

Apparate

Rethinking Early Exits to Tame Latency-Throughput Tensions in ML Serving

Yinwei Dai*, Rui Pan*, Anand Iyer, Kai Li, Ravi Netravali



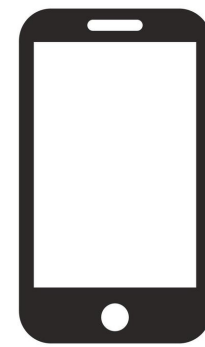
PRINCETON
UNIVERSITY



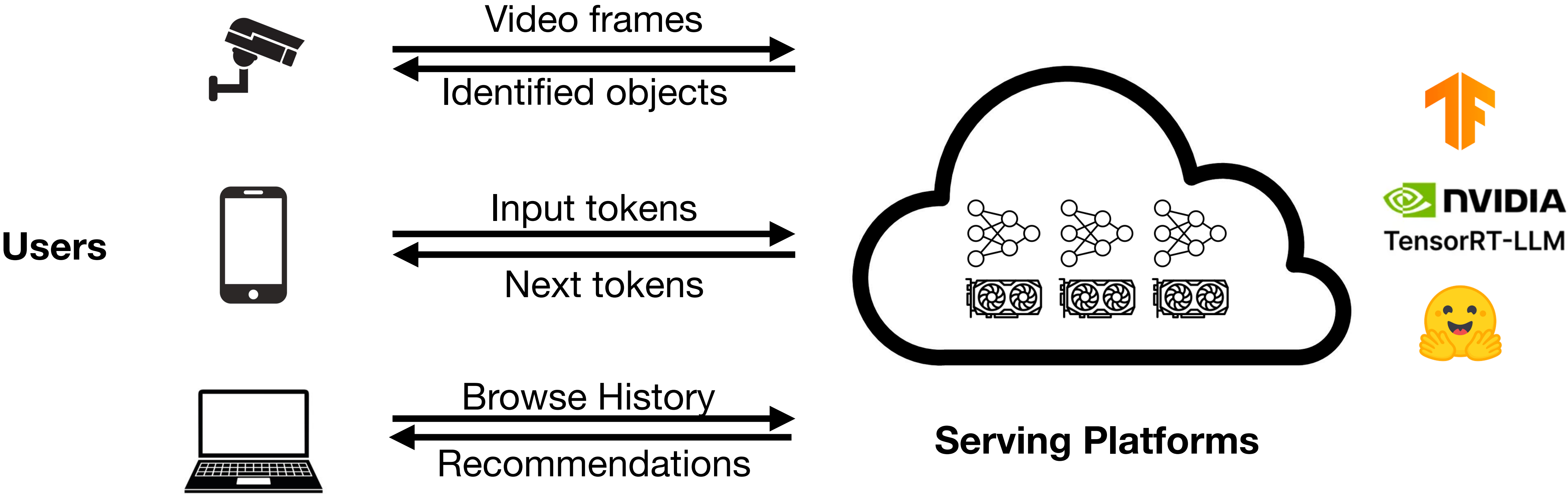
ML-Based Interactive Applications Are Pervasive



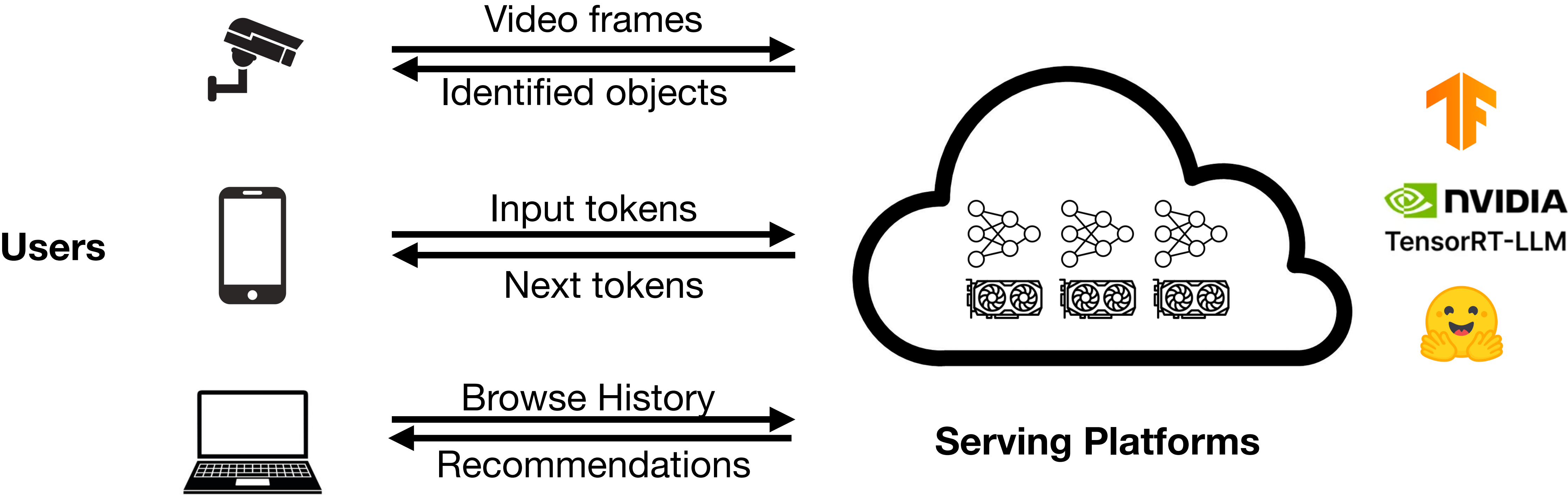
Users



ML-Based Interactive Applications Are Pervasive



ML-Based Interactive Applications Are Pervasive



Goal: maximize **throughput** and meet **latency SLOs**

Serving Platforms Use Batching as a Knob

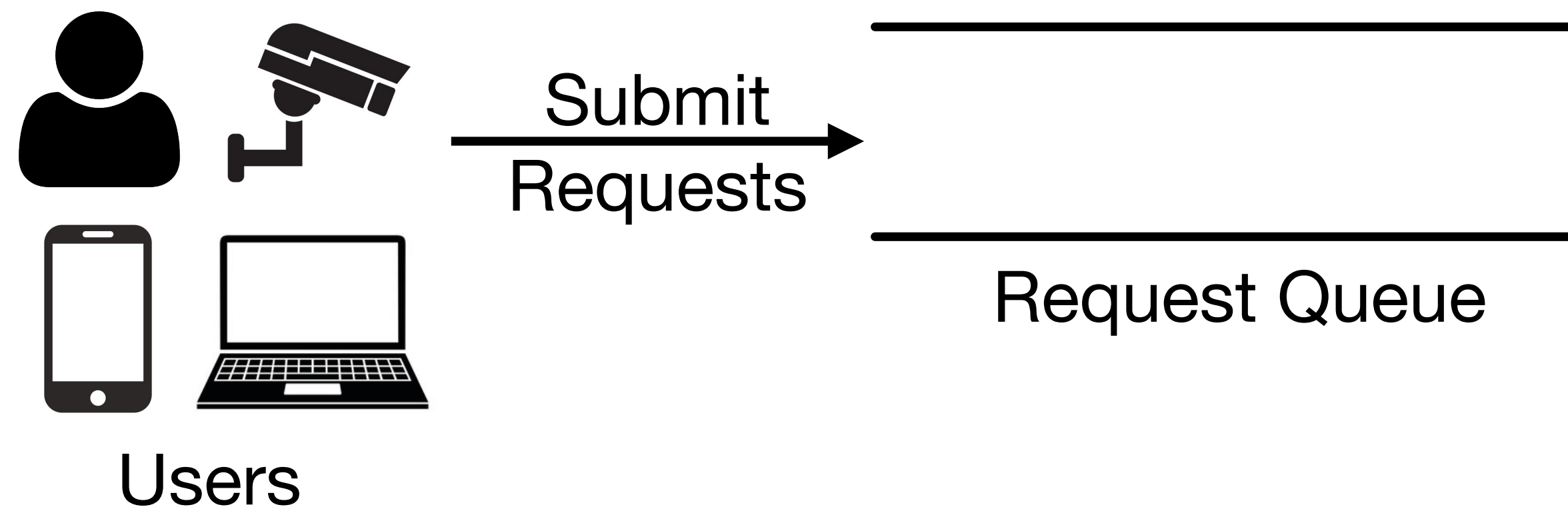
to maximize throughput and minimize SLO violations



Users

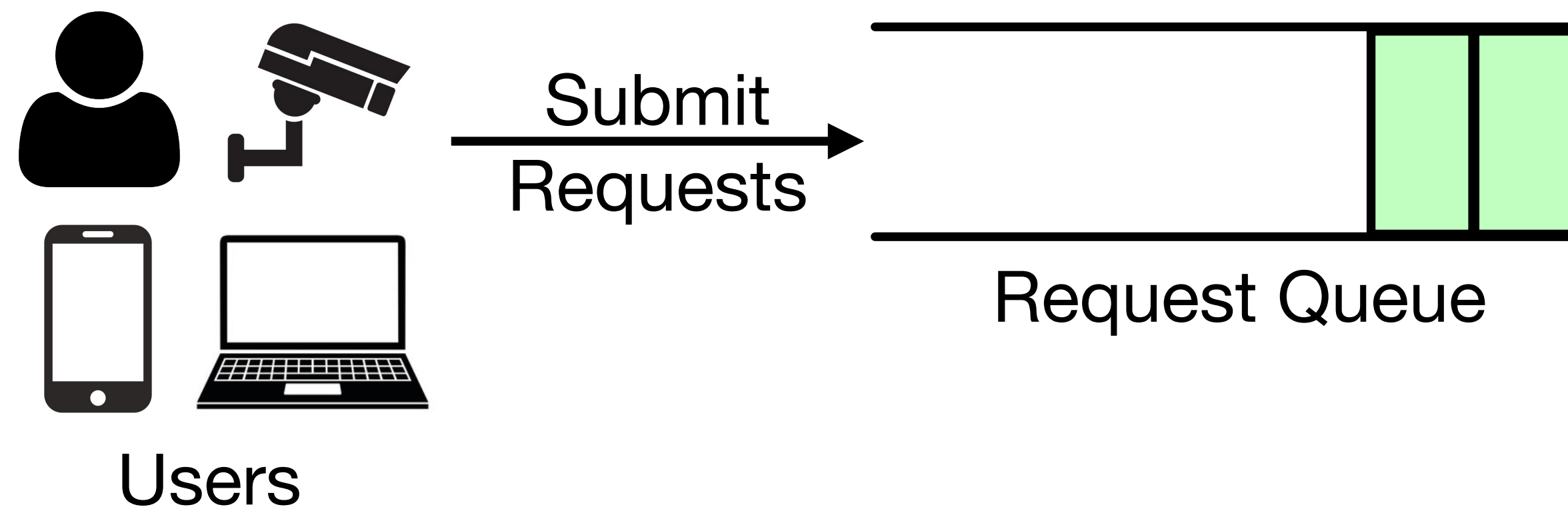
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



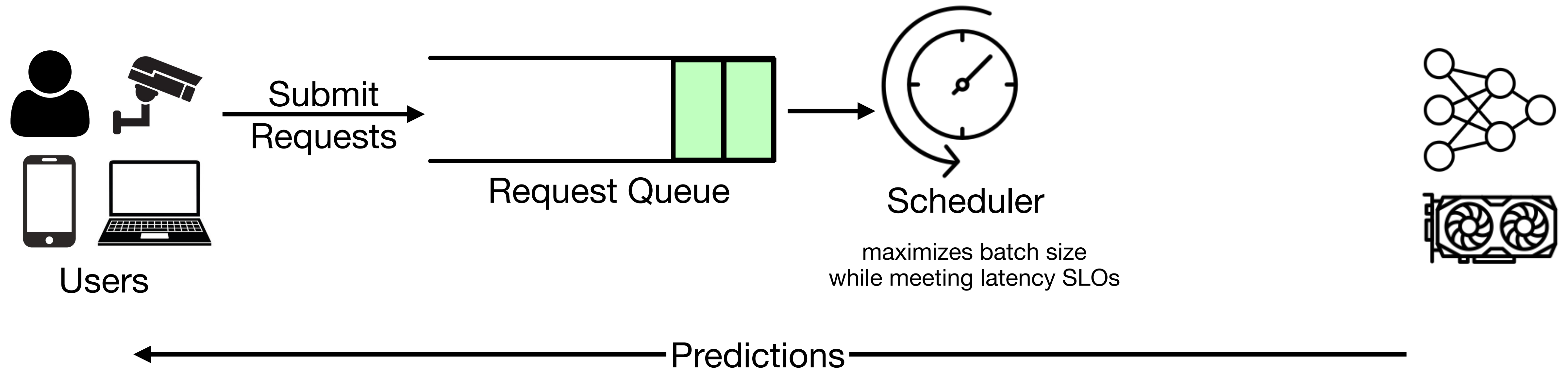
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



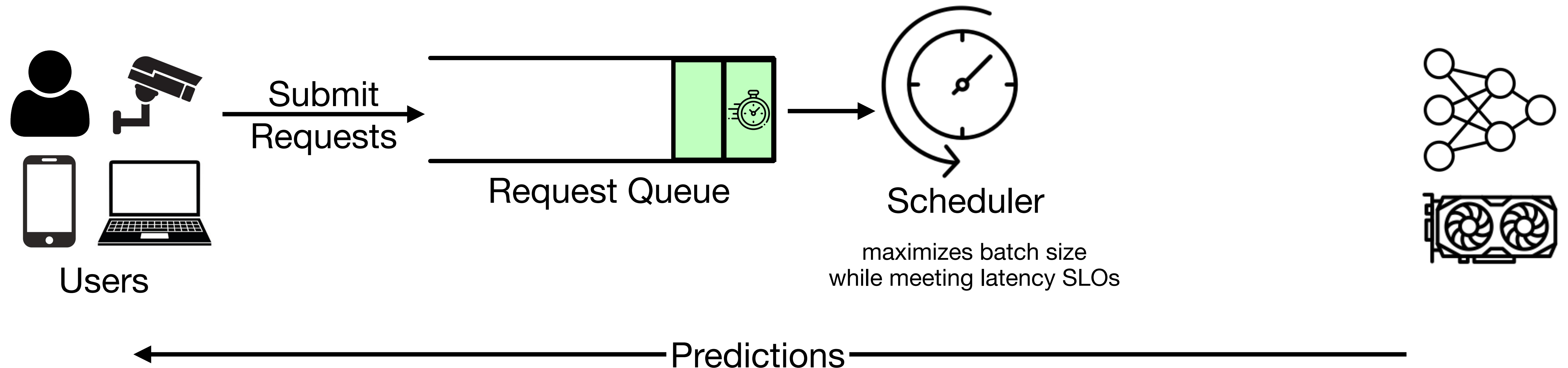
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



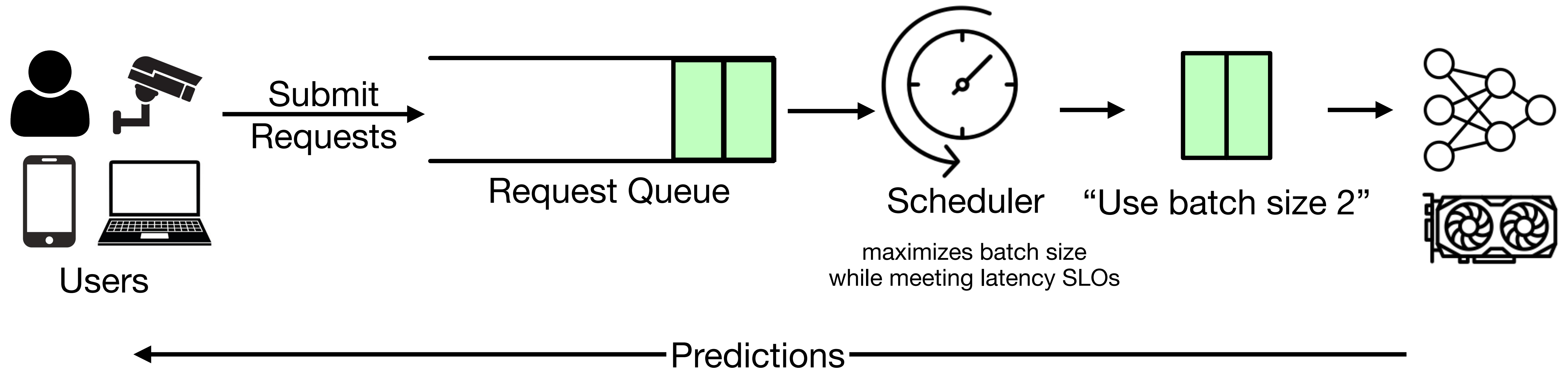
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



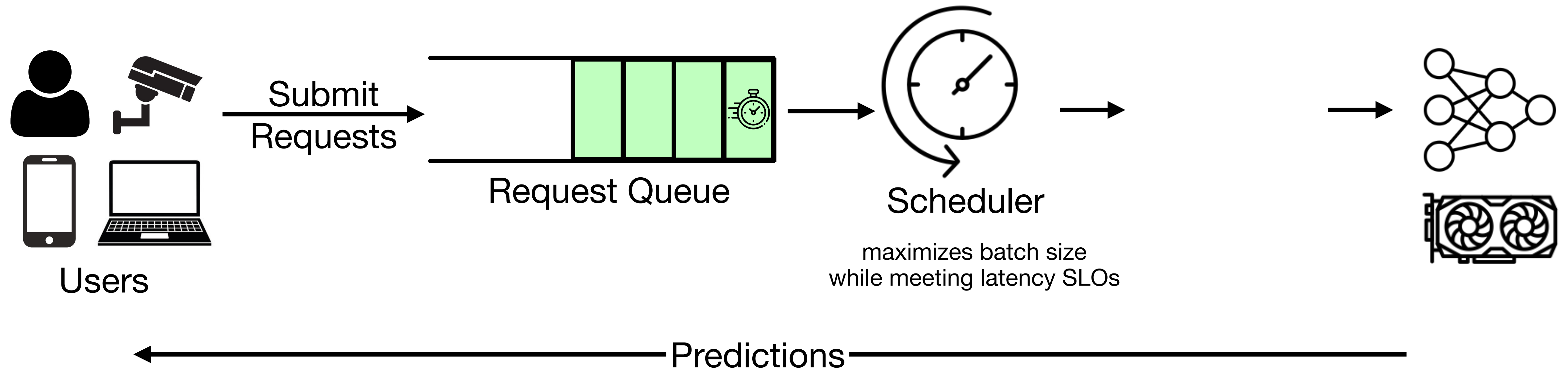
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



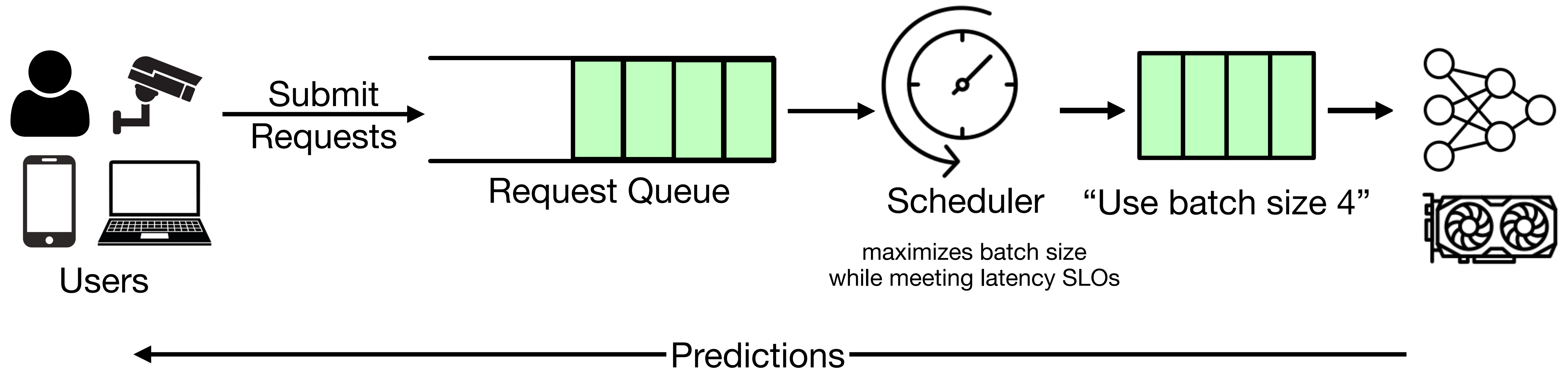
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



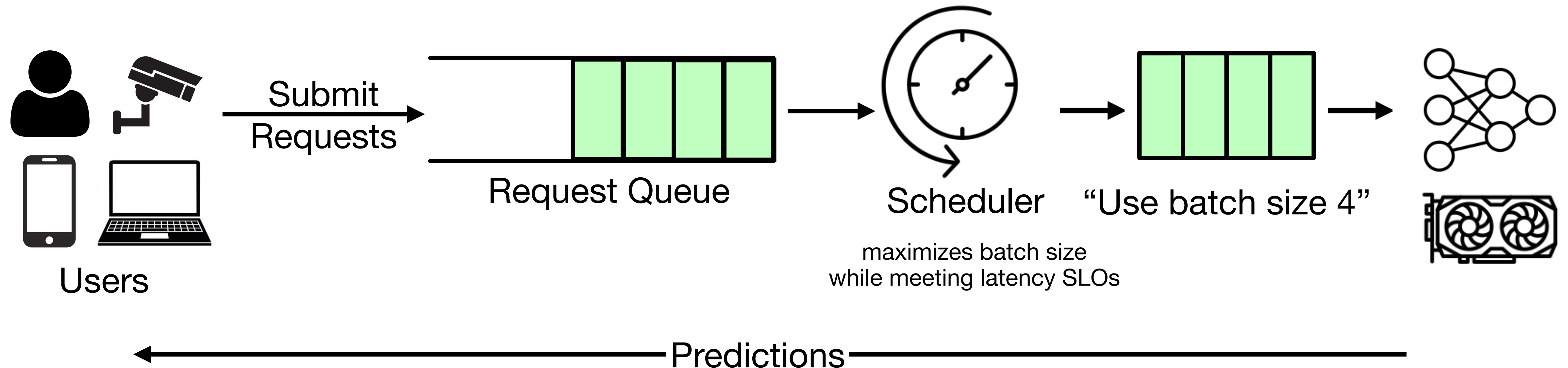
Serving Platforms Use Batching as a Knob

to maximize throughput and minimize SLO violations



Serving Platforms Use Batching as a Knob

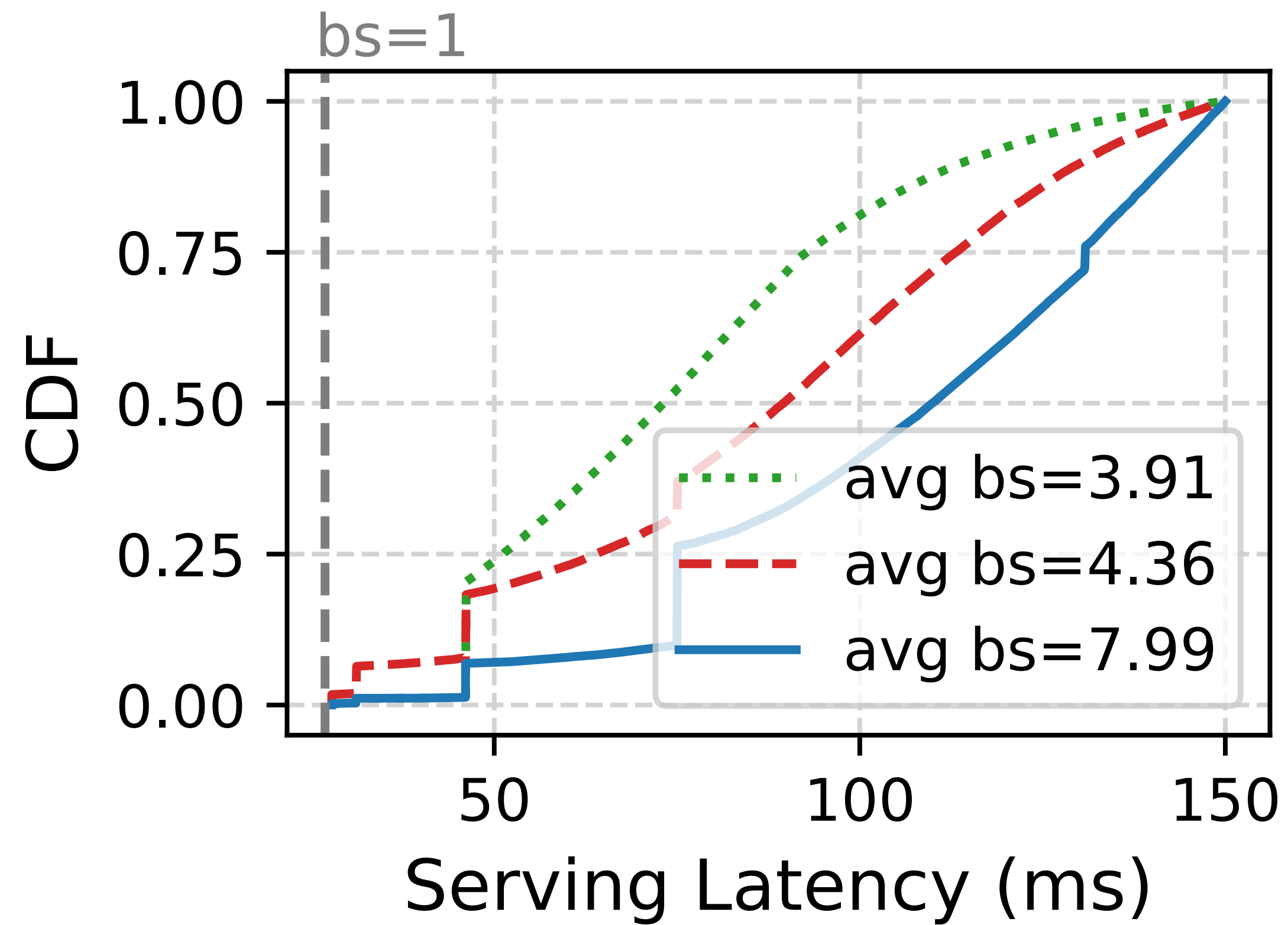
to maximize throughput and minimize SLO violations



