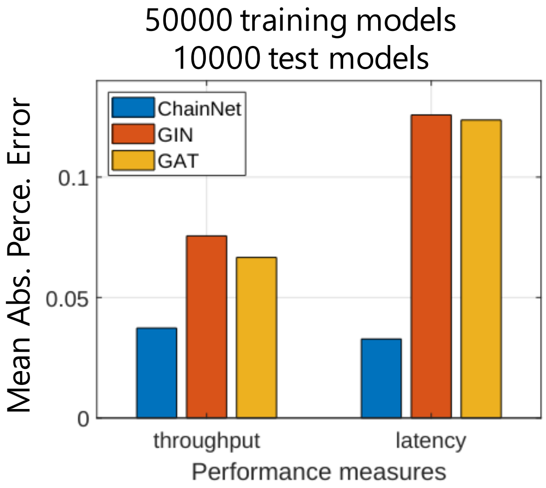


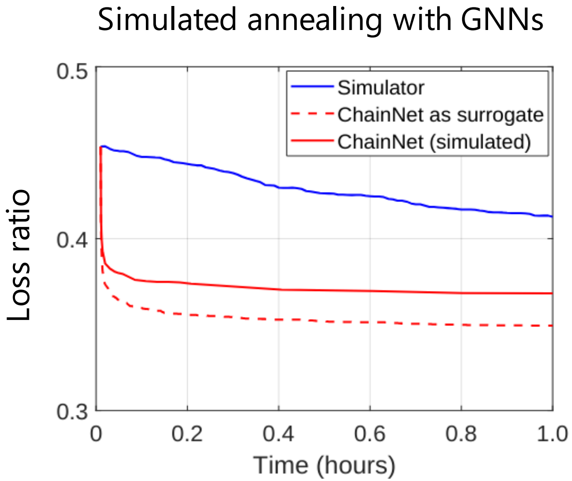
ChainNet: results

- 71% loss ratio reduction in real-world technological scenario
  - 2×OrangePi Zero, 2×Raspberry Pi A+, and 1×Raspberry Pi 3A+
- Systematic reduction also visible in generalization tests via simulation



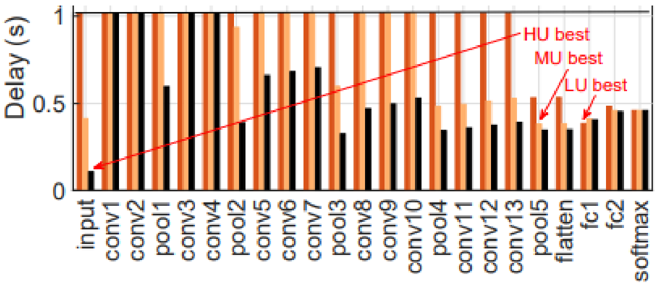






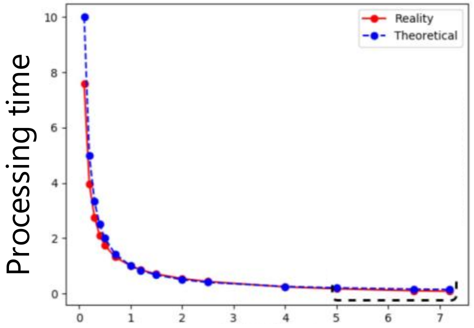
Online DNN layer-wise splitting

- Ideal split point can vary over time with the uplink speed



VGG-16 (High/Medium/Low uplink speed)

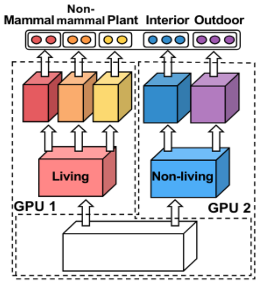
- Actual performance on the device also varies non-linearly with the number of cores



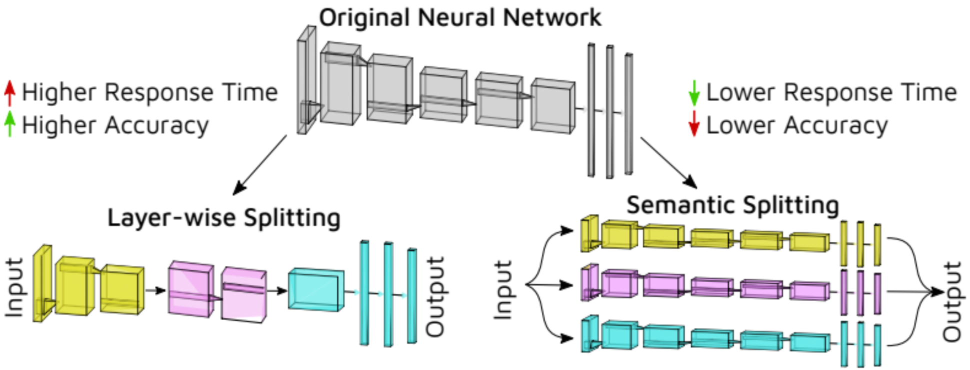
- This may require online DNN split and placement to cope with actual performance