Existing systems view **latency** utility as binary: **meeting SLOs or not**

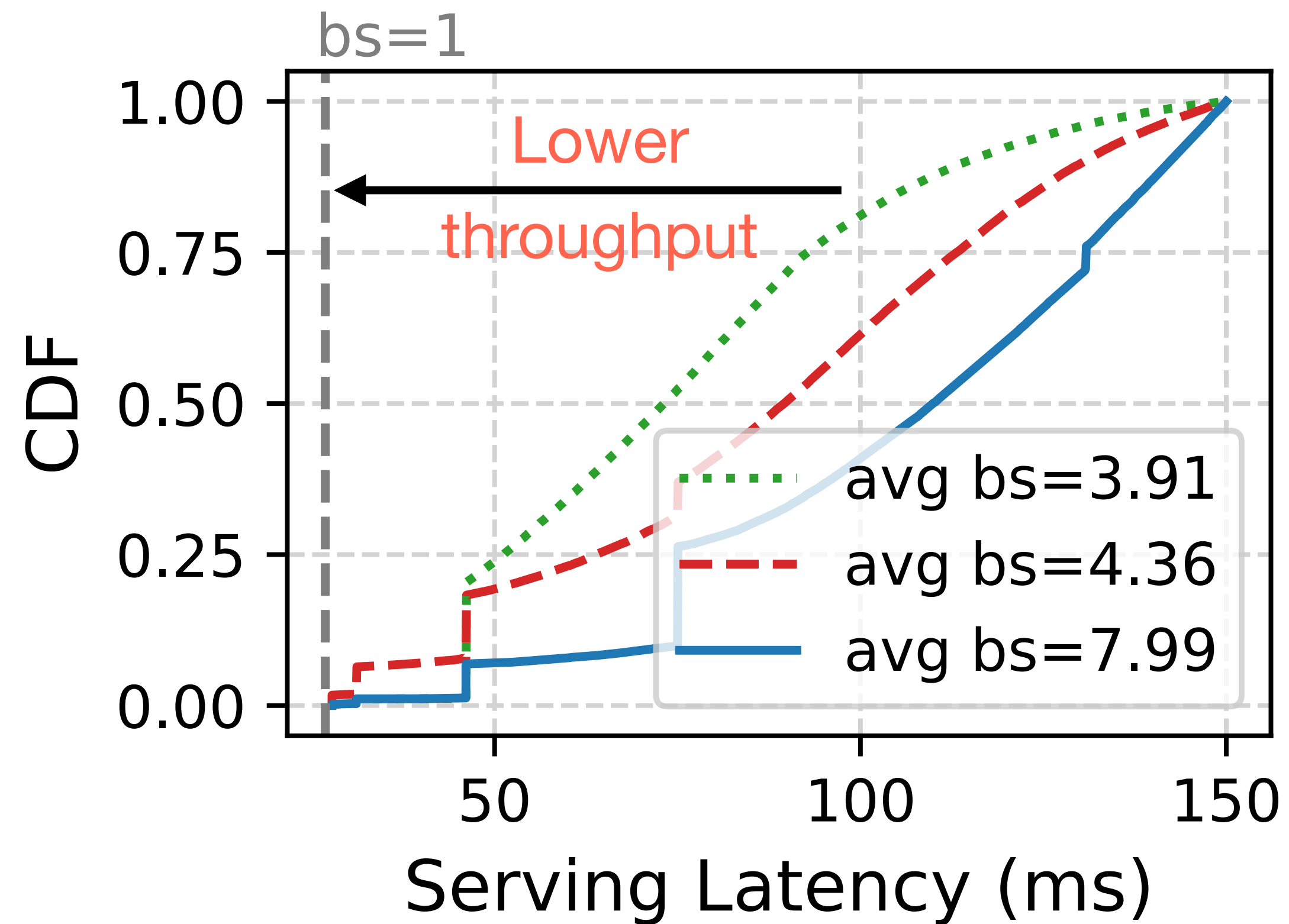
Batch Size Tuning

Presents harsh latency-throughput tradeoffs



Batch Size Tuning

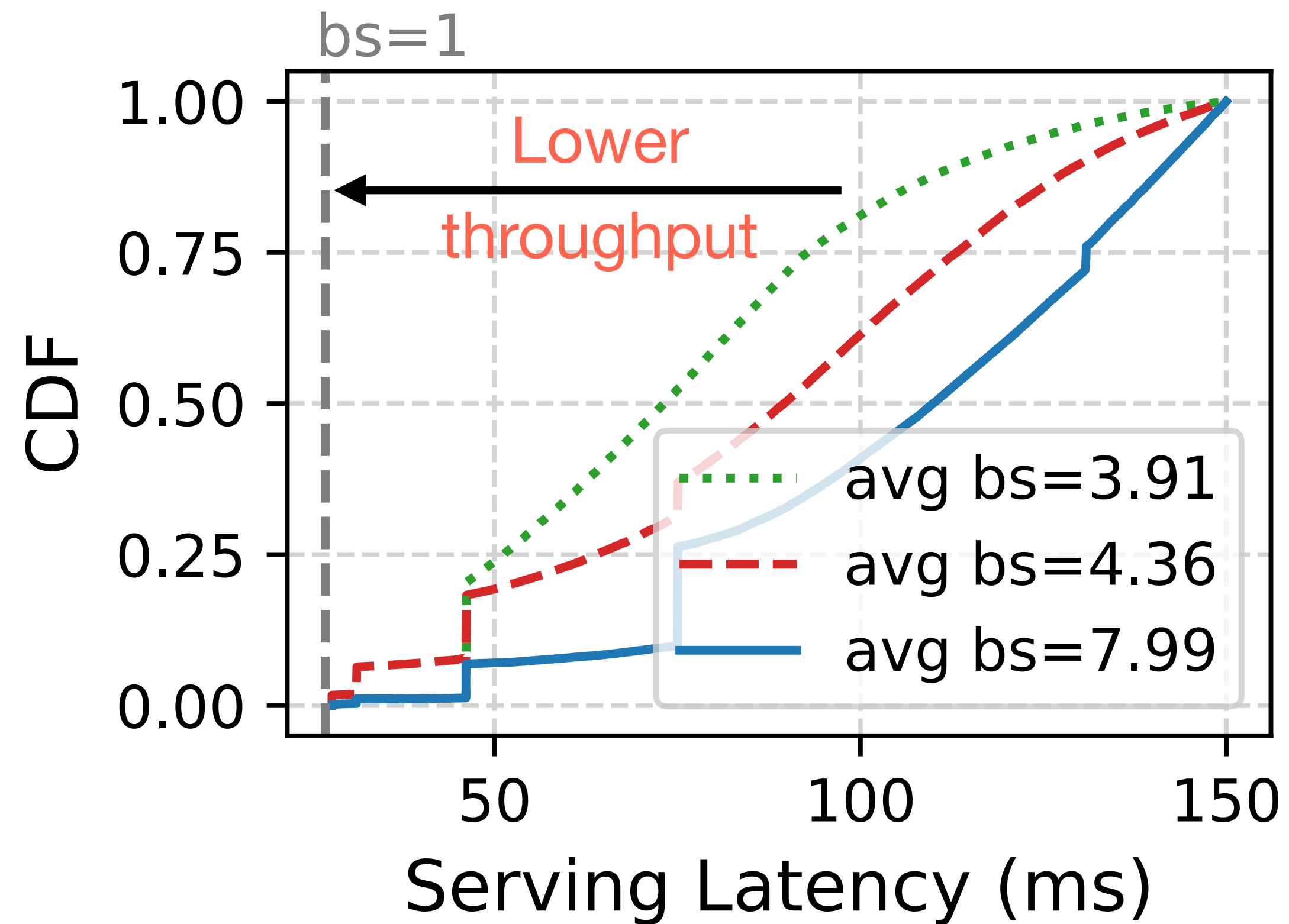
Presents harsh latency-throughput tradeoffs



Median latency improvements (17–39%)
cause up to 3.6× throughput reductions

Batch Size Tuning

Presents harsh latency-throughput tradeoffs

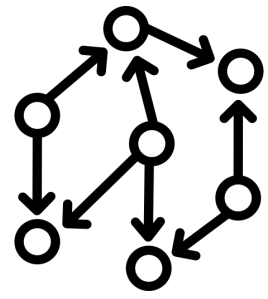


Median latency improvements (17–39%) cause up to 3.6× throughput reductions

Batch size is too coarse-grained to trade off per-input latency

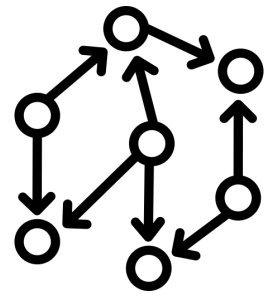
Opportunity: Per-Input Compute

Opportunity: Per-Input Compute

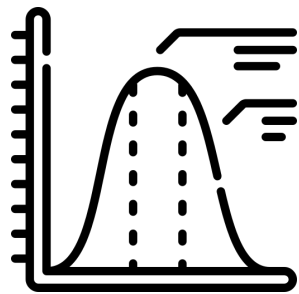


Models are over-parametrized

Opportunity: Per-Input Compute



Models are over-parametrized



Input easiness varies

Opportunity: Per-Input Compute



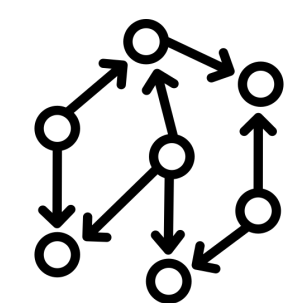
Models are over-parametrized



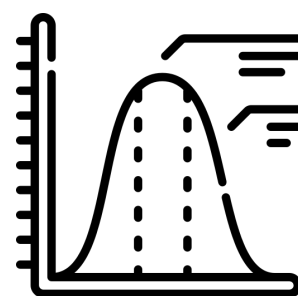
Input easiness varies

Sentiment Classification	Image Classification

Opportunity: Per-Input Compute




Models are over-parametrized

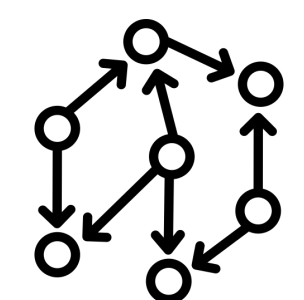


Input easiness varies

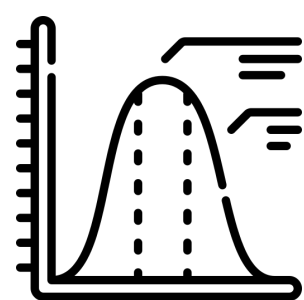


Sentiment Classification	Image Classification
I love this movie!	

Opportunity: Per-Input Compute



Models are over-parametrized



Input easiness varies



Sentiment Classification

I love this movie!

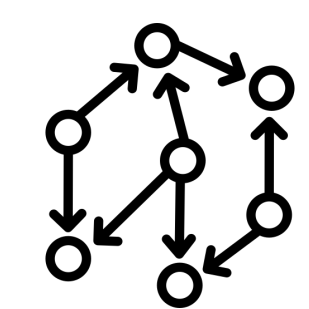


The plot was decent, but the acting felt lackluster.

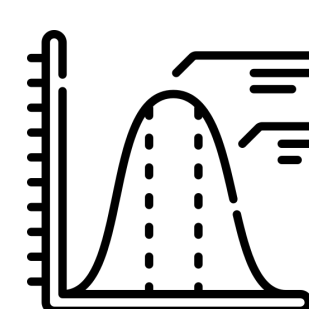
Image Classification



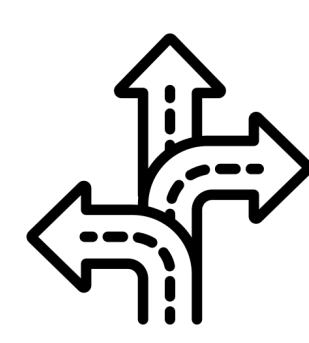
Opportunity: Per-Input Compute



Models are over-parametrized





Input easiness varies



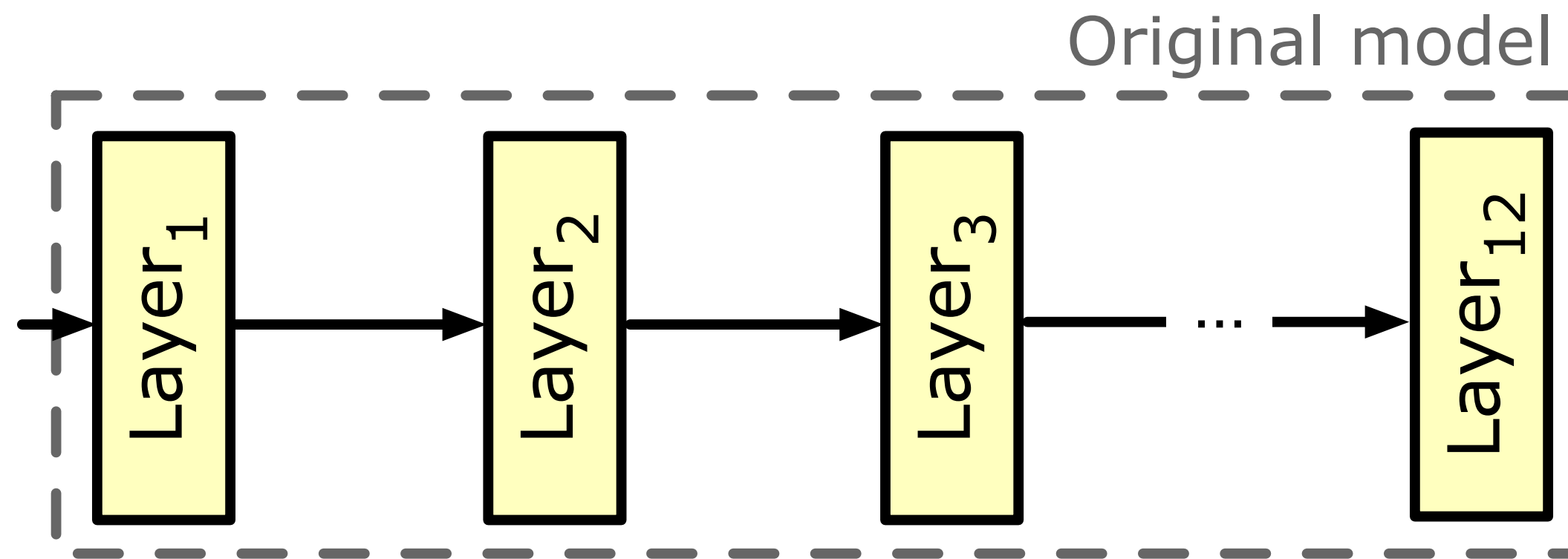
Adapt per-input compute!



Sentiment Classification	Image Classification
I love this movie!	
The plot was decent, but the acting felt lackluster.	

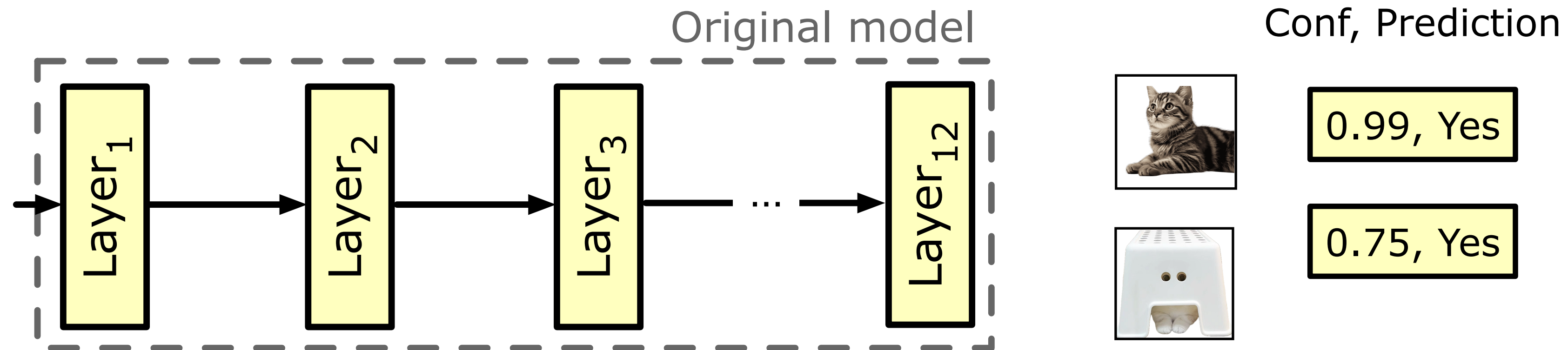
Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early



Early Exits Adapt Per-Input Compute

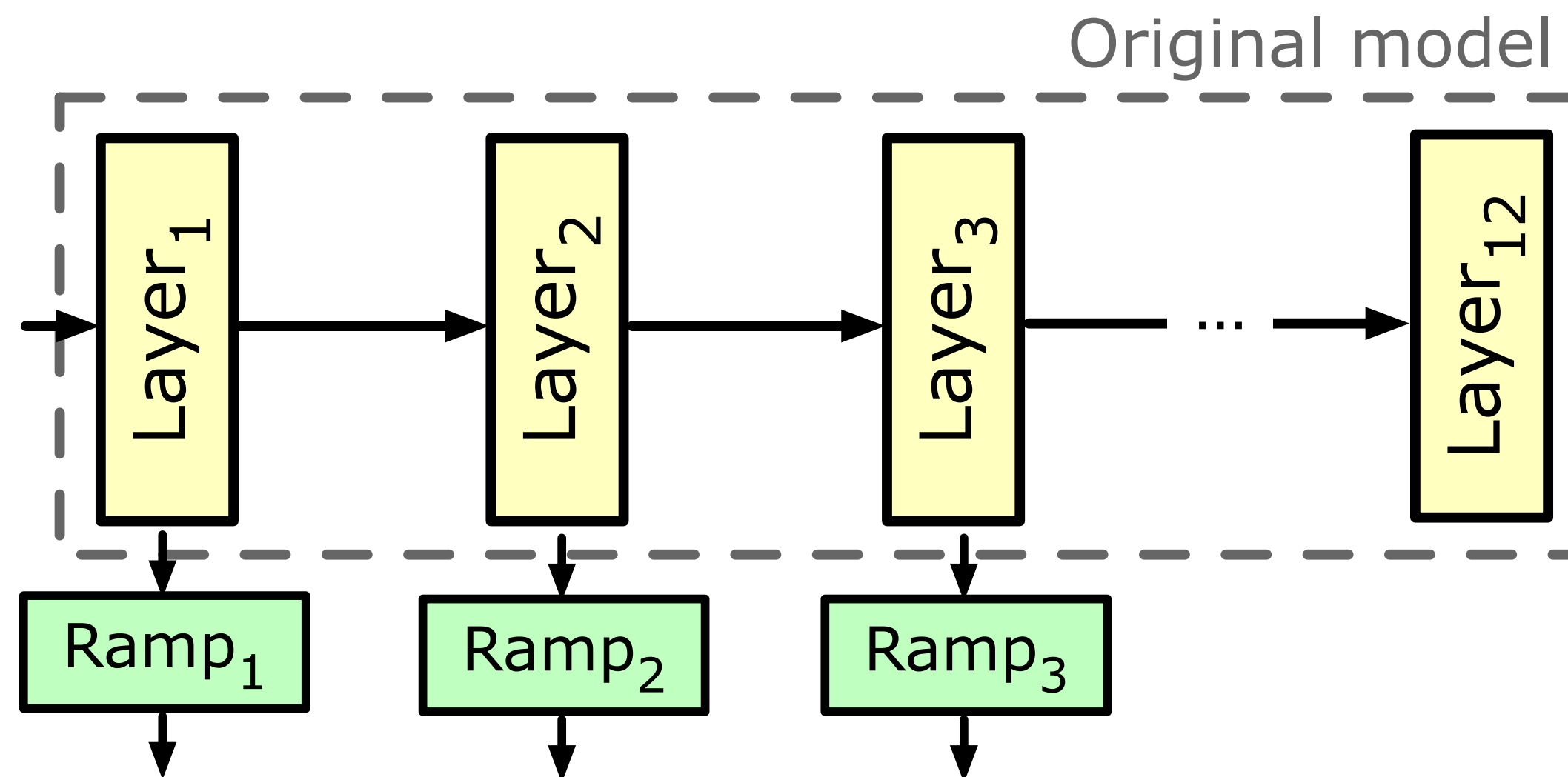
by allowing “easy” inputs to exit early



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

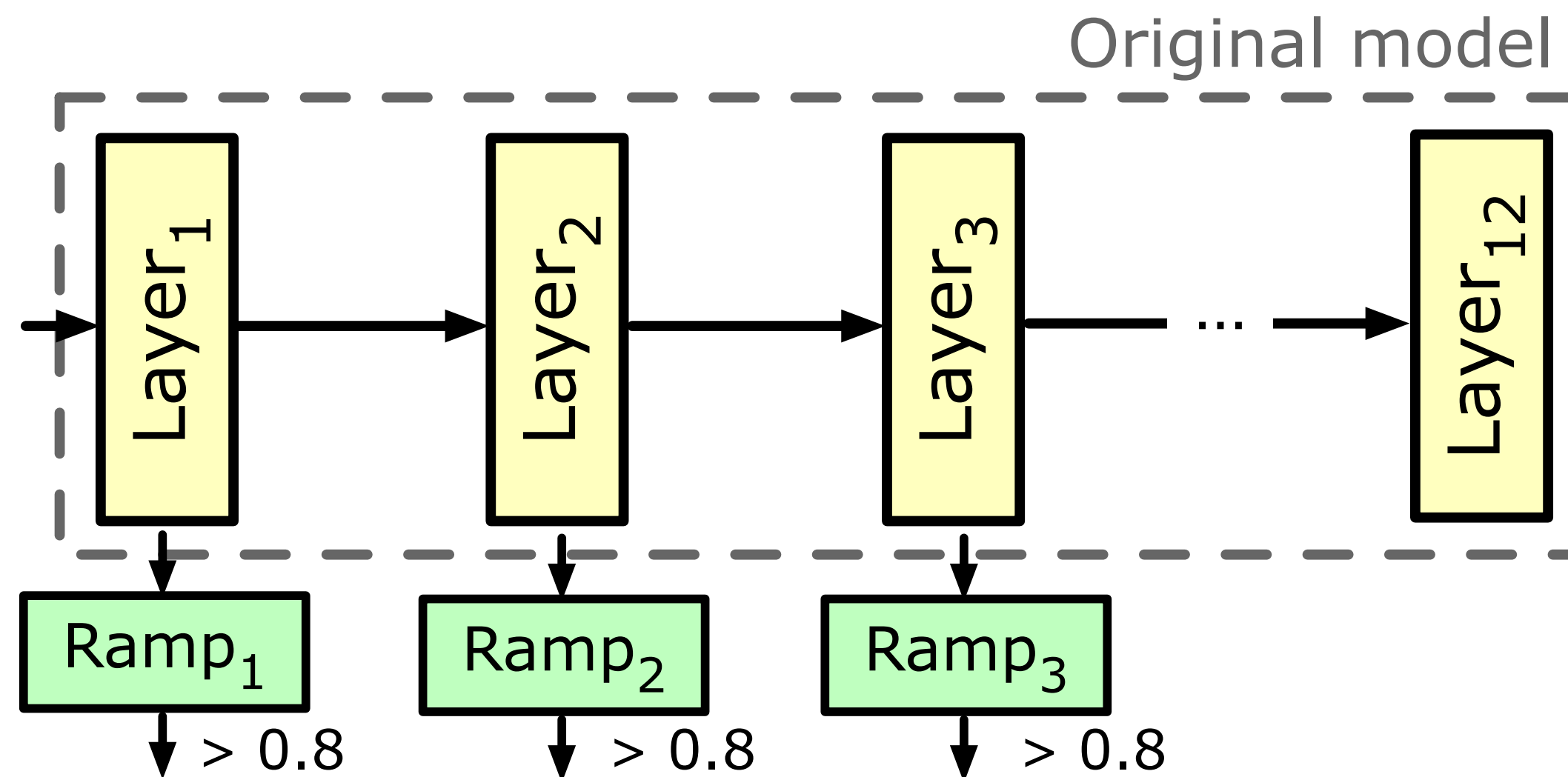
- Augment intermediate layers with **ramps** (intermediate classifiers)



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

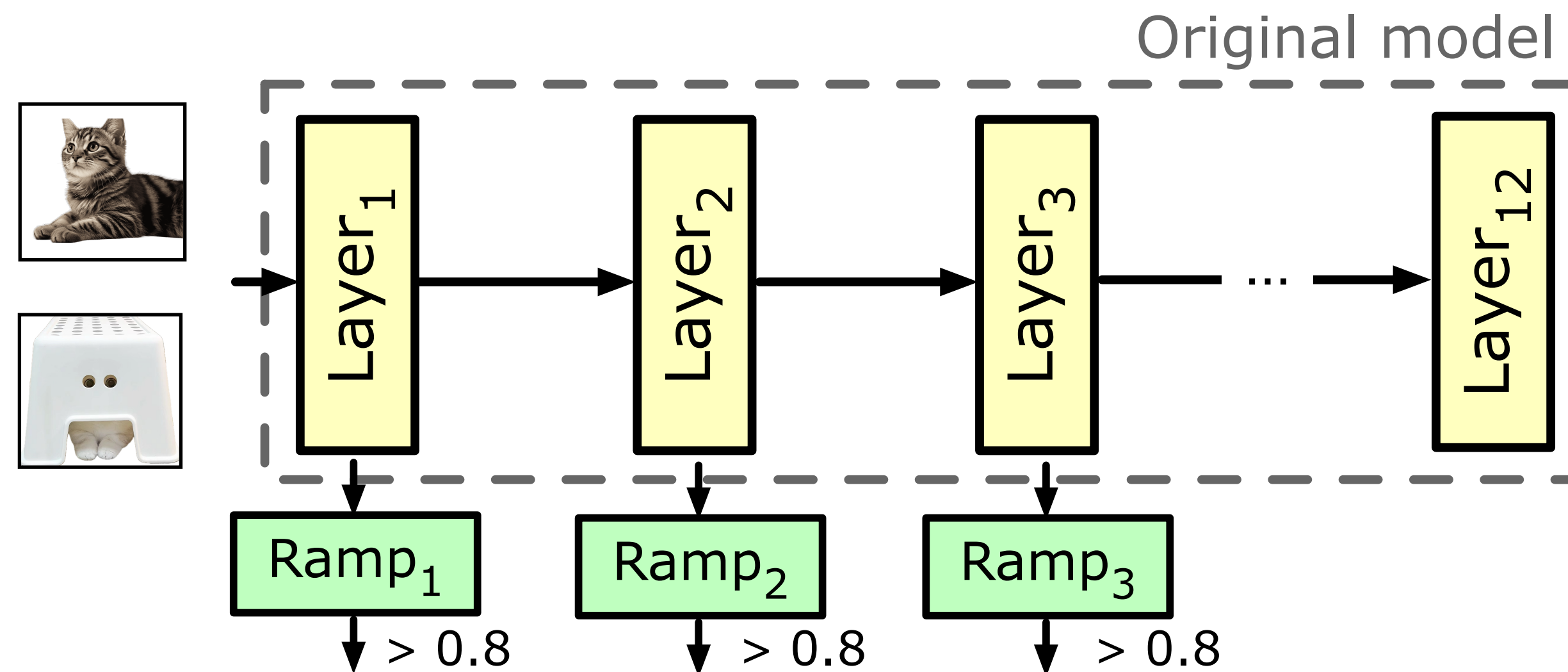
- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

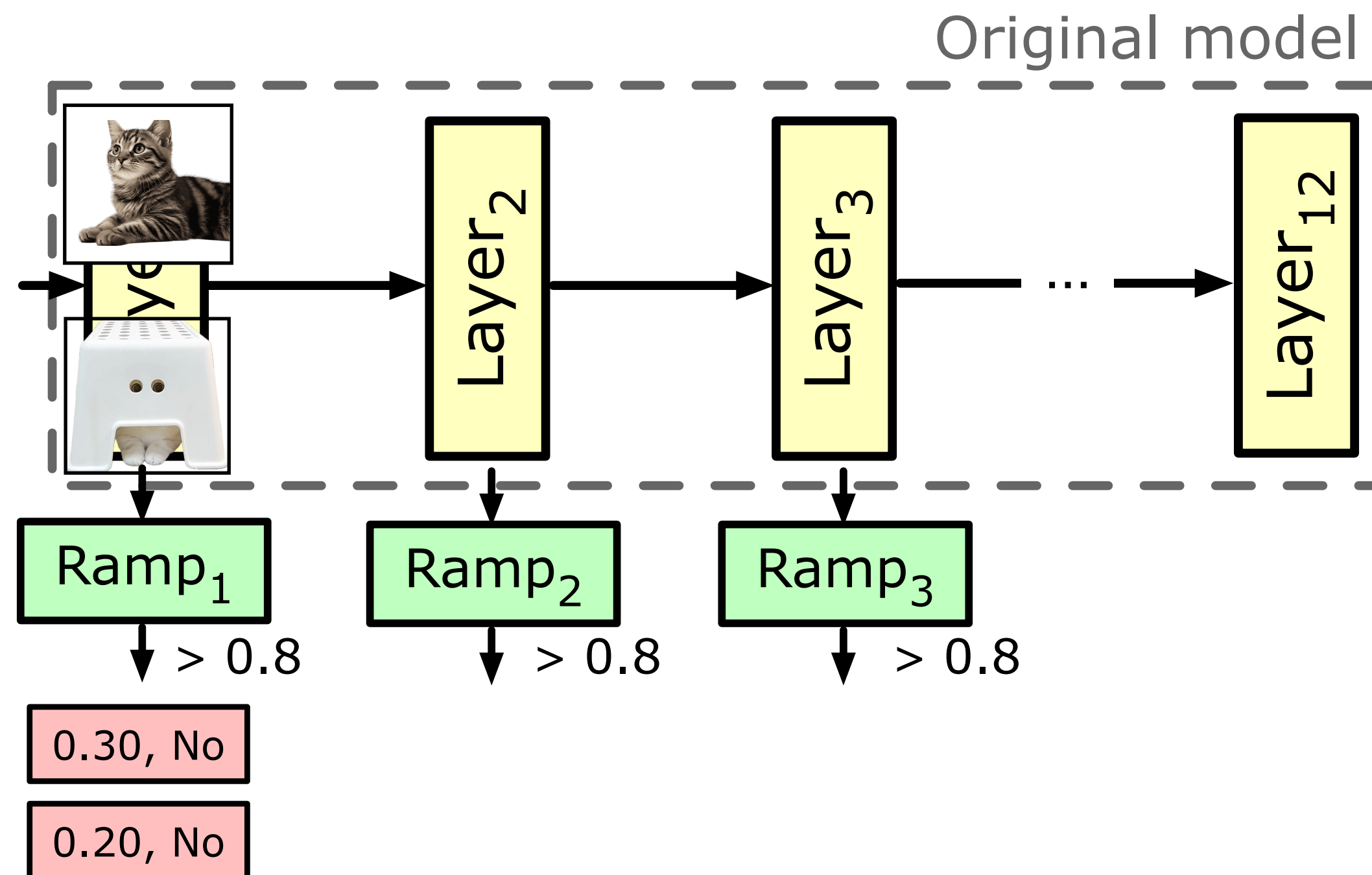
- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

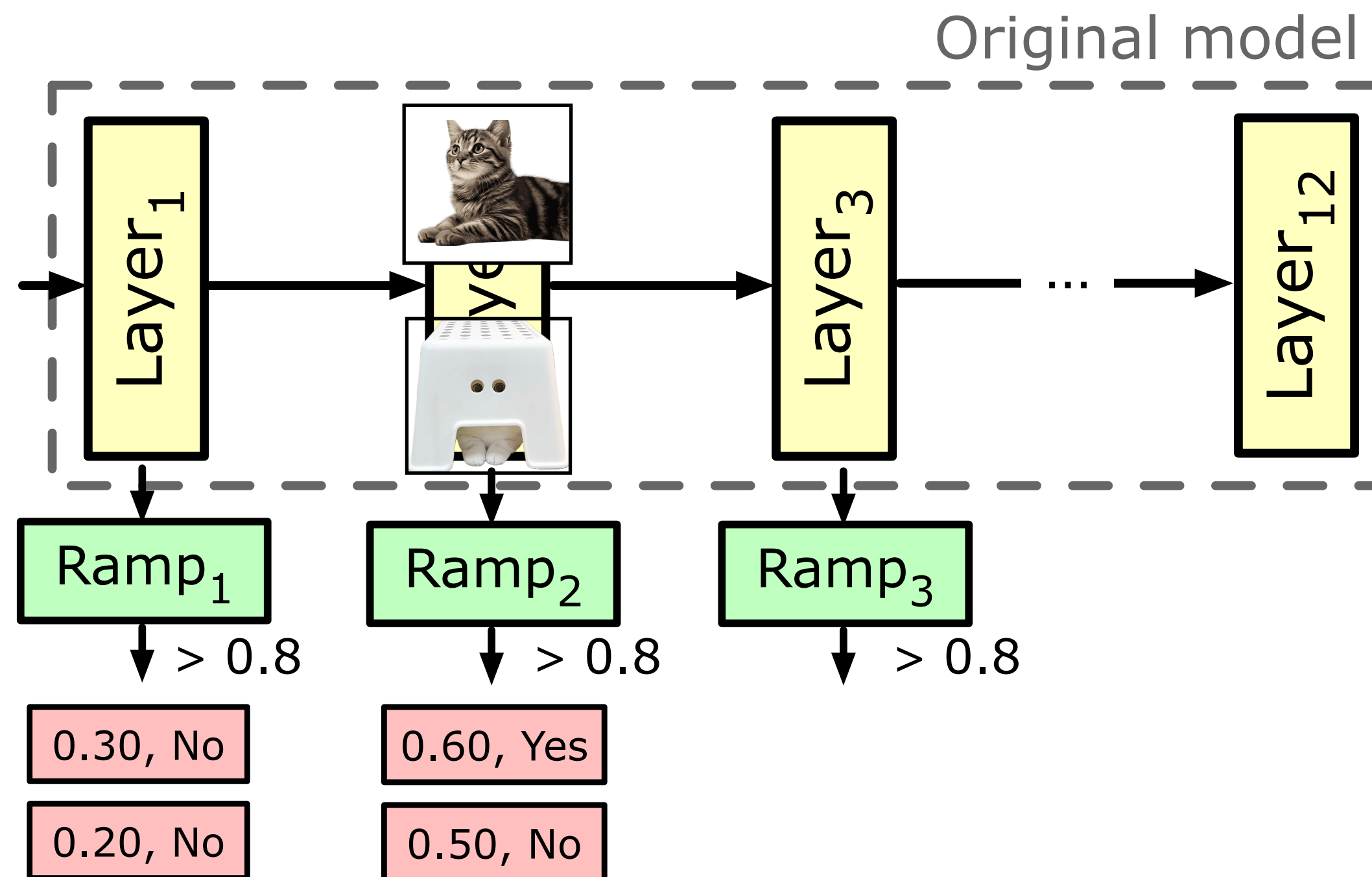
- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

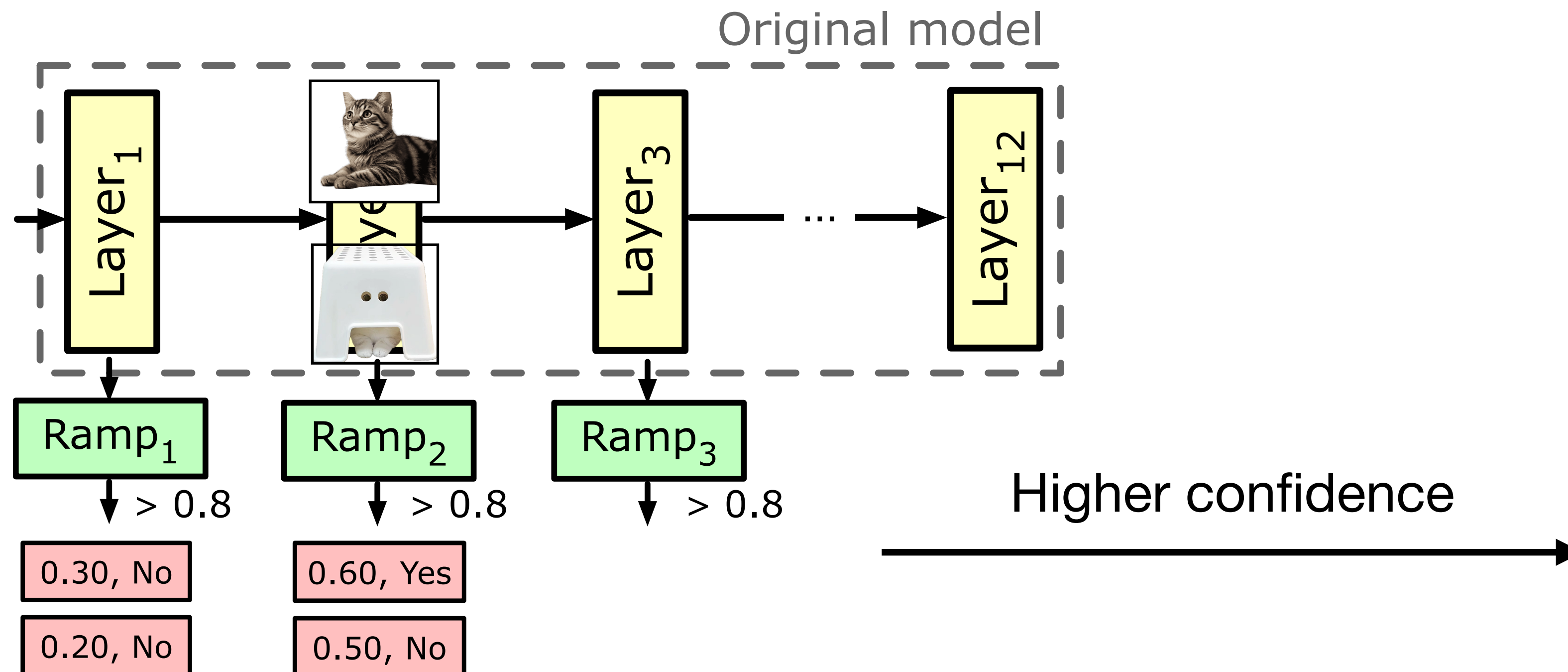
- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

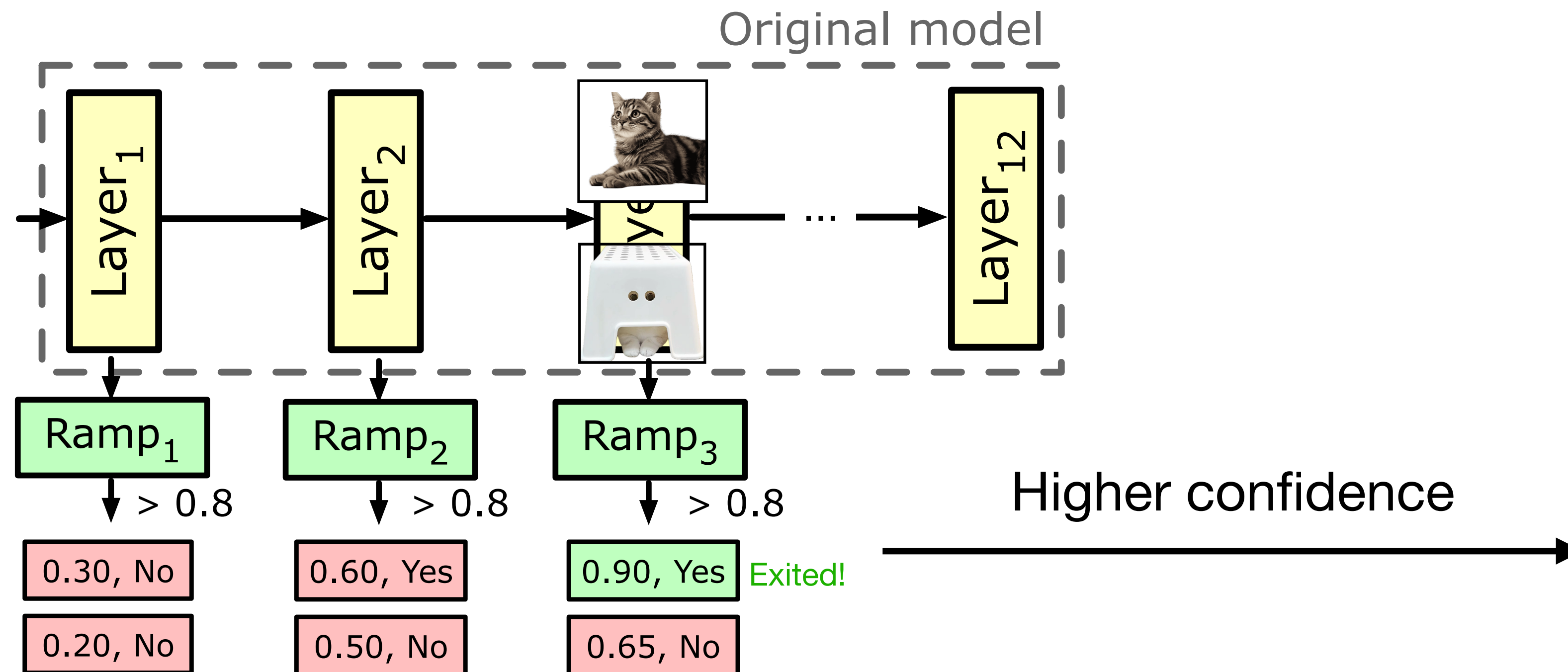
- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

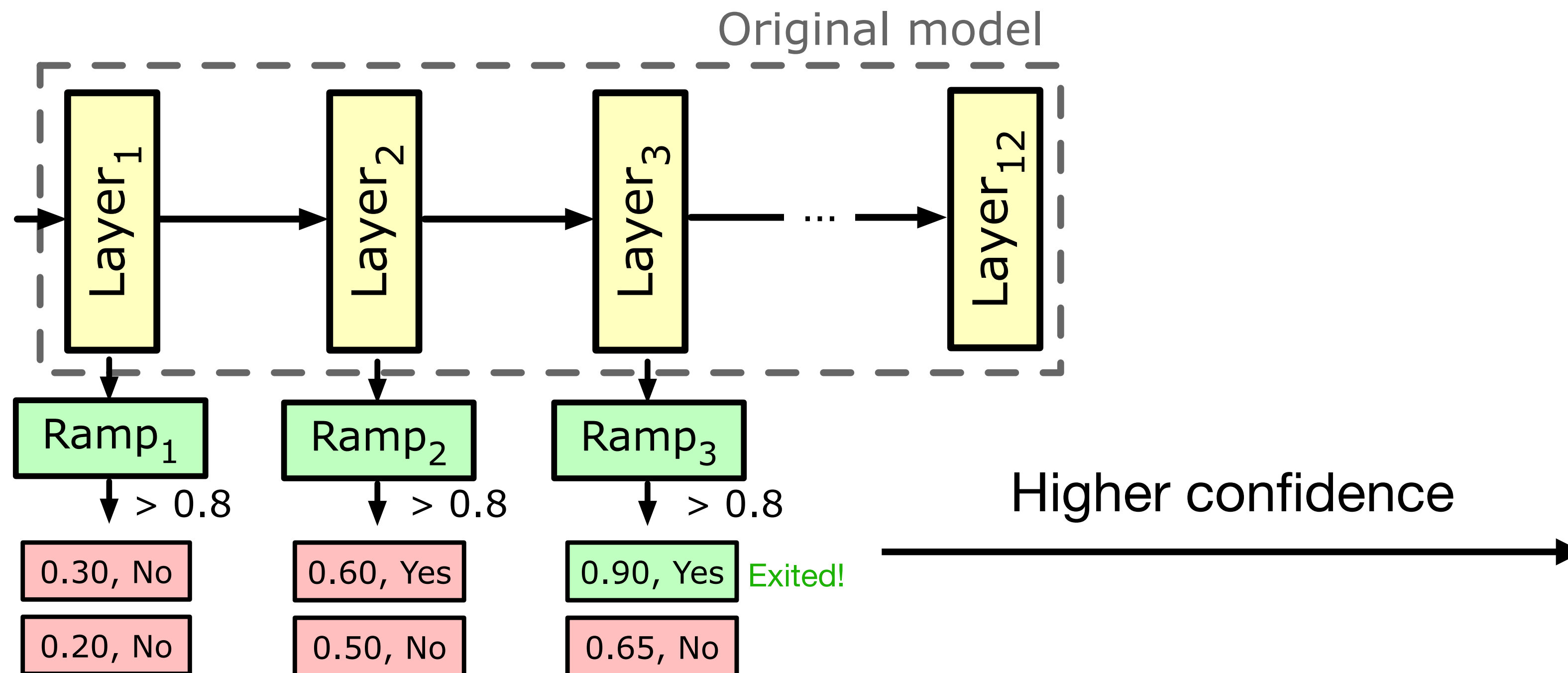
- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

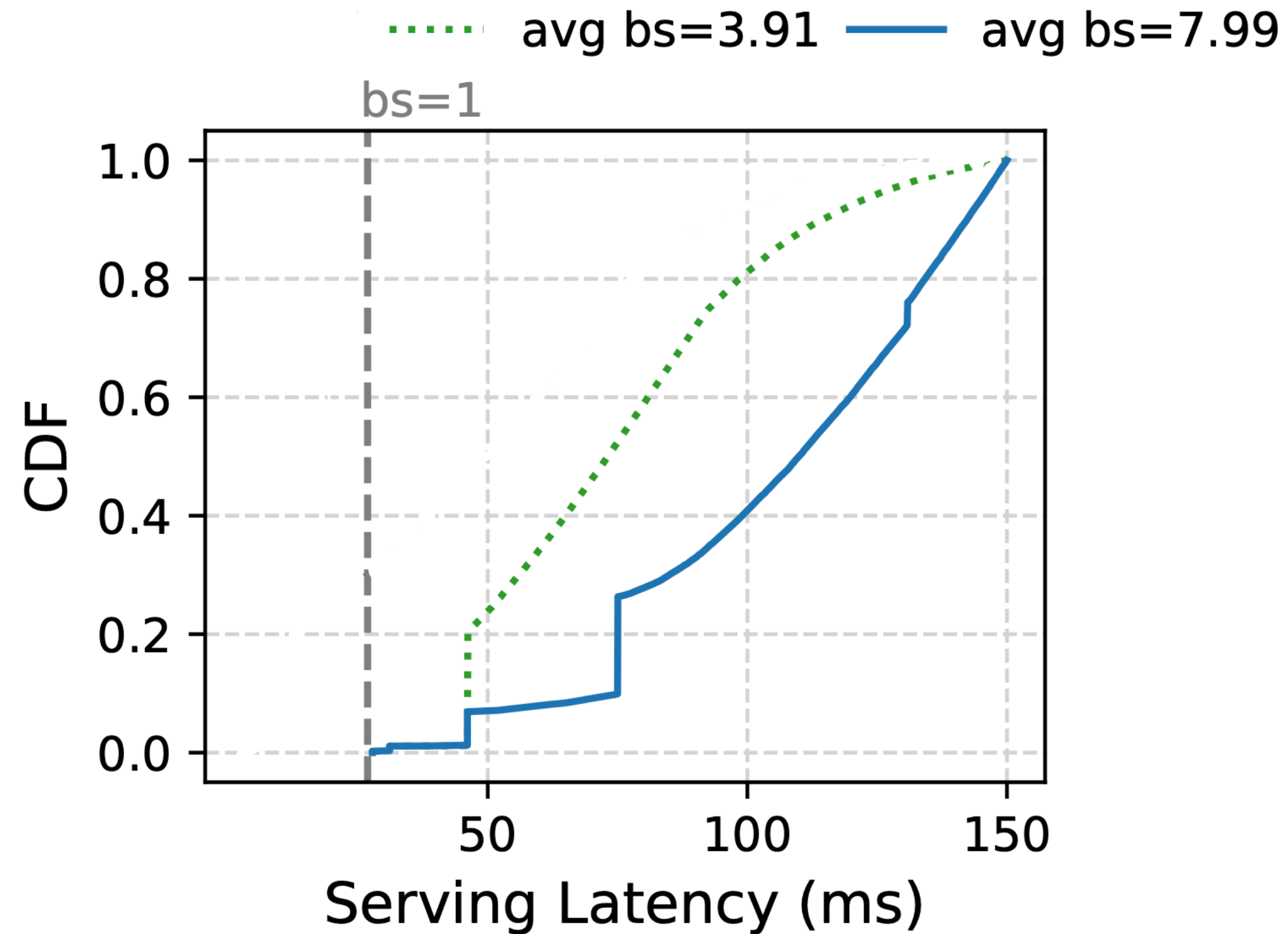
by allowing “easy” inputs to exit early

- Augment intermediate layers with **ramps** (intermediate classifiers)
- Exiting decisions: **confidence** > **threshold** (e.g., 0.8)



Early Exits Adapt Per-Input Compute

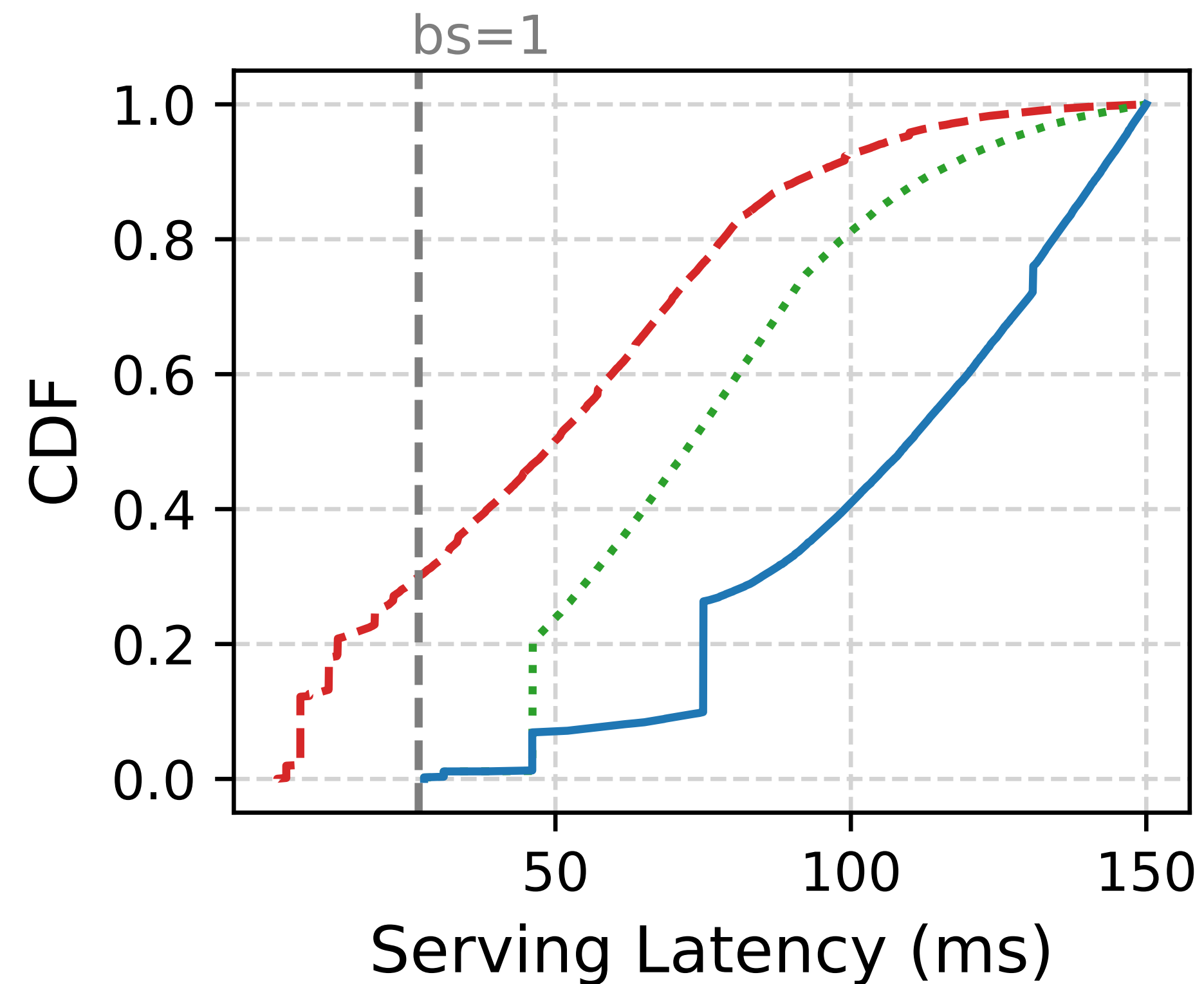
by allowing “easy” inputs to exit early



Early Exits Adapt Per-Input Compute

by allowing “easy” inputs to exit early

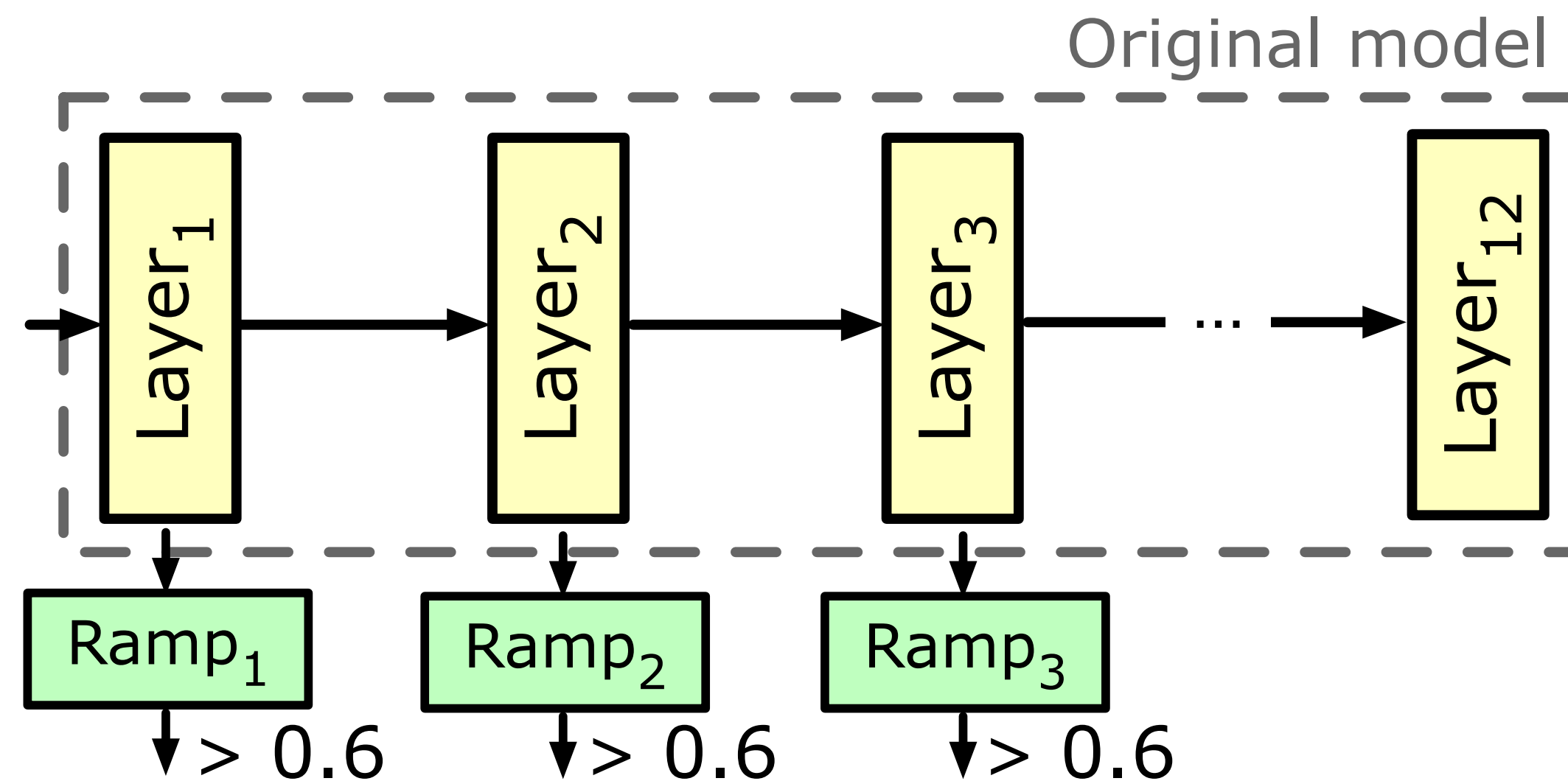
--- Optimal EE avg bs=3.91 — avg bs=7.99