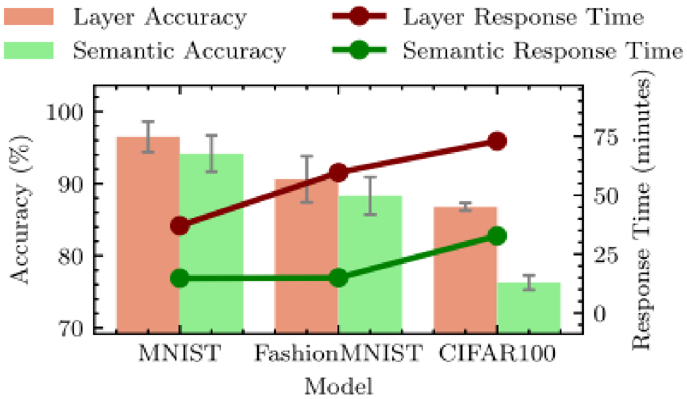
X. Tang *et al*. Joint Multiuser DNN Partitioning and Computational Resource Allocation for Collaborative Edge Intelligence, IEEE IoT 2021.
H. Liang *et al*. DNN Surgery: Accelerating DNN Inference on the Edge Through Layer Partitioning, IEEE TCC 2023.

How to best parallelize DNN execution?



- SplitNets learning
  - Semantically disparate classes require largely disjoint sets of features
- Semantic splits vs layer-wise splits
  - Lower accuracy, but parallelizable
  - Requires re-training



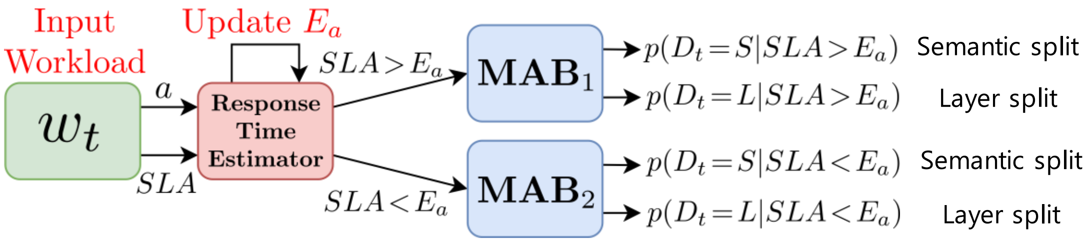


- How to choose at runtime the best DNN split topology taking into account performance?

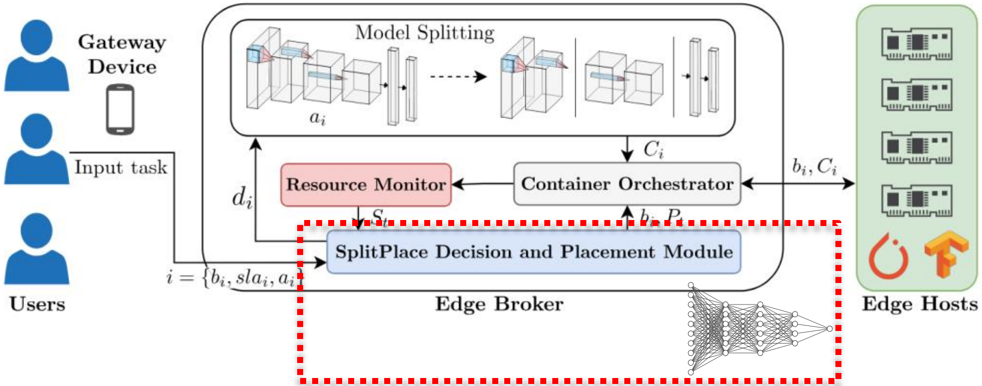
J. Kim, Y. Park, G. Kim, and S. J. Hwang, "SplitNet: Learning to semantically split deep networks for parameter reduction and model parallelization," ICML, 2017.
Tuli, S., Casale, G., & Jennings, N. R. Splitplace: AI augmented splitting and placement of large-scale neural networks in mobile edge environments. *TMC, 2013.*