Optimal Early Exits minimize
latency without sacrificing throughput

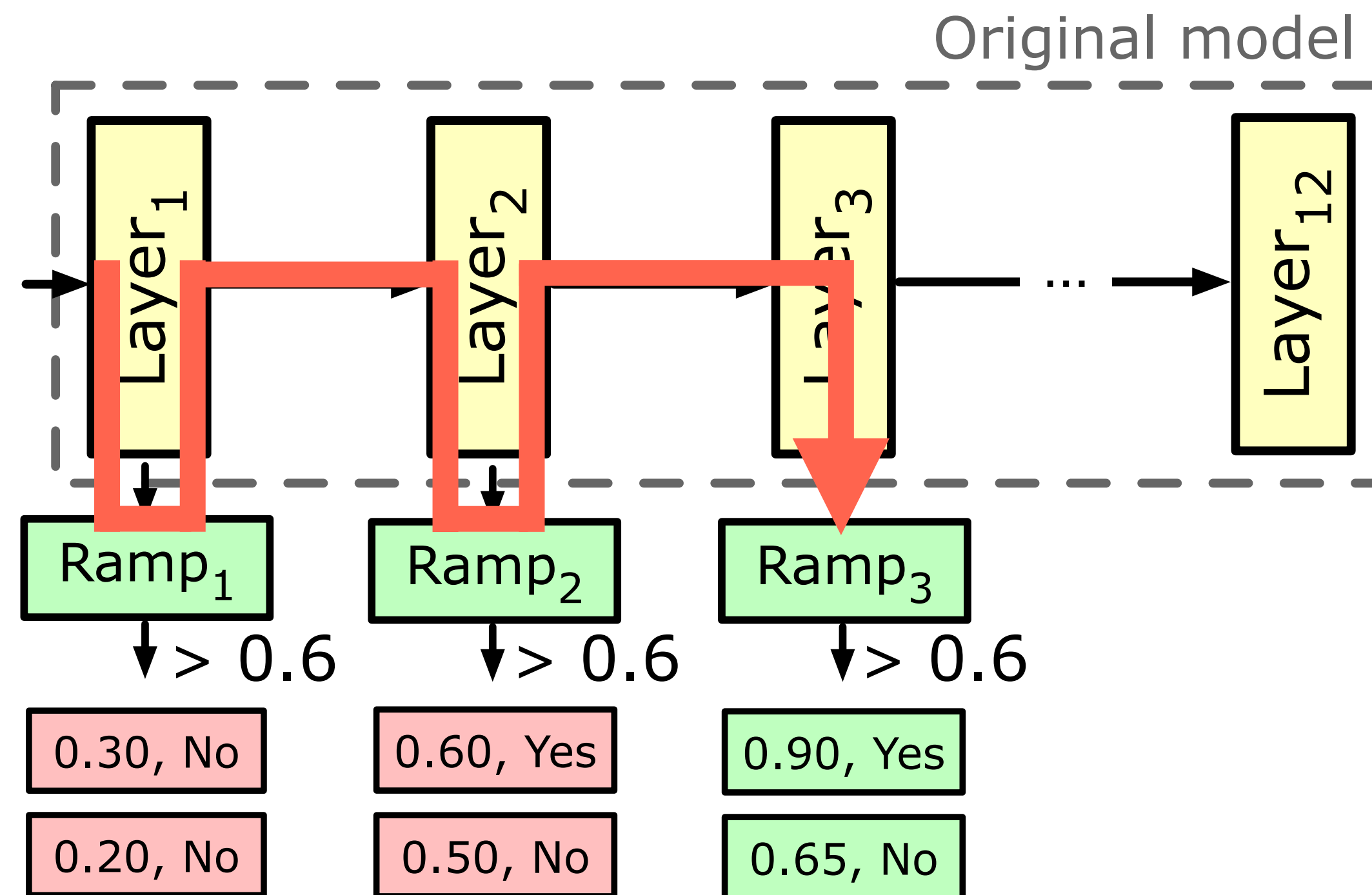
Early Exits: Practical Challenges

low accuracy and high overheads



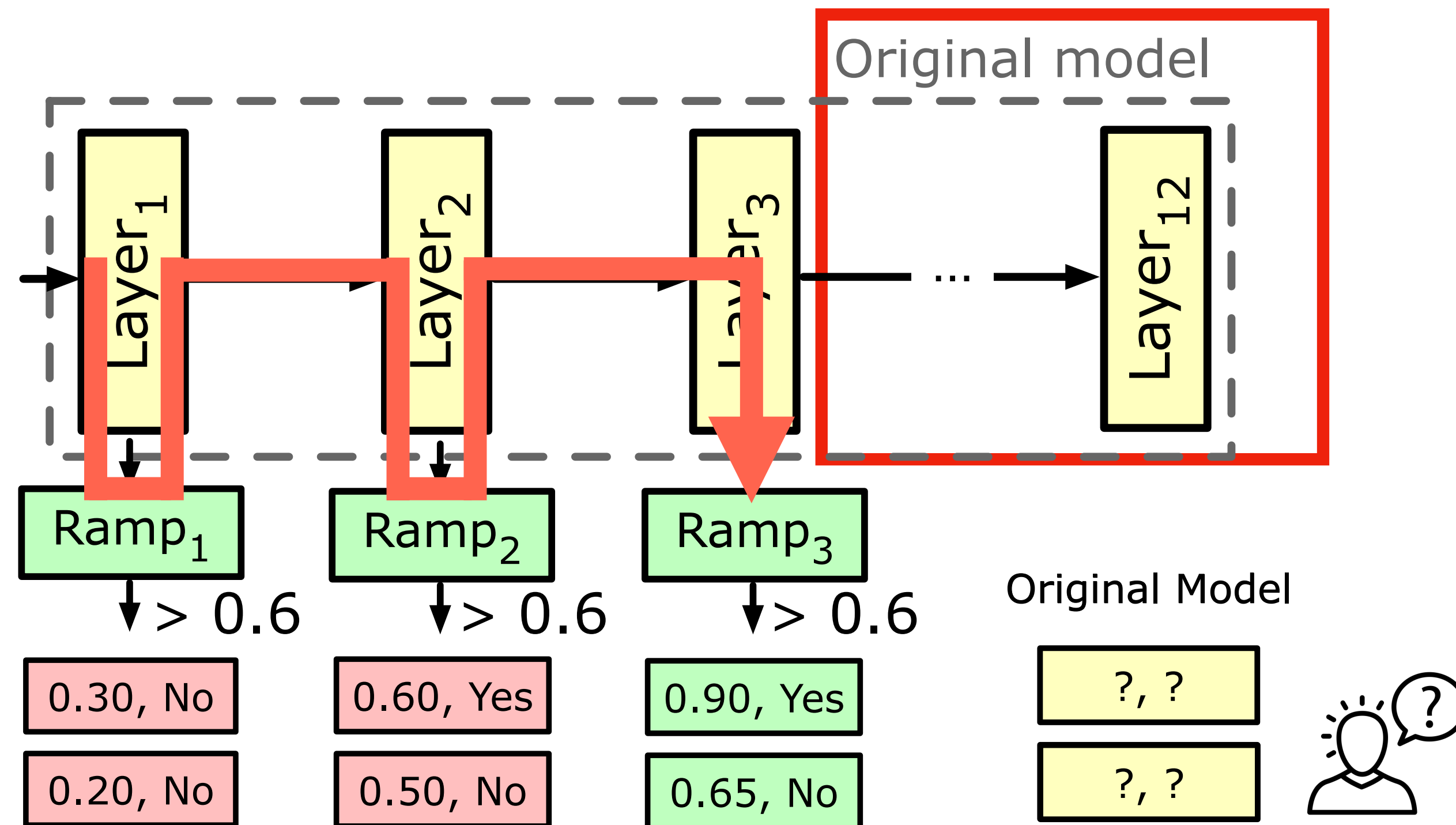
Early Exits: Practical Challenges

low accuracy and high overheads



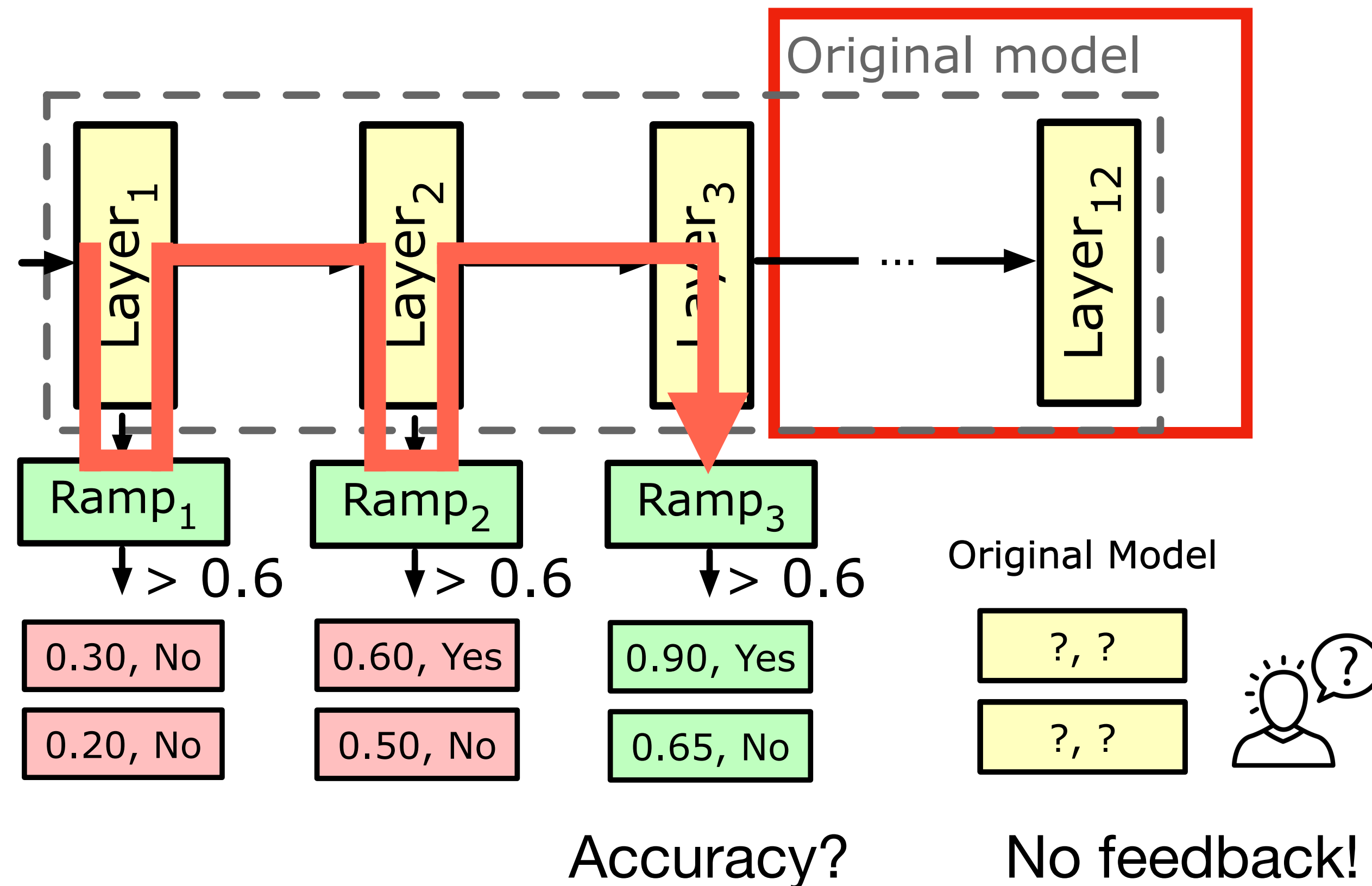
Early Exits: Practical Challenges

low accuracy and high overheads



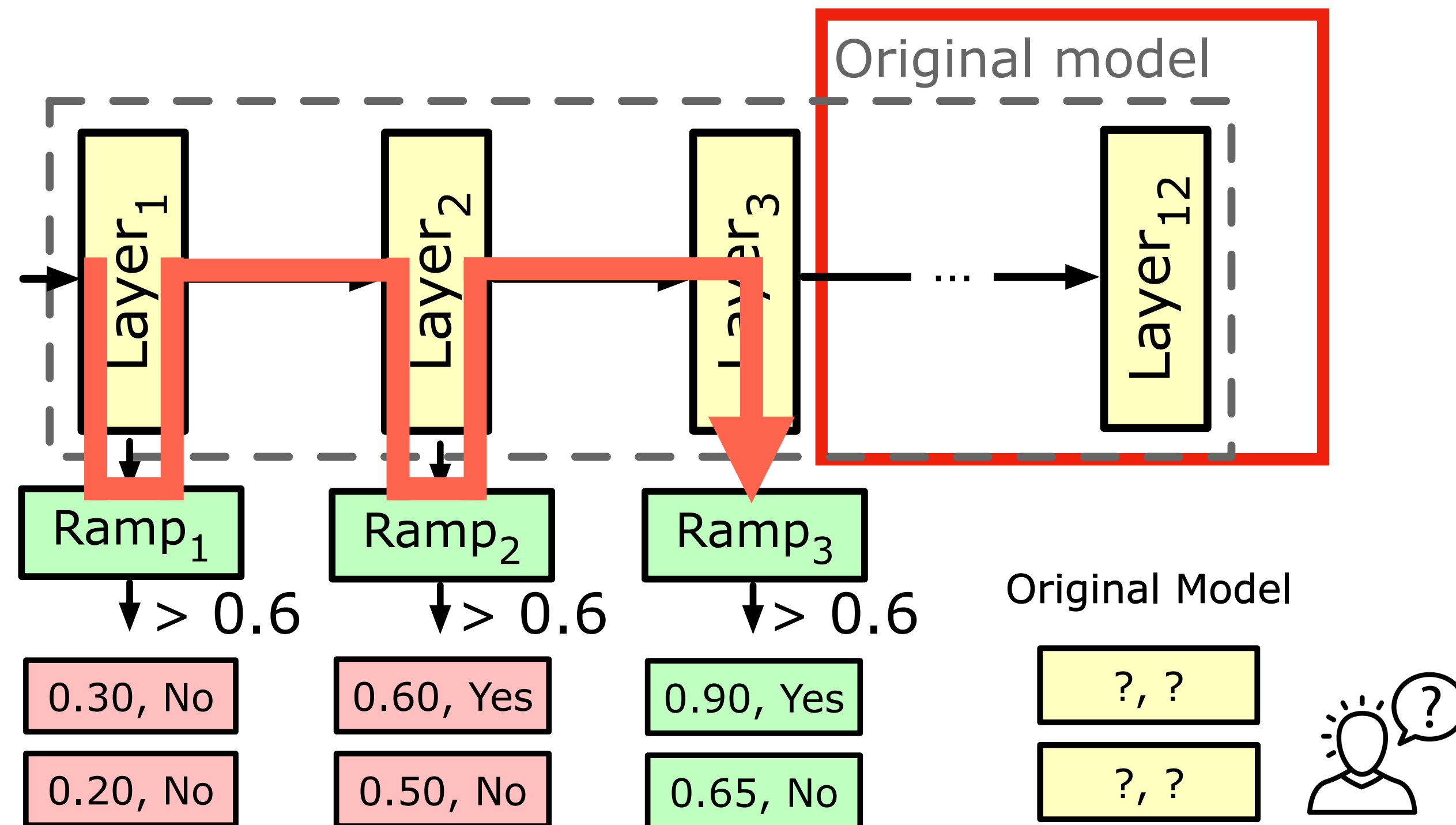
Early Exits: Practical Challenges

low accuracy and high overheads



Early Exits: Practical Challenges

low accuracy and high overheads



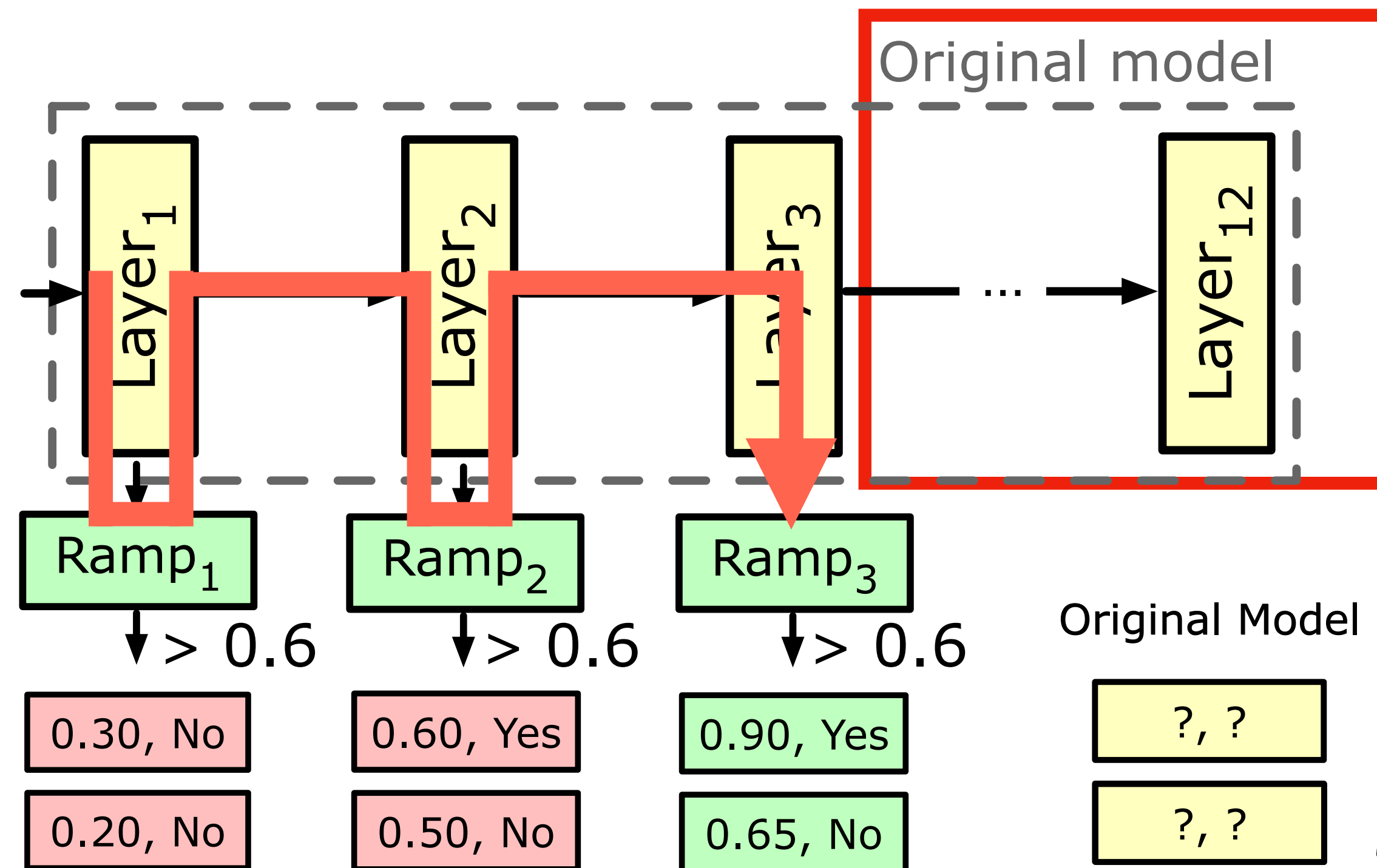
Overheads?

Accuracy?

No feedback!

Early Exits: Practical Challenges

low accuracy and high overheads



Overheads?

Accuracy?

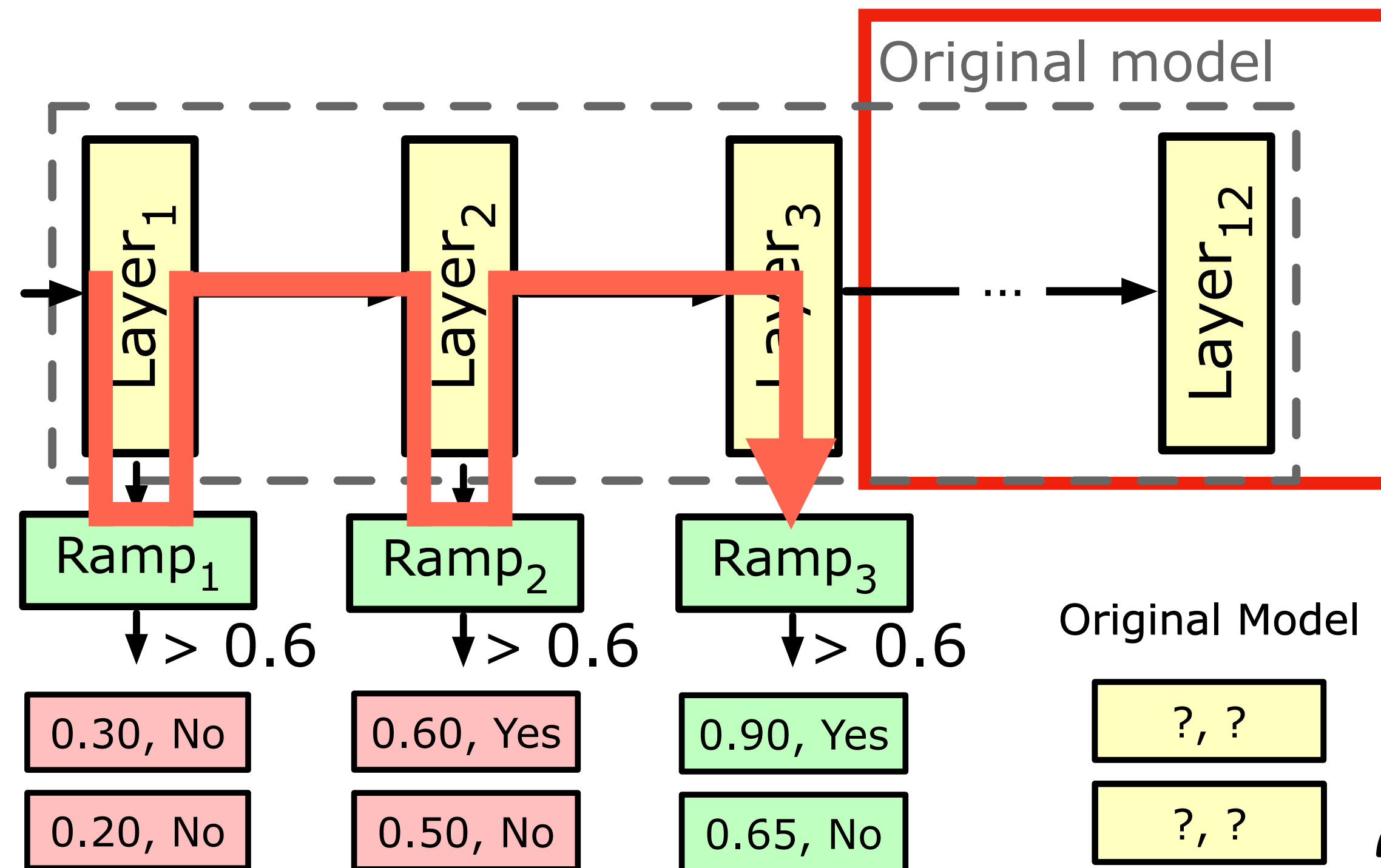
No feedback!

Early Exits aim for saving both **latency** and **computation**

- No feedback for online adaptation

Early Exits: Practical Challenges

low accuracy and high overheads



Overheads?

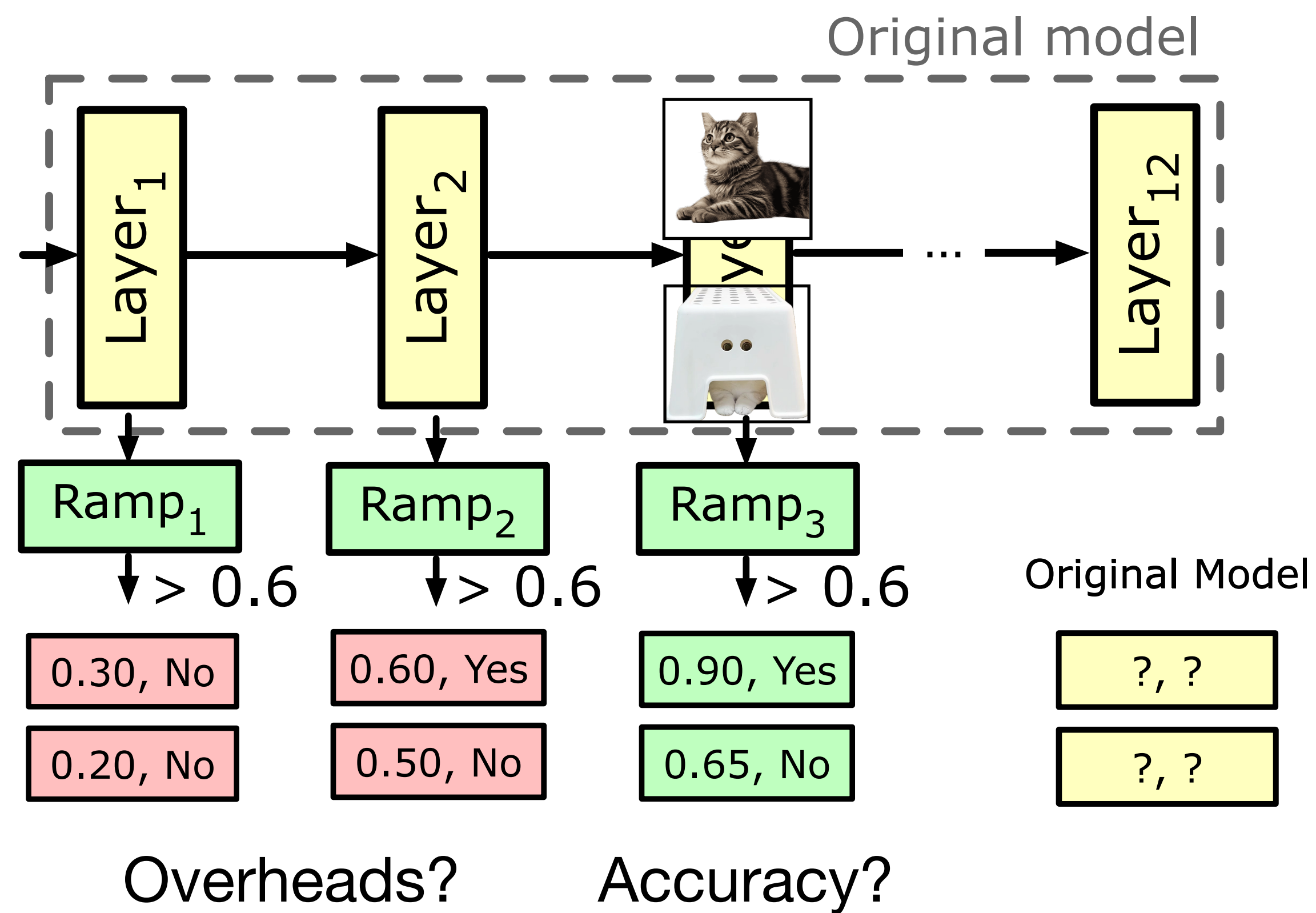
Accuracy?

No feedback!

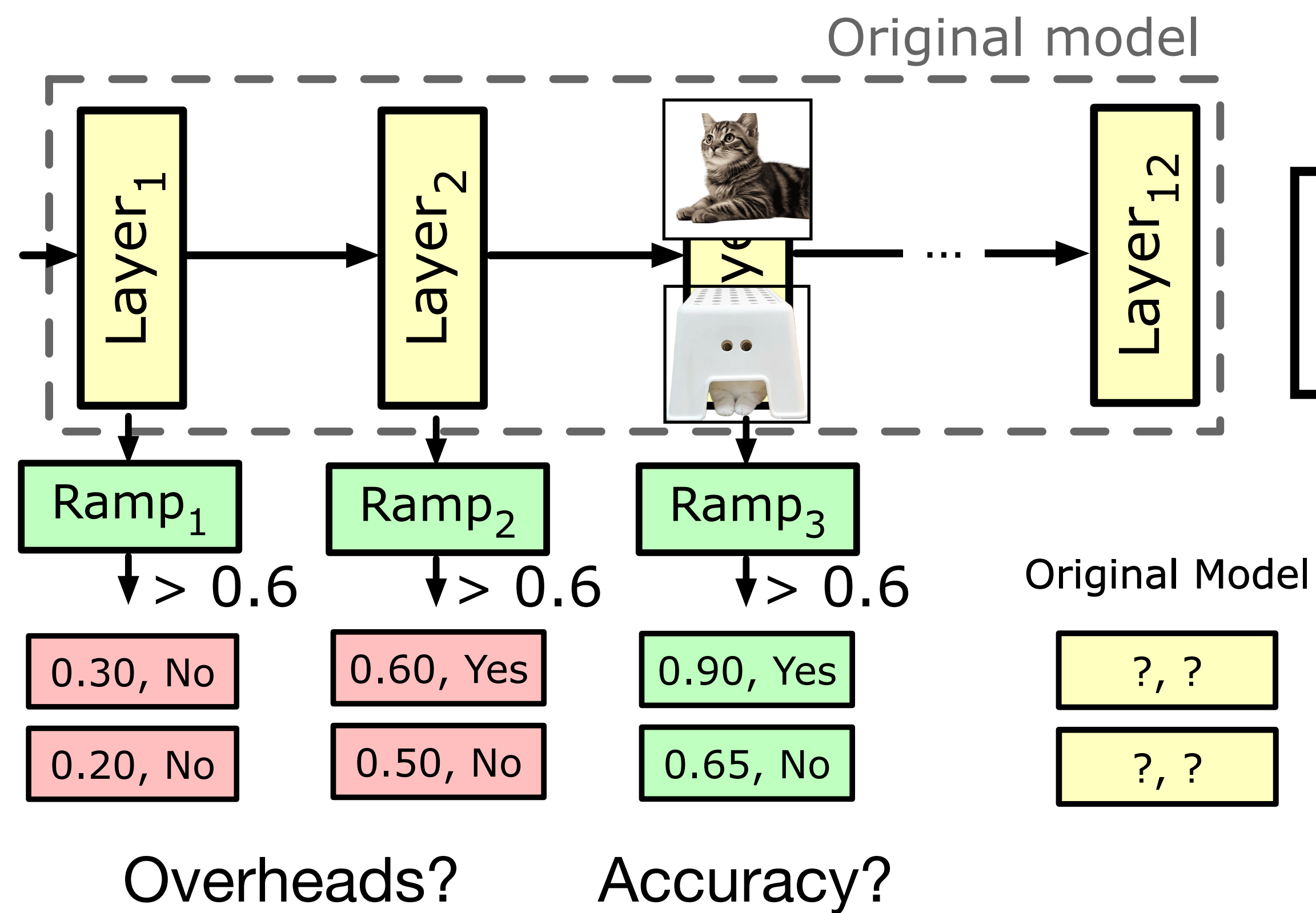
Early Exits aim for saving both **latency** and **computation**

- No feedback for online adaptation
- Up to **24% accuracy loss** and **22% latency overheads**

Insight: Forego Compute Savings for Feedback

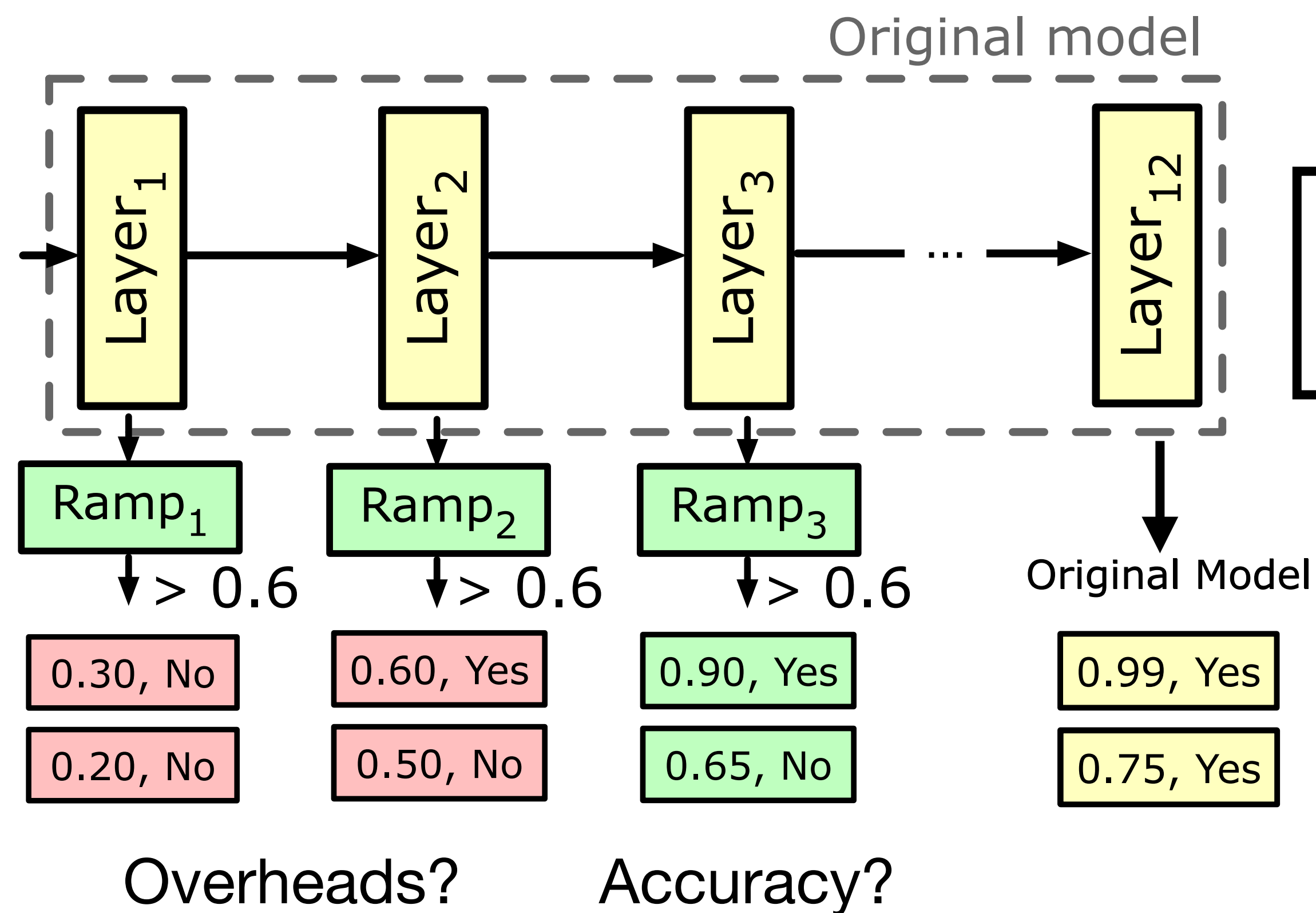


Insight: Forego Compute Savings for Feedback



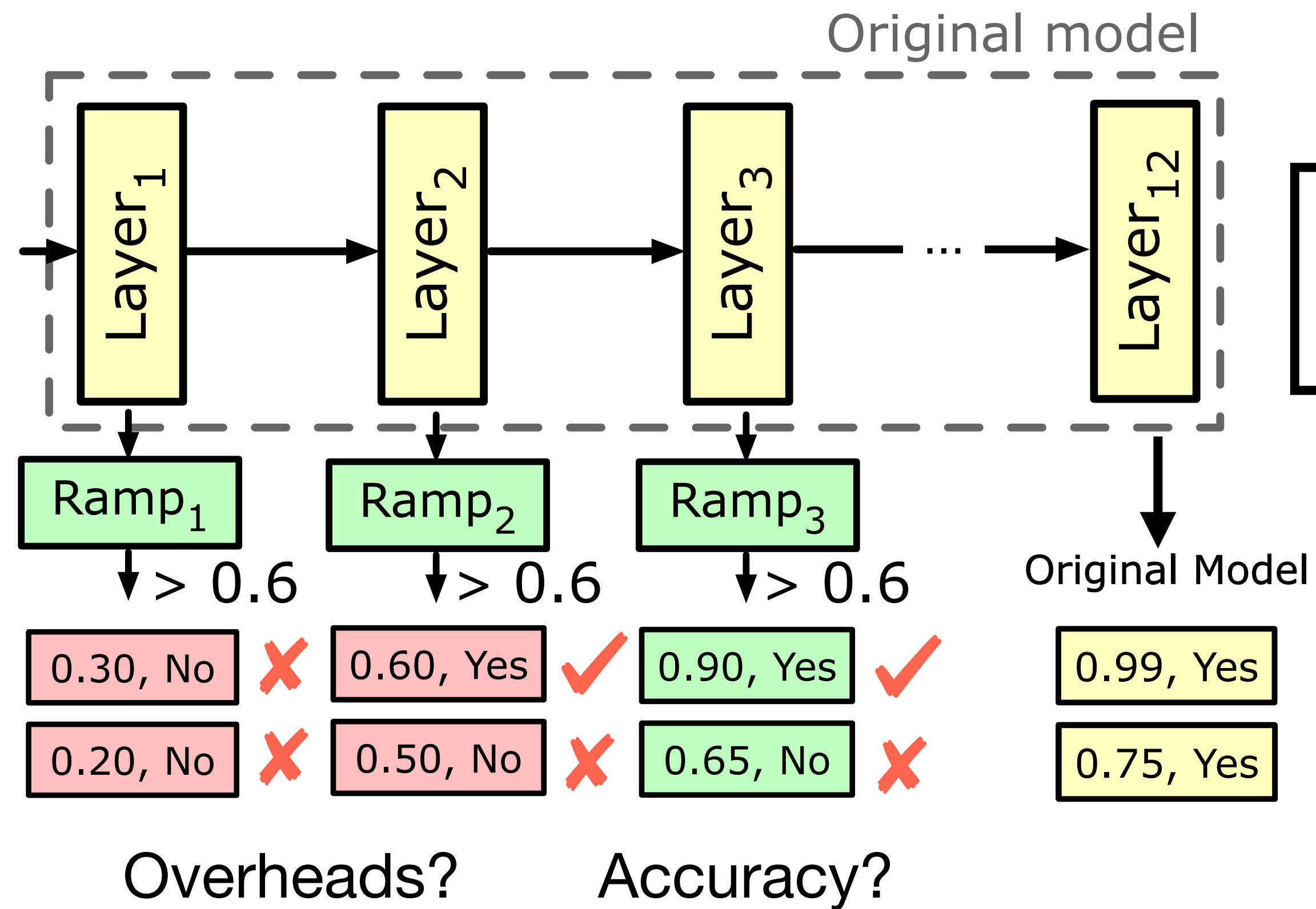
Apparate only exits results for latency, while all inputs run to completion

Insight: Forego Compute Savings for Feedback



**Apparate only exits results for latency,
while all inputs run to completion**

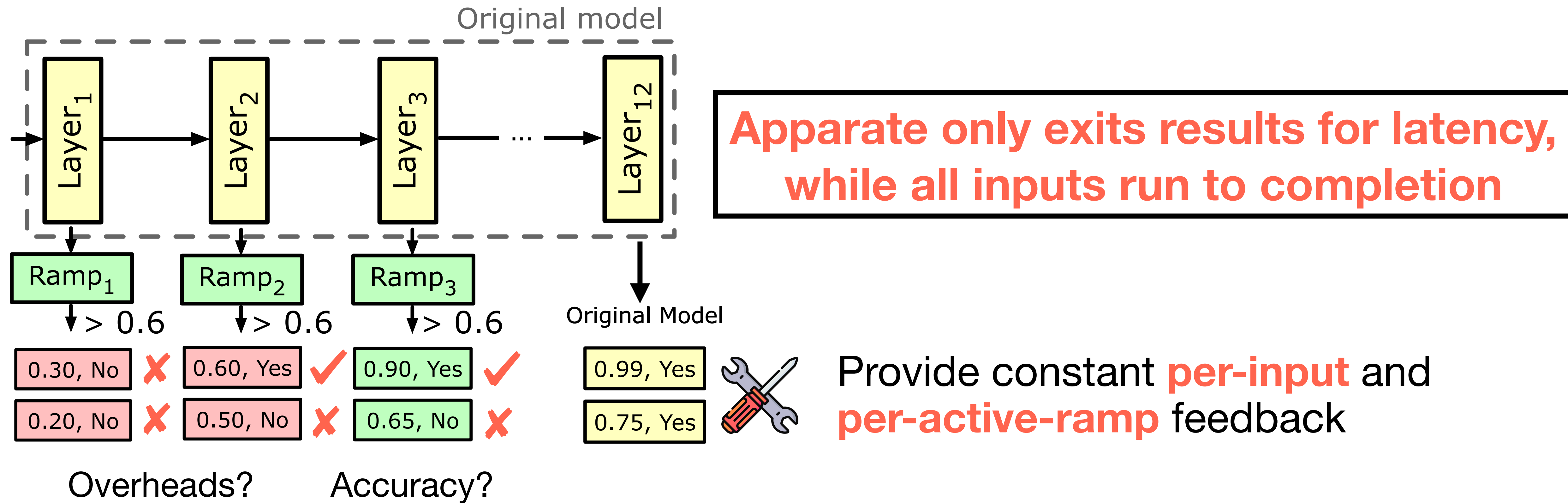
Insight: Forego Compute Savings for Feedback



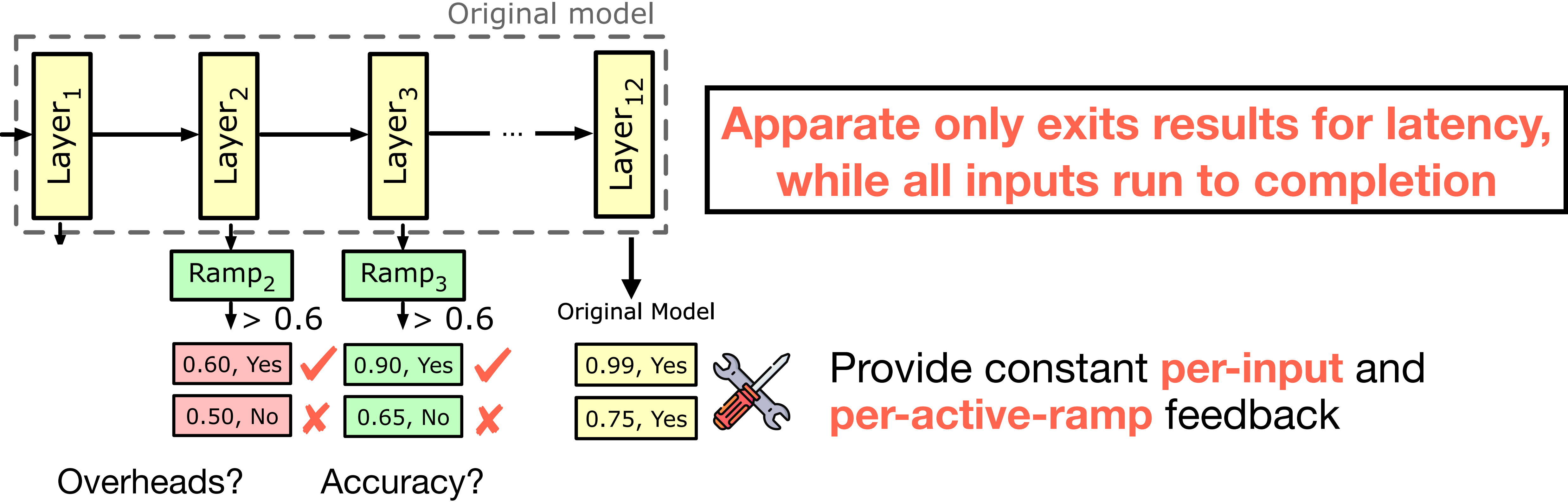
Apparate only exits results for latency,
while all inputs run to completion

Provide constant **per-input** and
per-active-ramp feedback

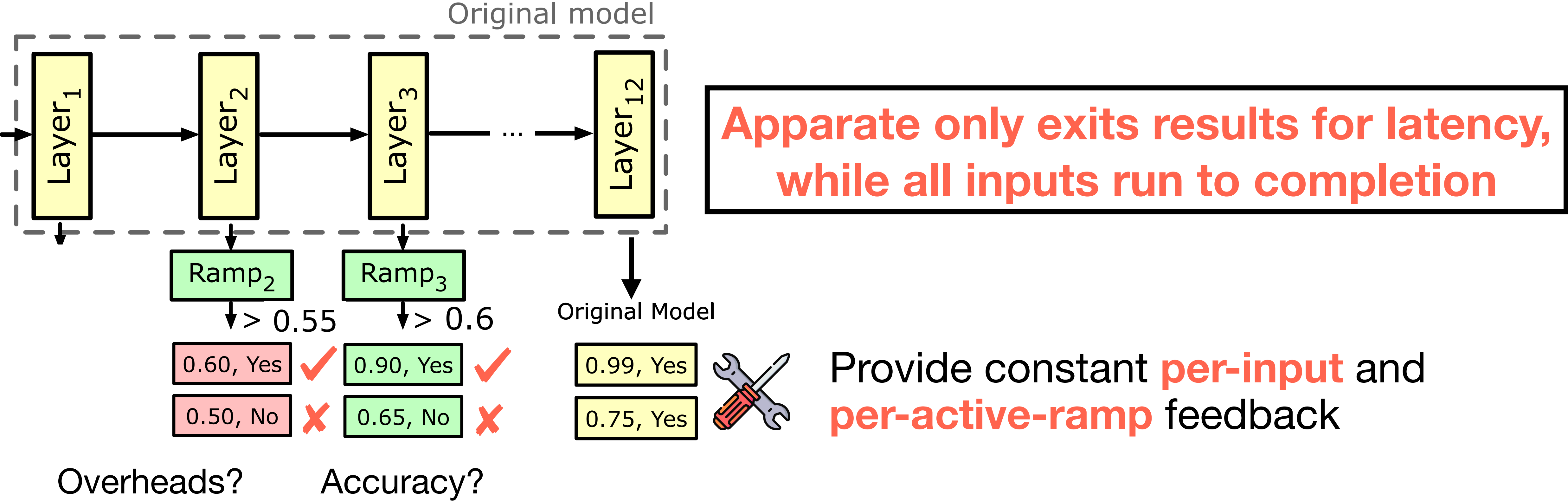
Insight: Forego Compute Savings for Feedback



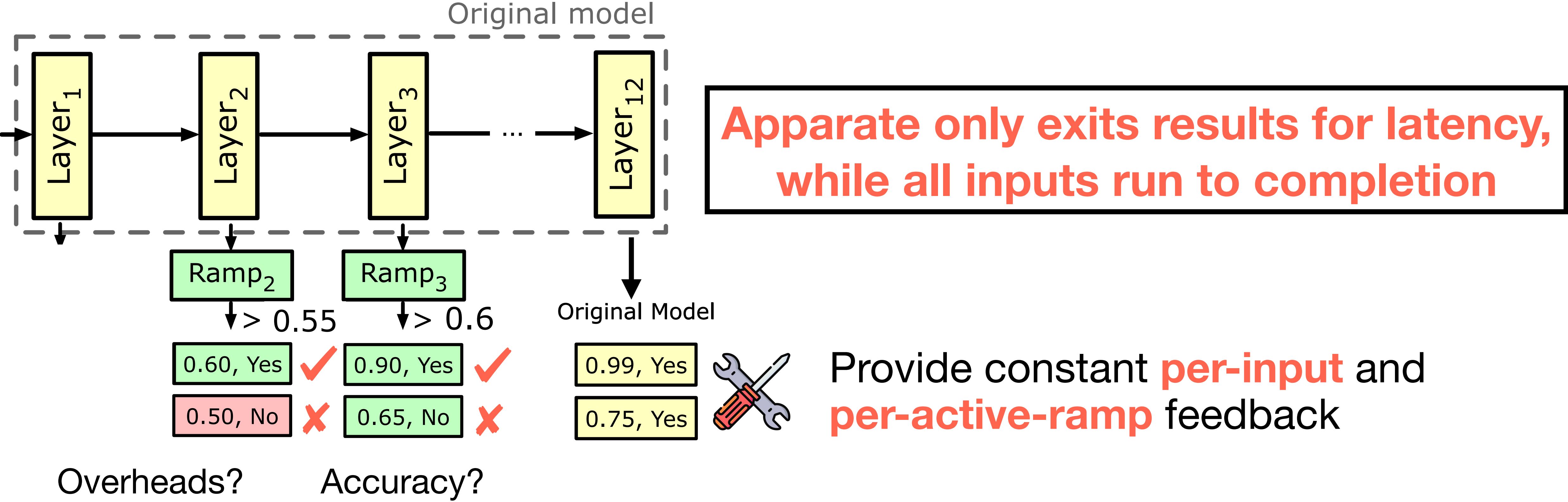
Insight: Forego Compute Savings for Feedback



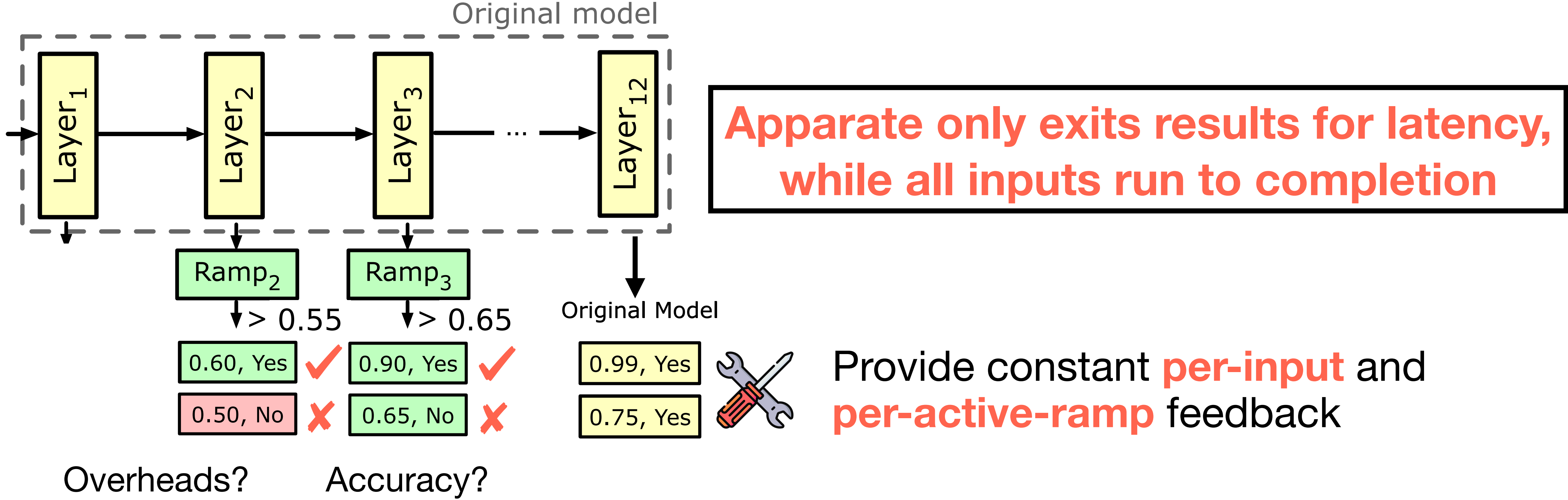
Insight: Forego Compute Savings for Feedback



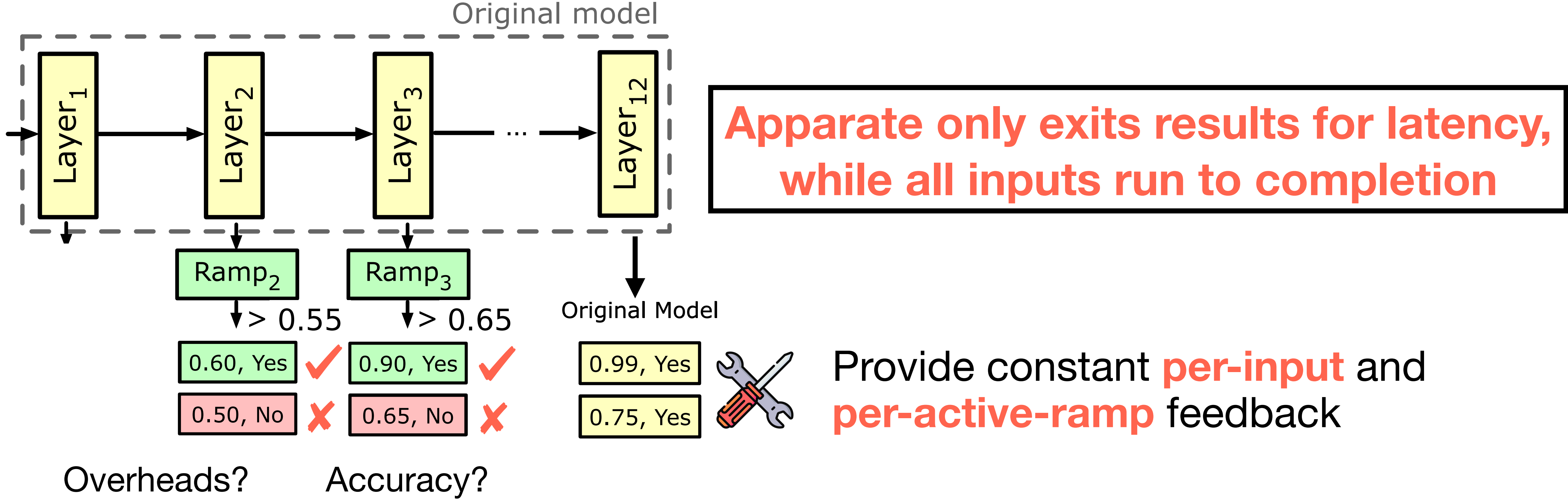
Insight: Forego Compute Savings for Feedback