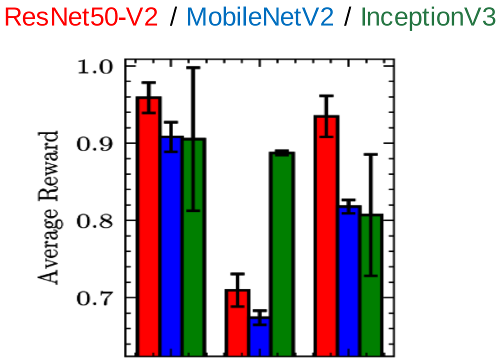
SplitPlace: Dynamic DNN splitting

- Contextual Multi-Armed Bandit (MAB) decides split-type
  - Reward proportional to SLA compliance and accuracy $O \propto \mathbb{1}(R_i \leq SLA_i) + A_i$
  - Two contexts: High response time / Low response time w.r.t. SLA



- SplitPlace architecture with "digital twin" for placement





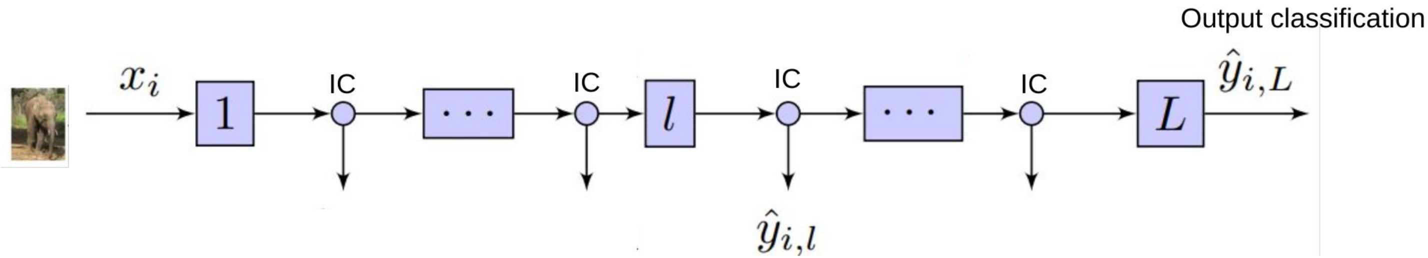
SplitPlace vs. Layer-wise vs. Pruning

Tuli, S., Casale, G., & Jennings, N. R.. Splitplace: AI augmented splitting and placement of large-scale neural networks in mobile edge environments. *IEEE TMC*.

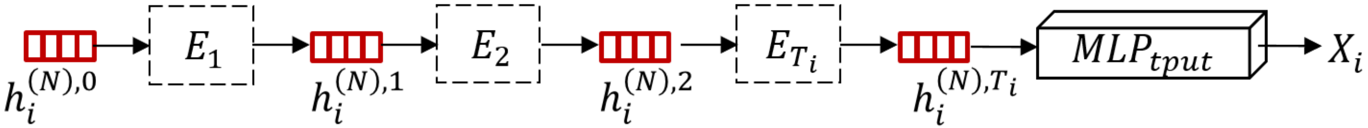
Conclusion

Summary

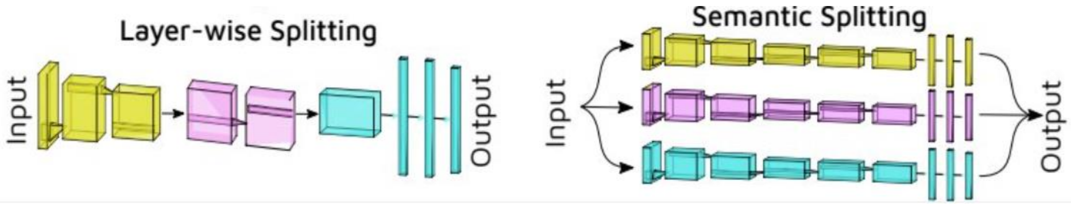
- We can recast early-exits as a mechanism to tune performance and reliability



- We can tailor GNNs to performance prediction tasks: https://github.com/imperial-qore/ChainNet



- Runtime DNN splitting and placement



Opportunities for research in ICPE

How shall software performance engineering evolve to support AI systems?

- The number of software products embedding DNNs keeps growing
  - Increasing collaboration of DevOps teams and Data science teams (eg MLOps)
  - The boundary between traditional services and DNN inference is fading!
- Edge AI splitting and placement has many commonalities with SPE
  - Many opportunities for cross-fertilization

Product lines	DNN variants (splits, compression, …)
Search-based software engineering	Network Architecture Search
Orchestration	DNN partitioning & placement
…	…