Insight: Forego Compute Savings for Feedback

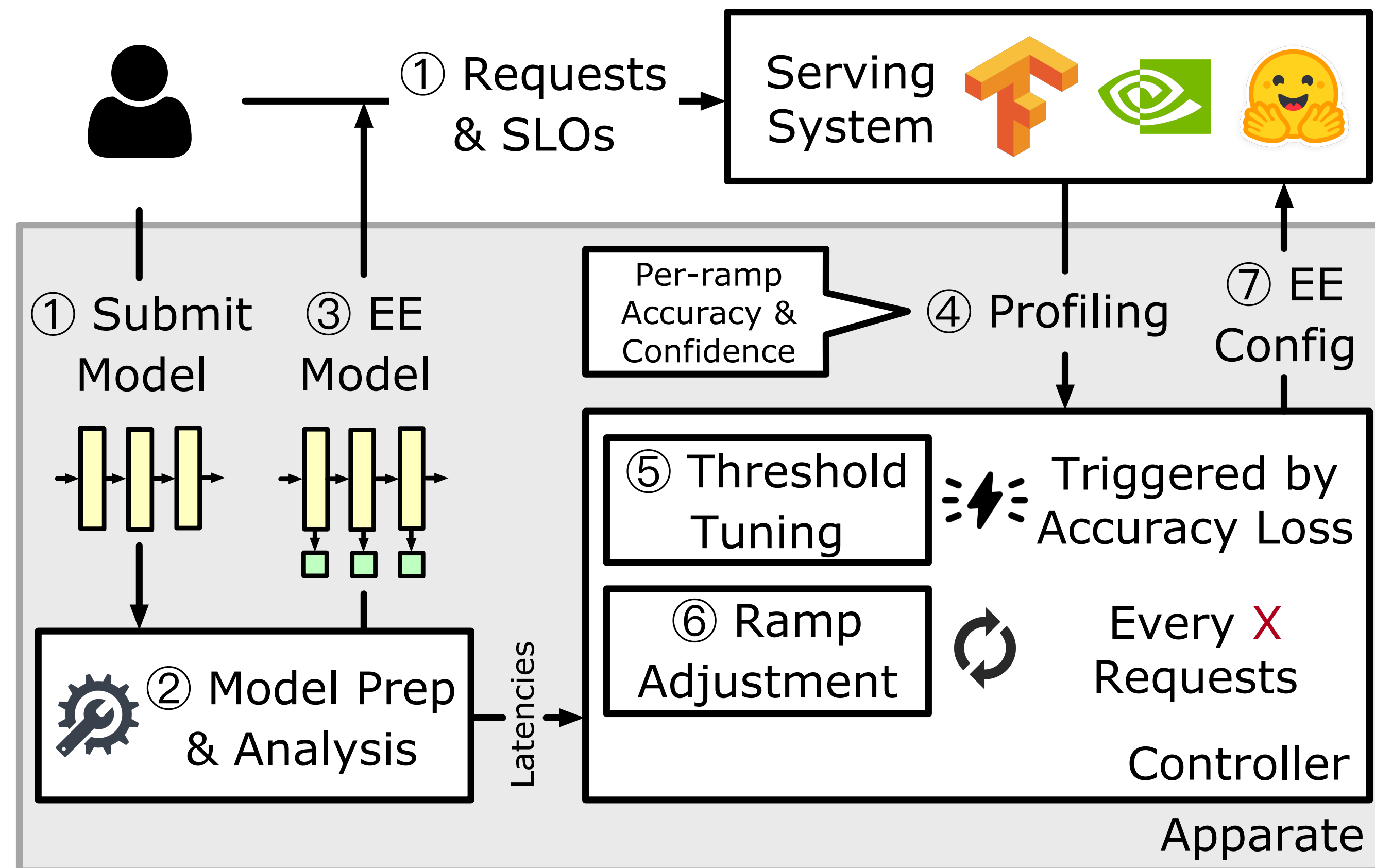


Insight: Forego Compute Savings for Feedback



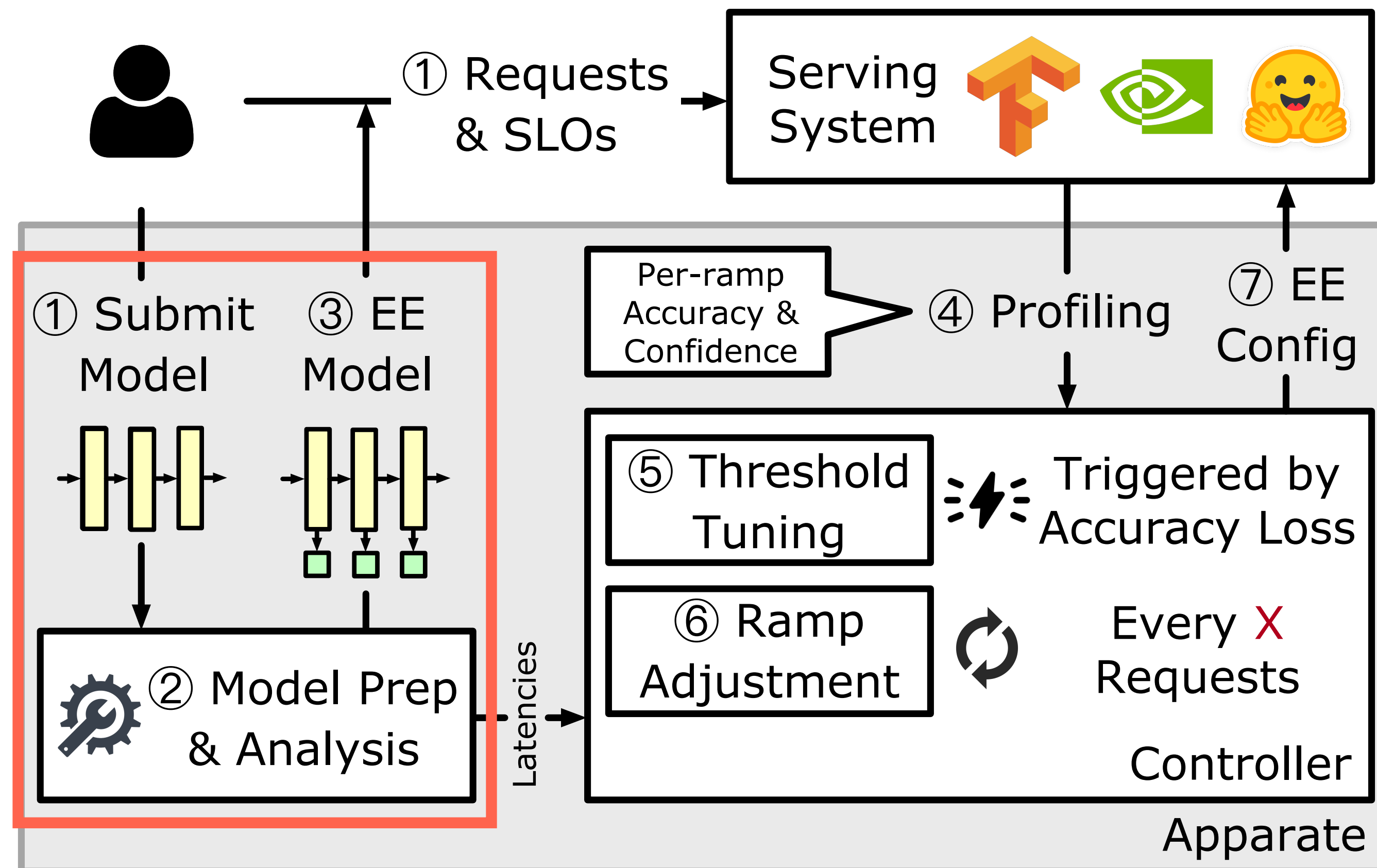
Apparate

the first system that automatically integrates and manages EEs for ML inference



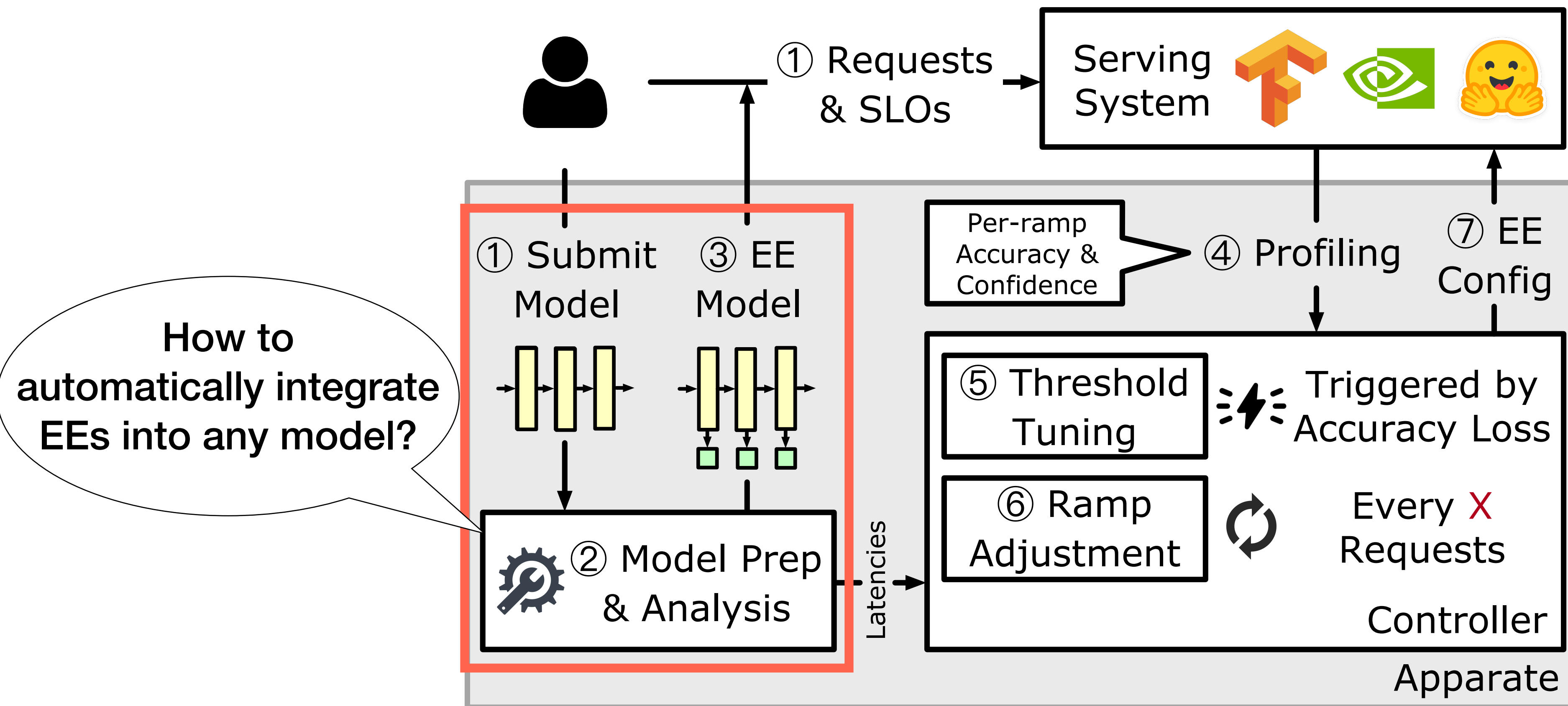
Apparate

the first system that automatically integrates and manages EEs for ML inference



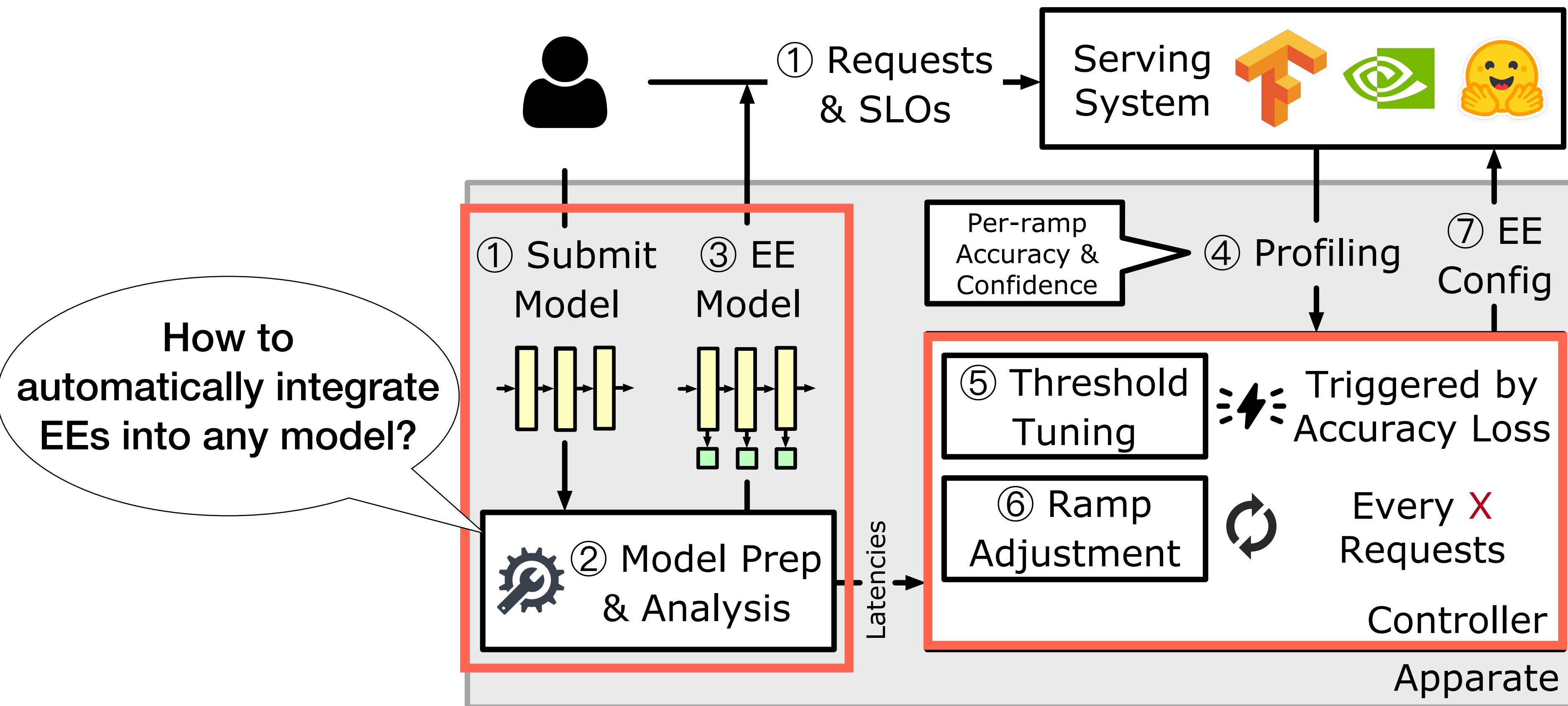
Apparate

the first system that automatically integrates and manages EEs for ML inference



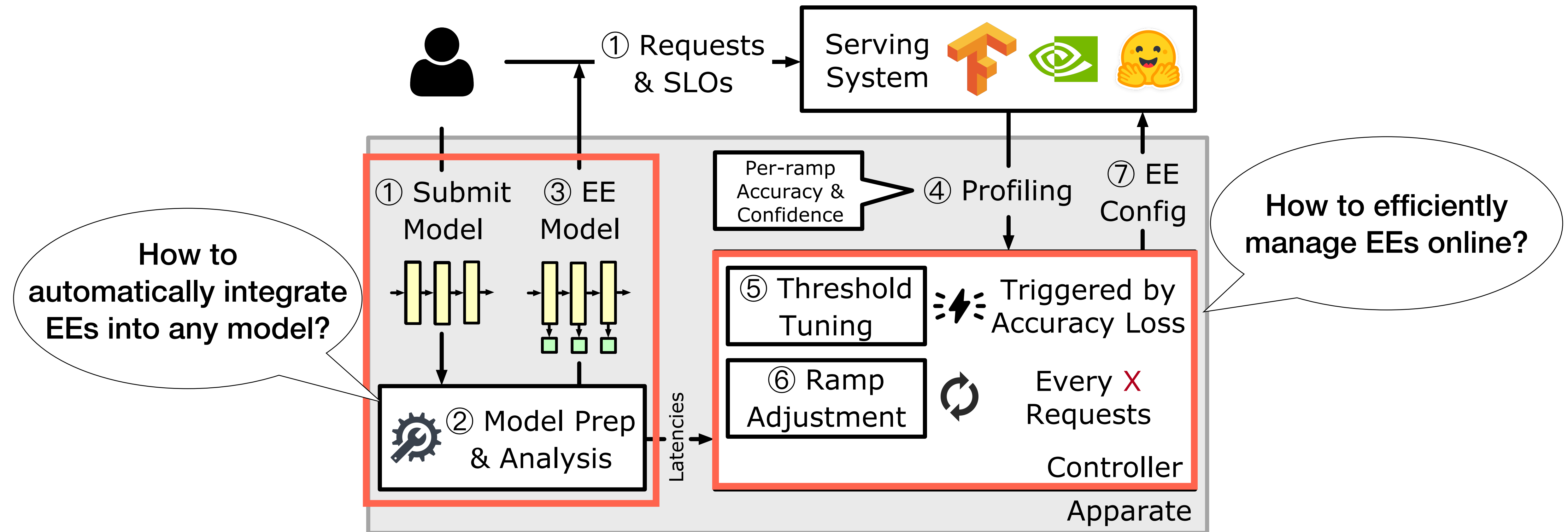
Apparate

the first system that automatically integrates and manages EEs for ML inference



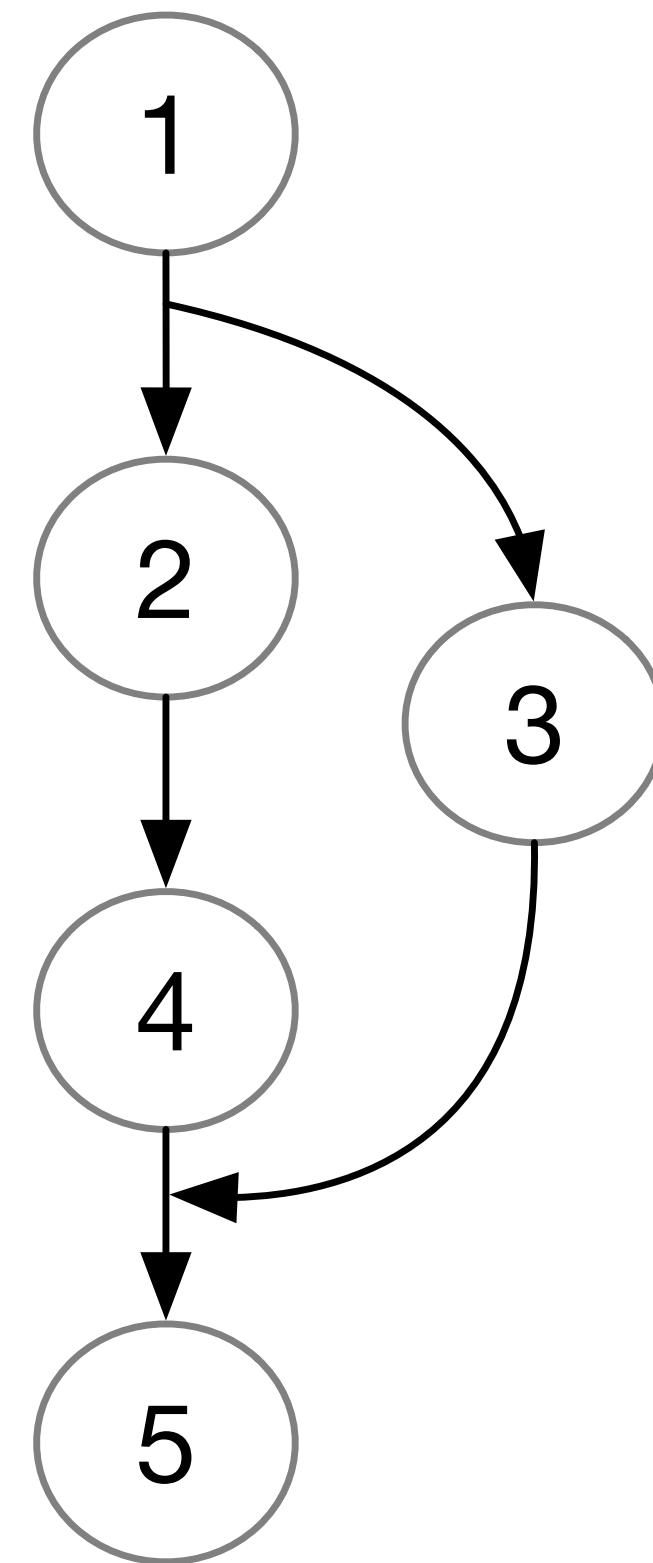
Apparate

the first system that automatically integrates and manages EEs for ML inference



Apparate: Automatic integration for EEs

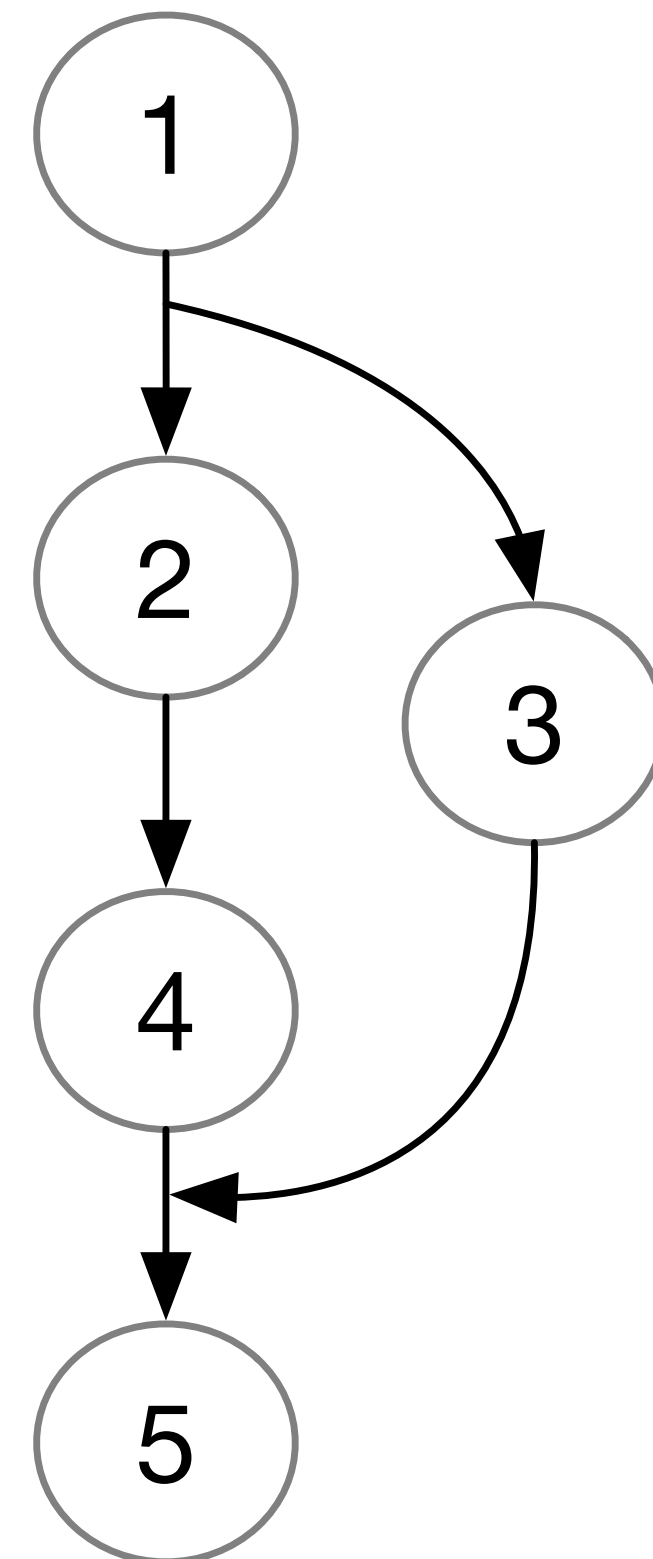
challenge: ramp location, architecture, and weights



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

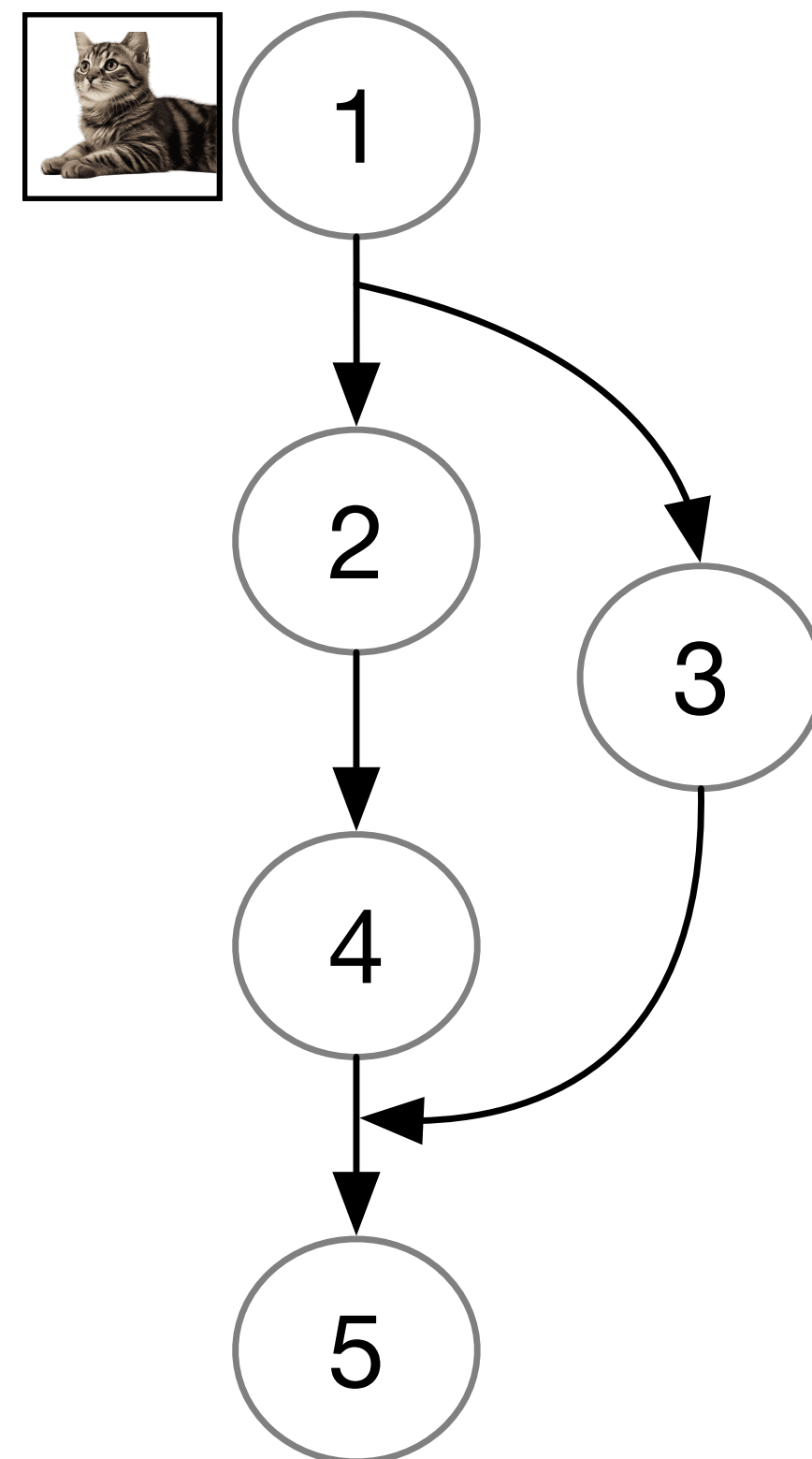
Ramp Location



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

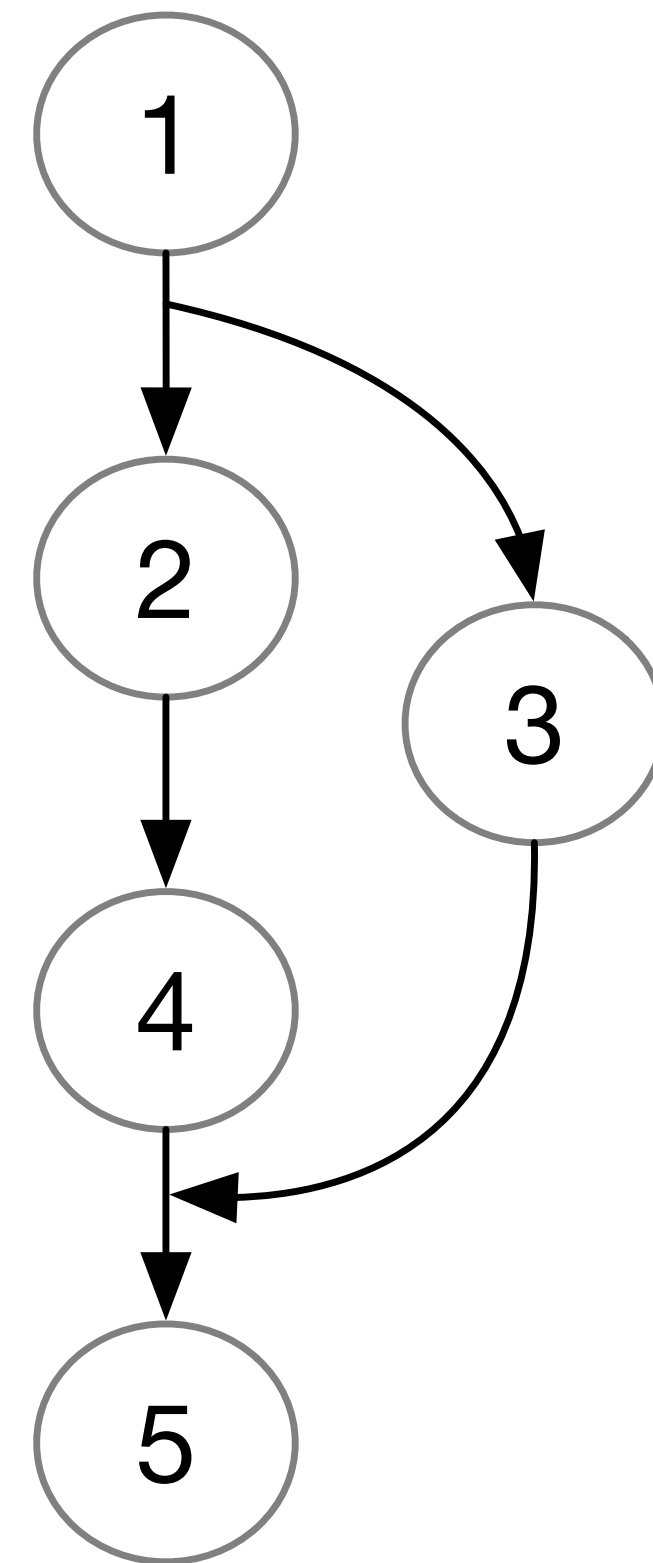
Ramp Location



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

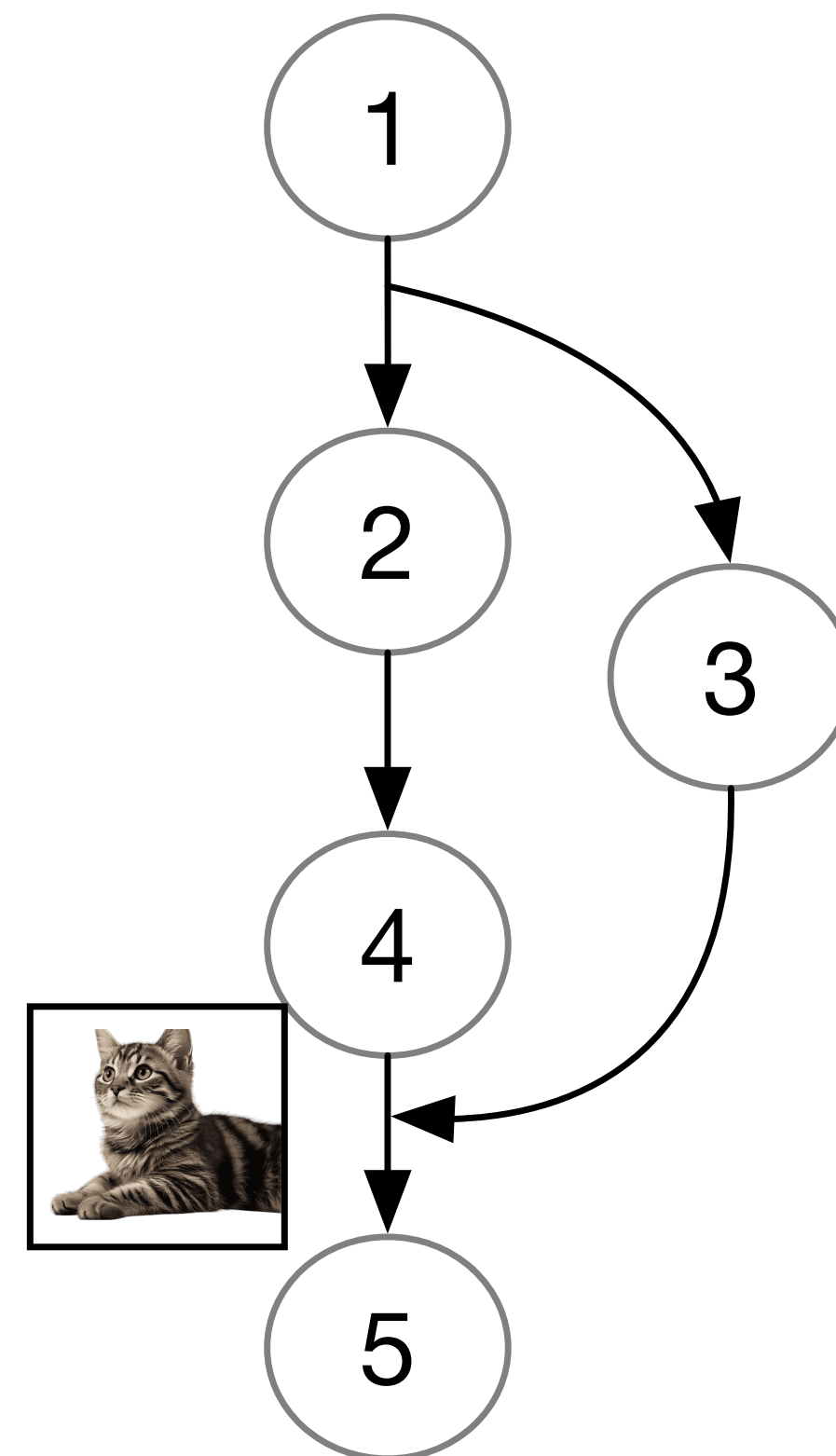
Ramp Location



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

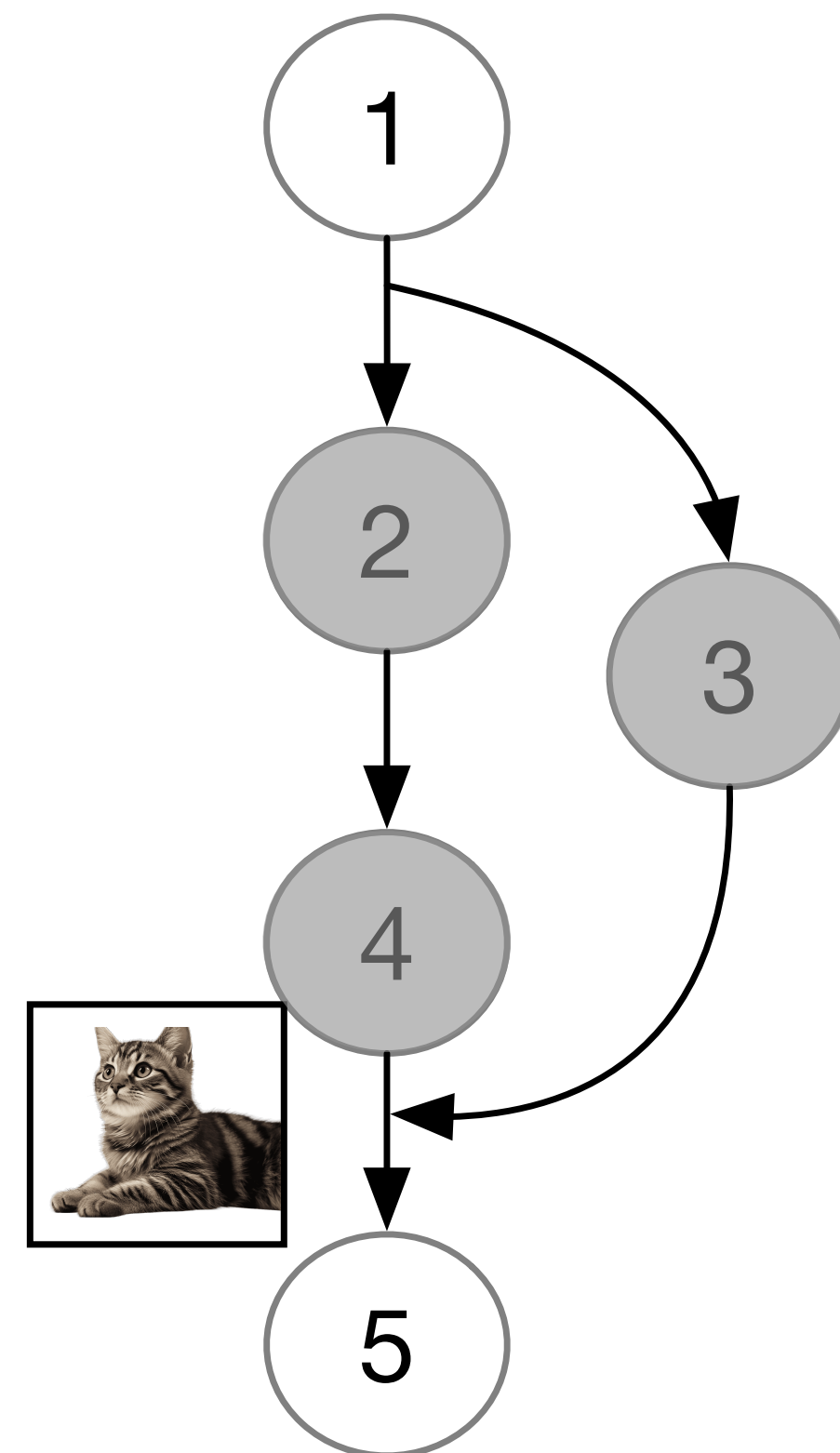
Ramp Location



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

Ramp Location



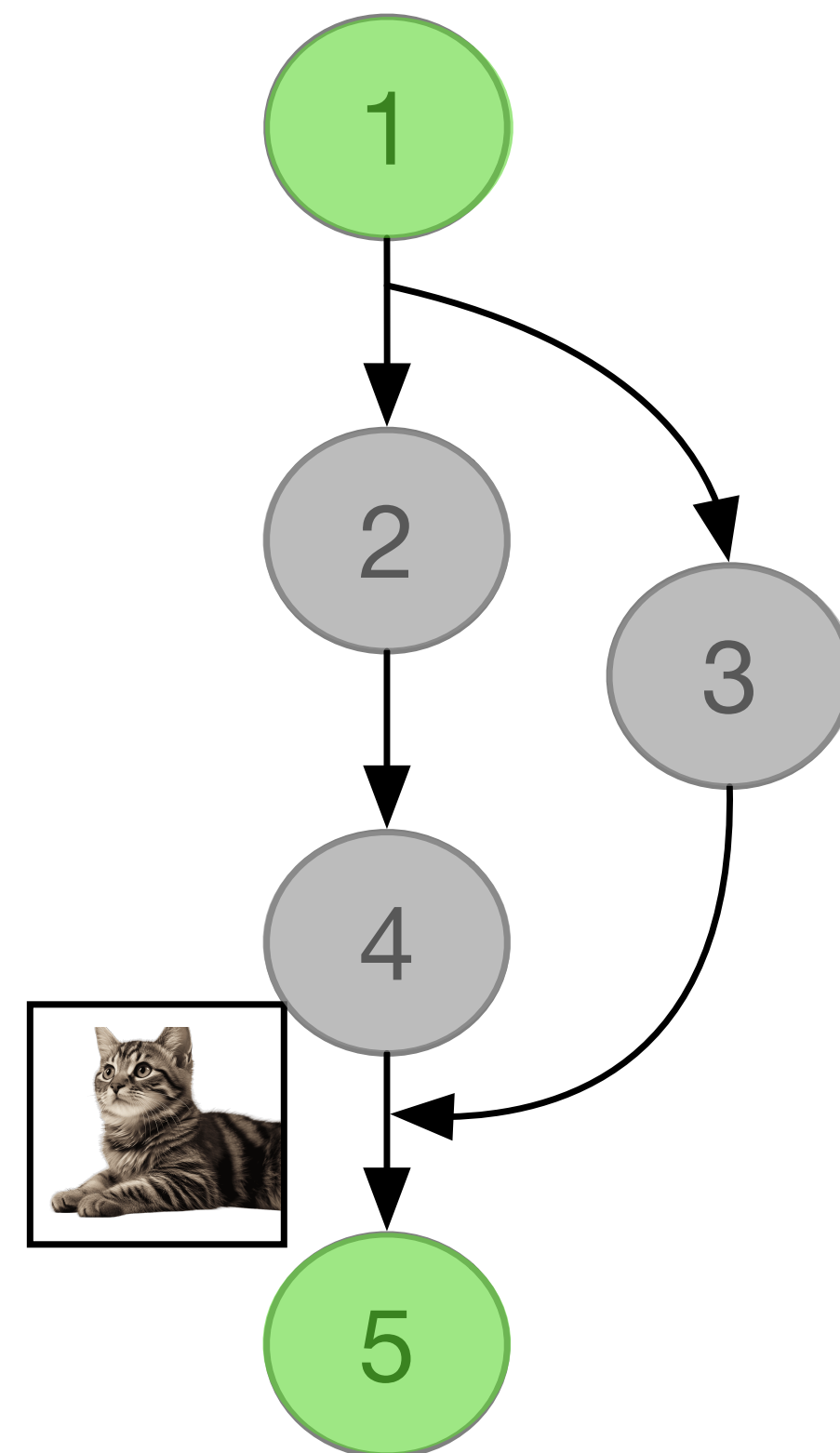
Partial features
result in low ramp accuracy!

Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

Ramp Location

- cutting vertices

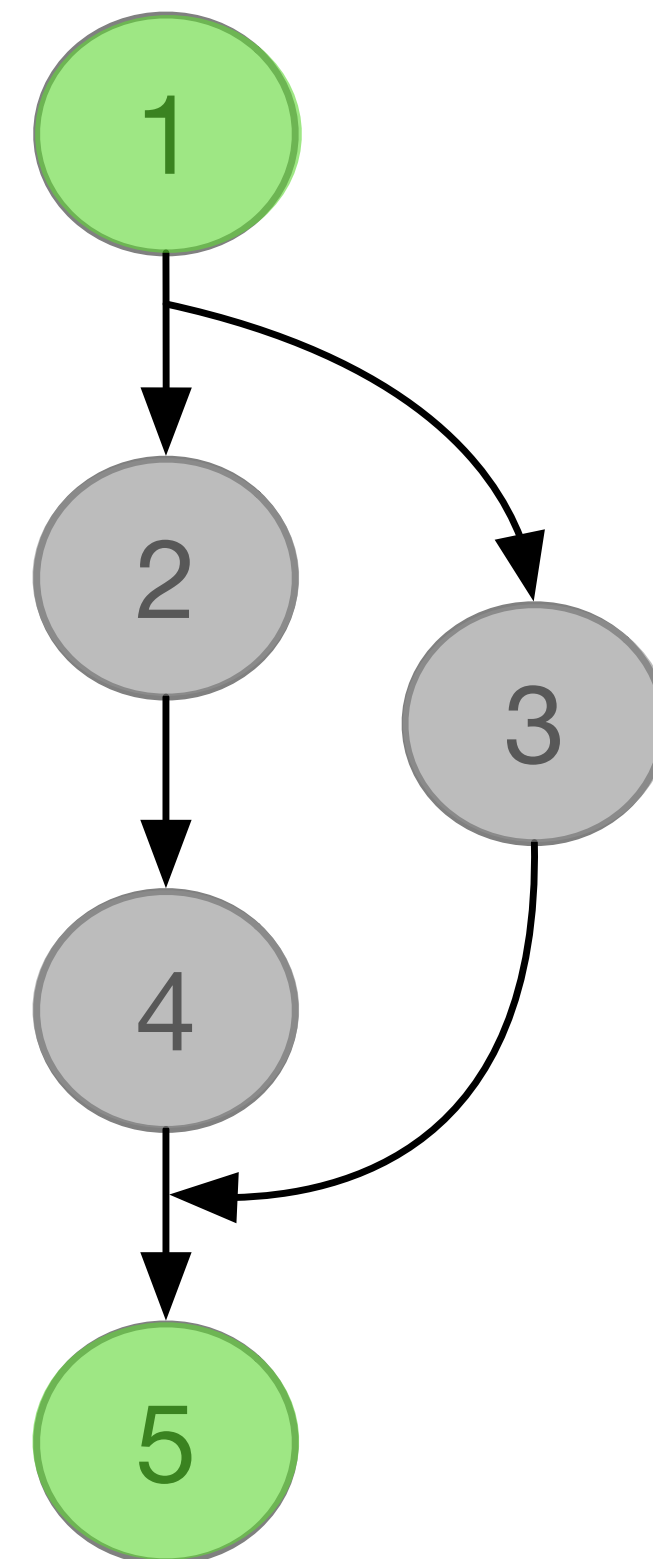


Partial features
result in low ramp accuracy!

Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

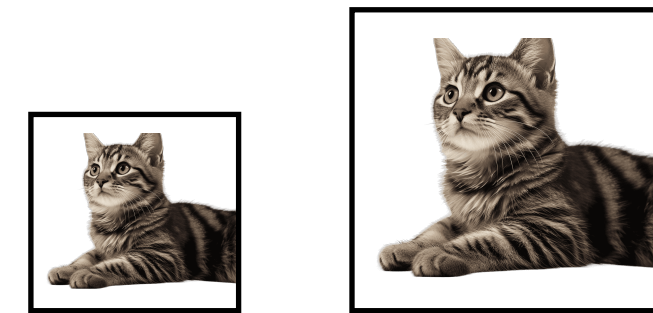
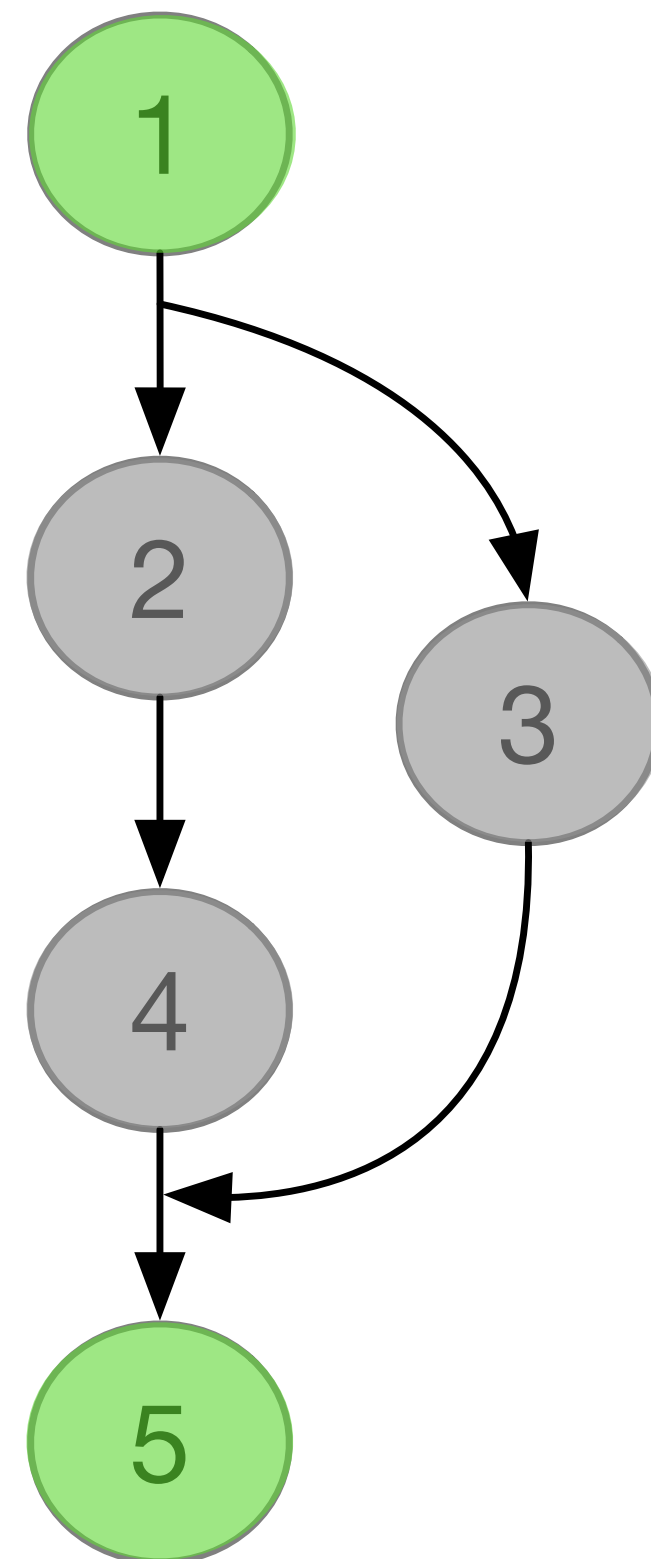
Ramp Architecture



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

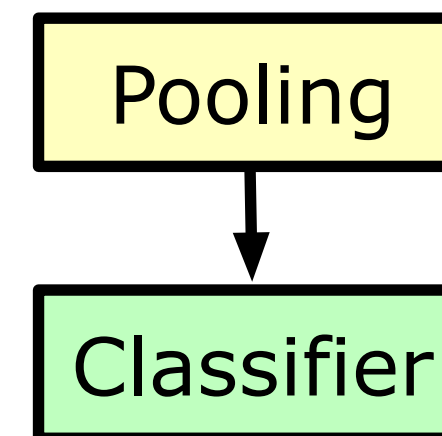
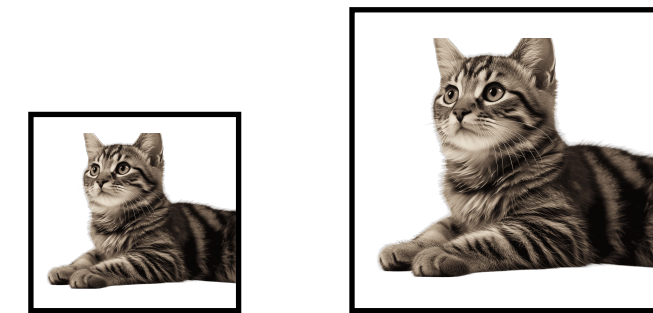
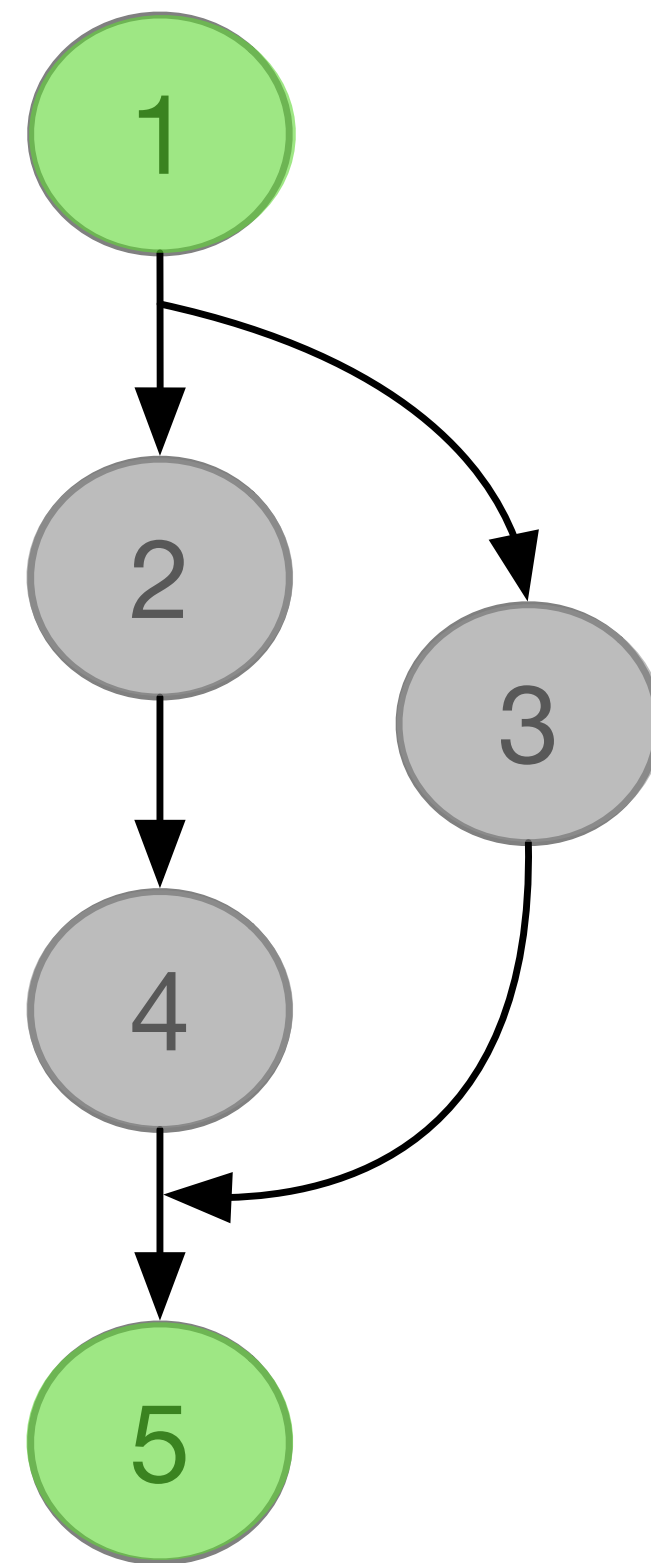
Ramp Architecture



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

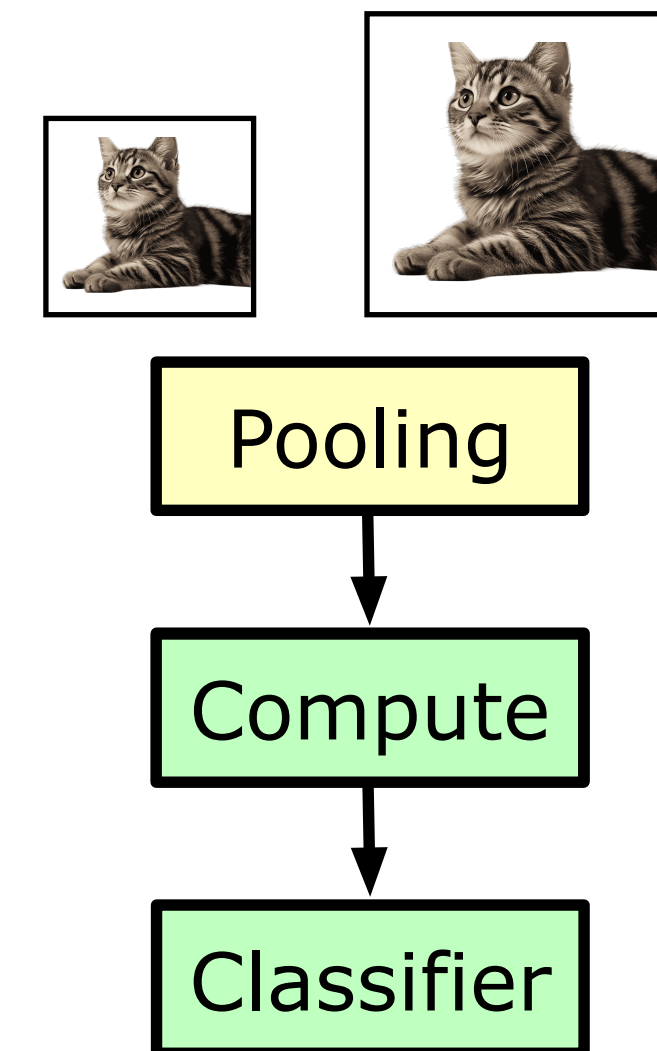
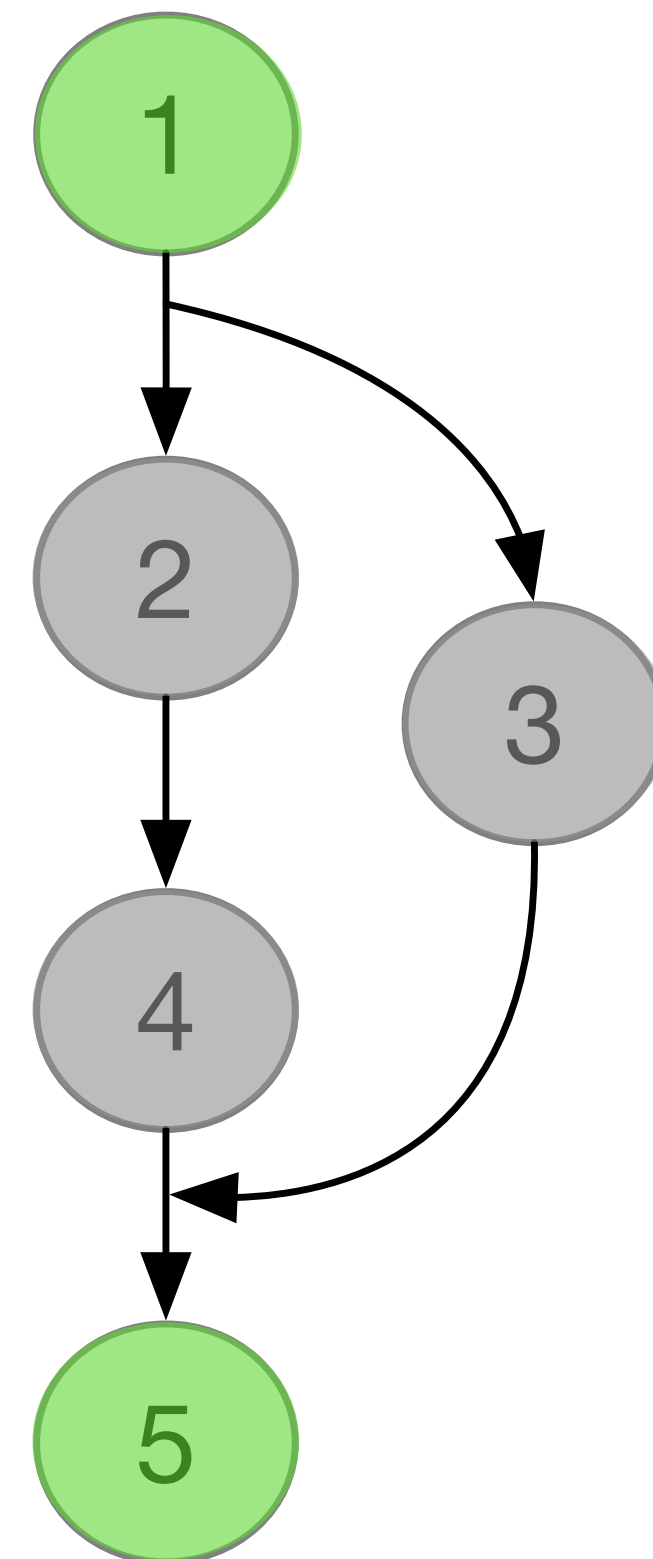
Ramp Architecture



Apparate: Automatic integration for EEs

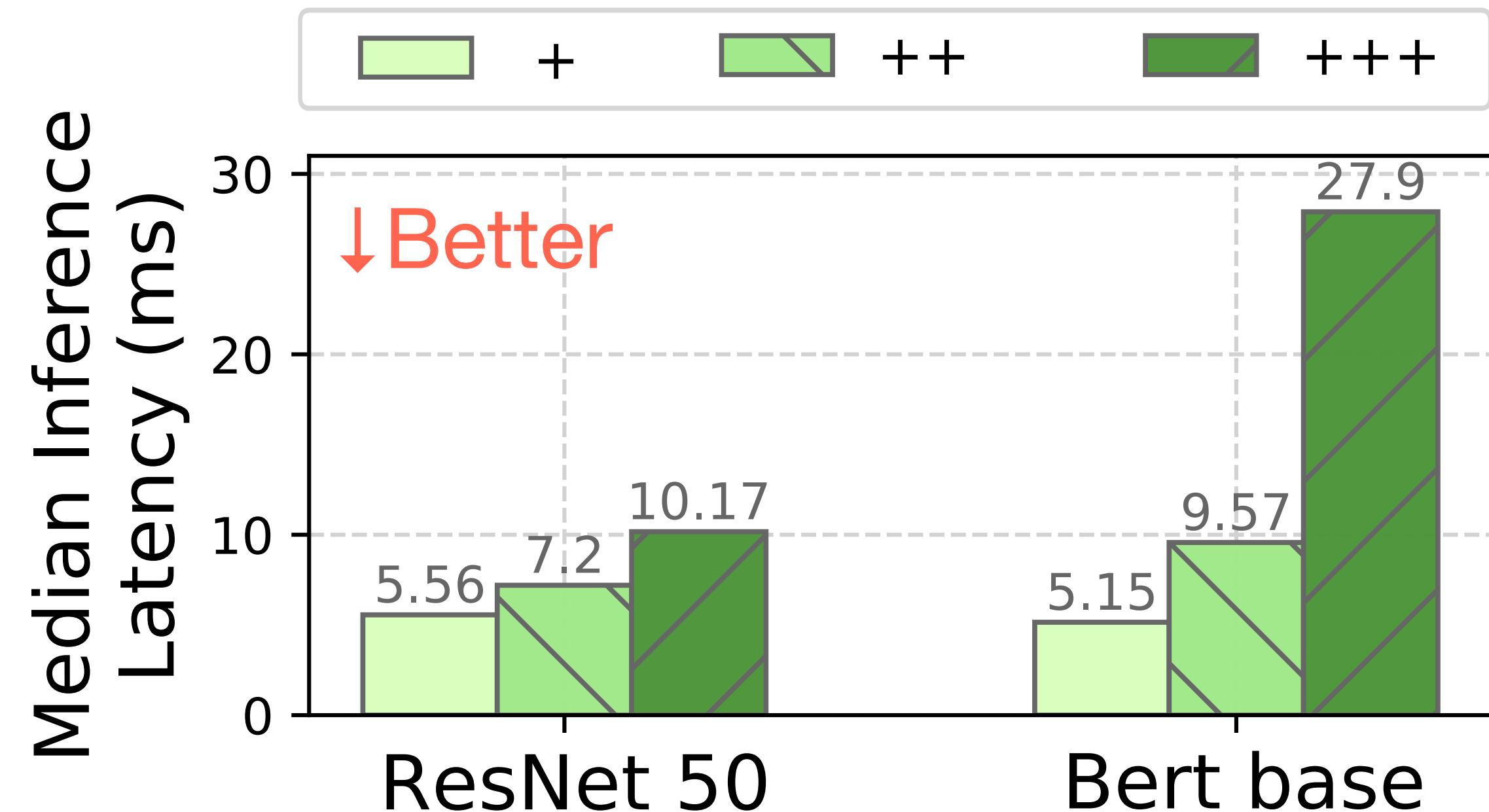
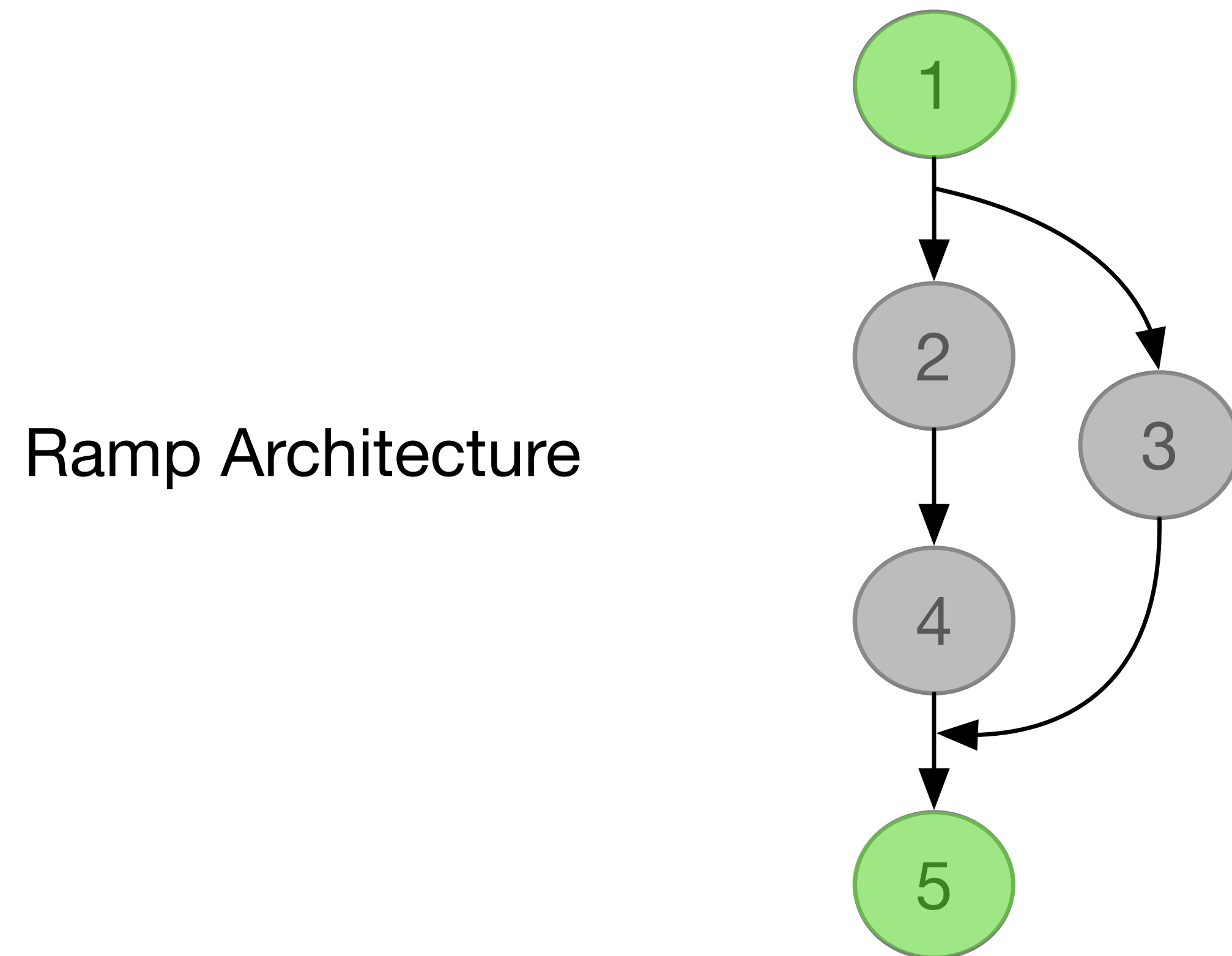
challenge: ramp location, architecture, and weights

Ramp Architecture



Apparate: Automatic integration for EEs

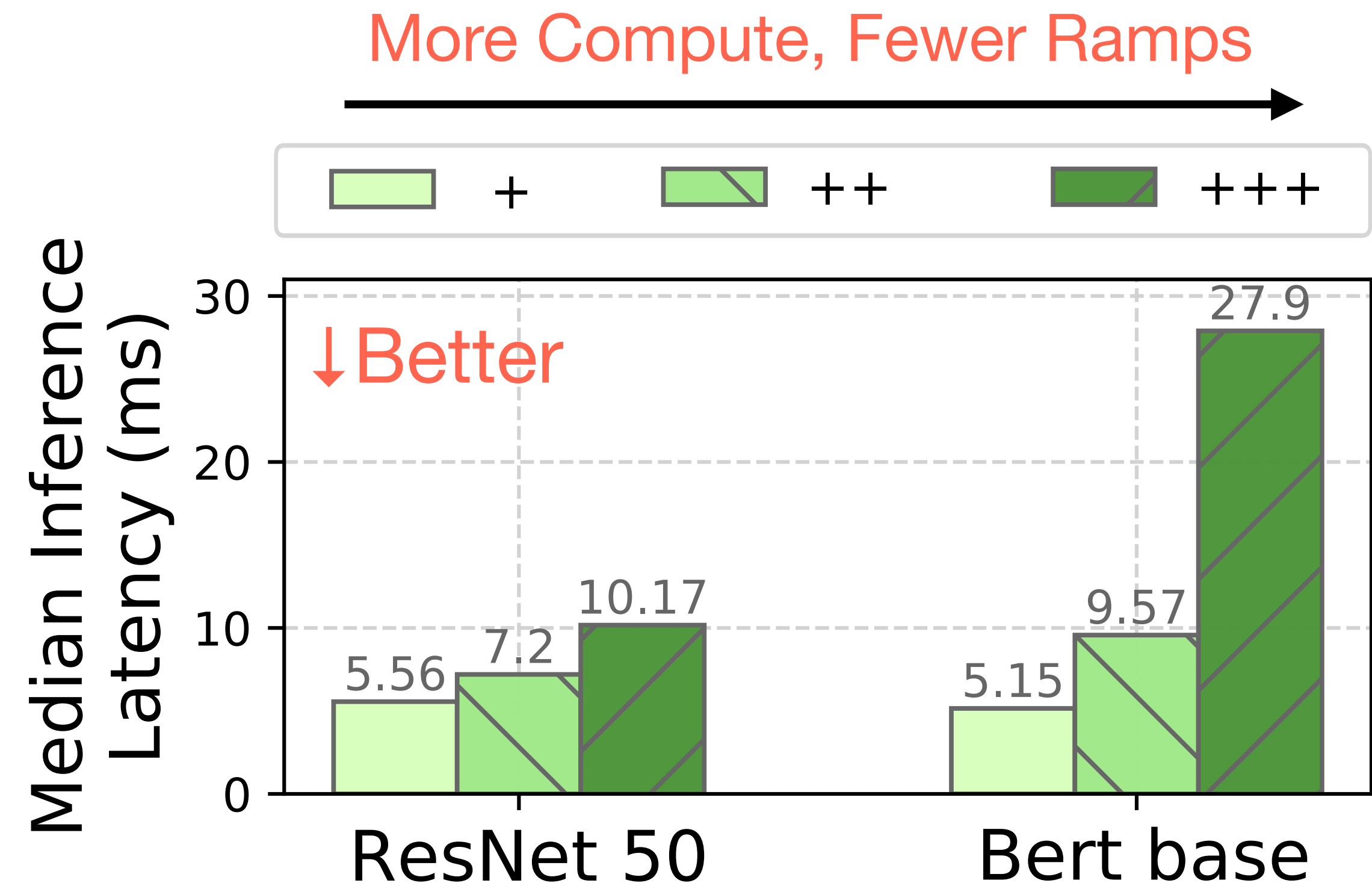
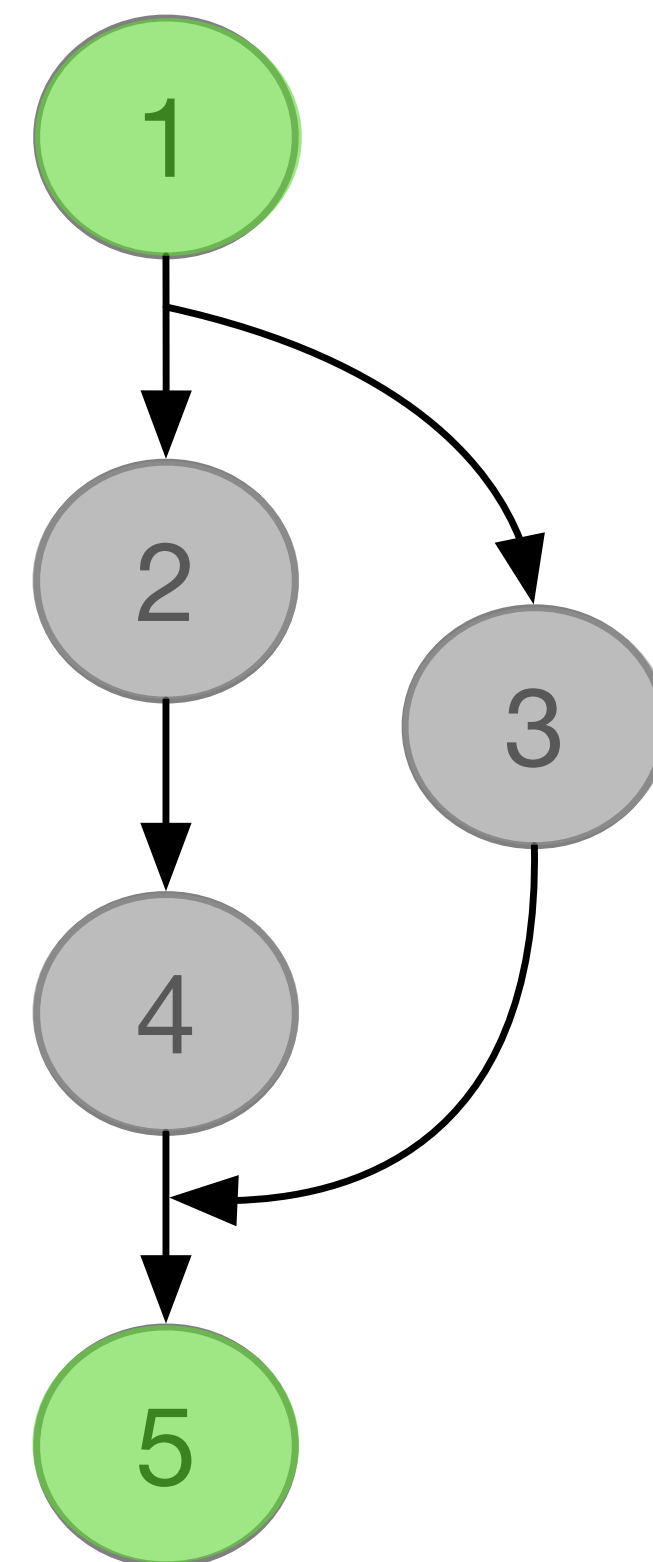
challenge: ramp location, architecture, and weights



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

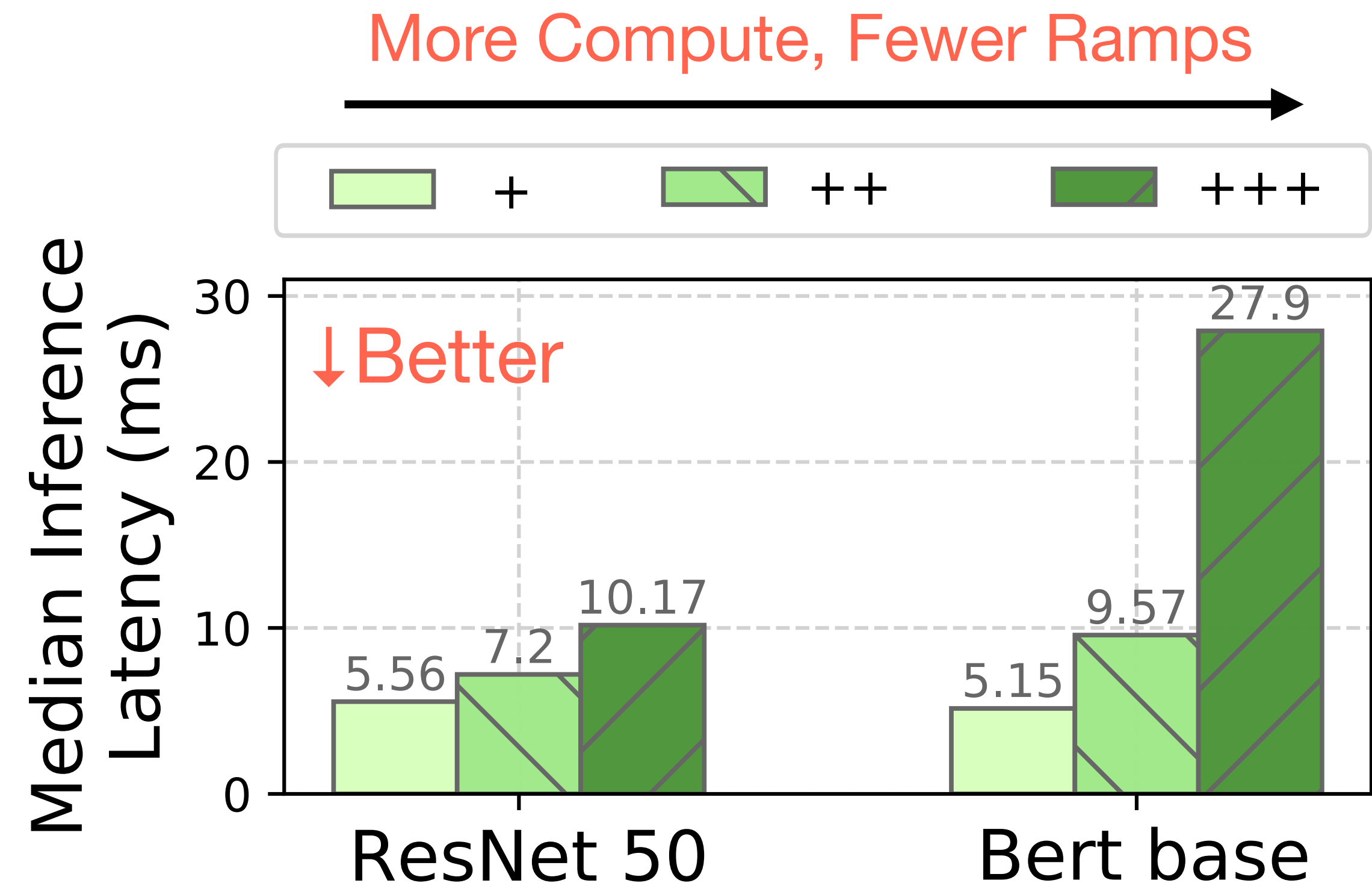
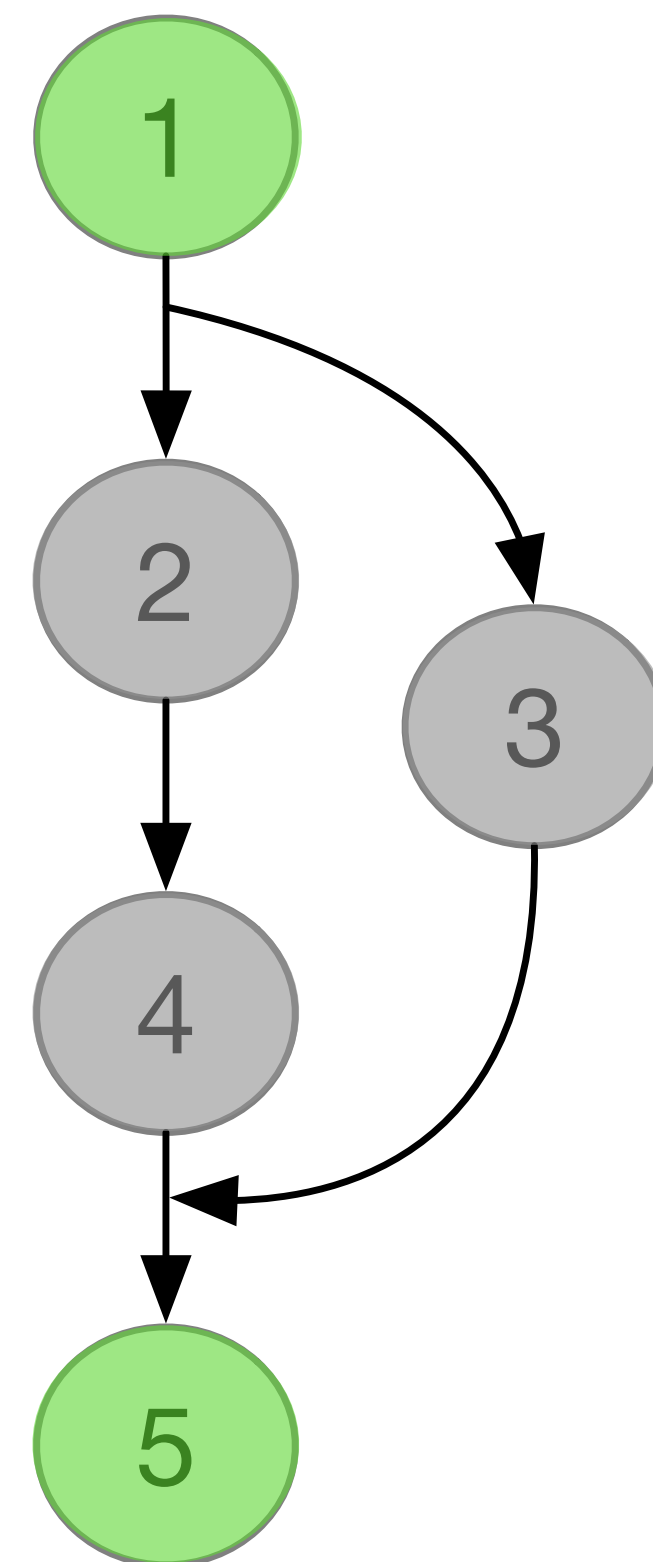
Ramp Architecture



Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

Ramp Architecture



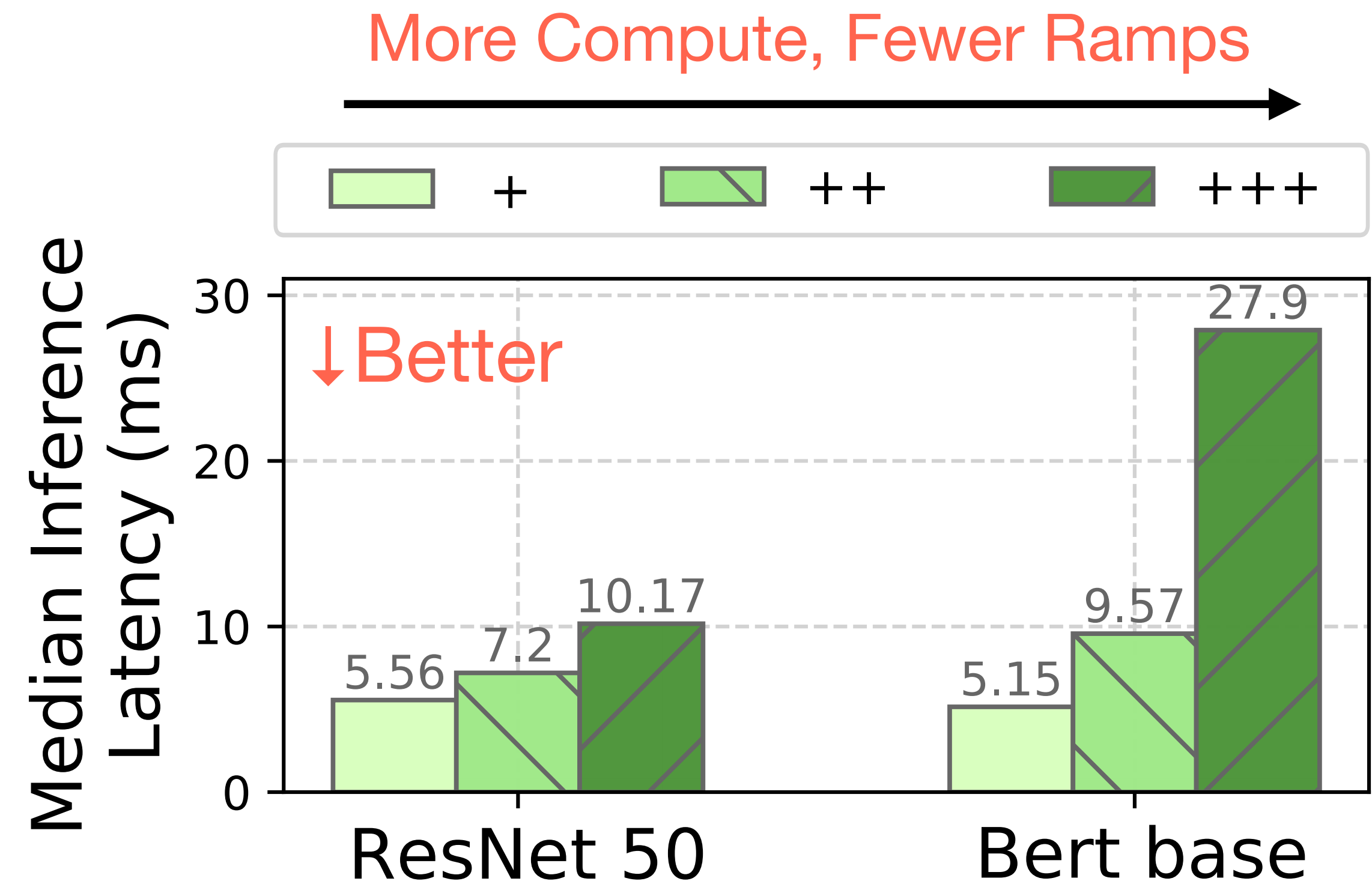
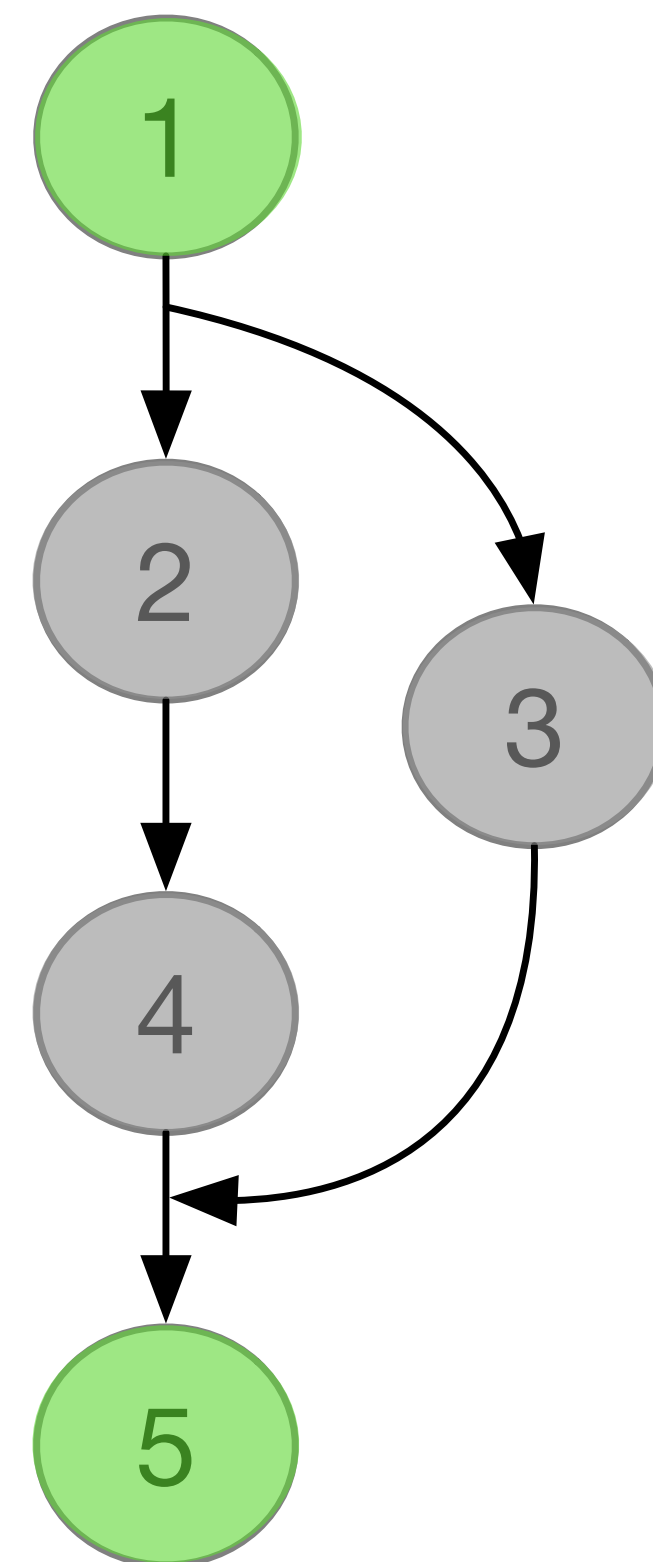
Coverage matters more than complexity!

Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

Ramp Architecture

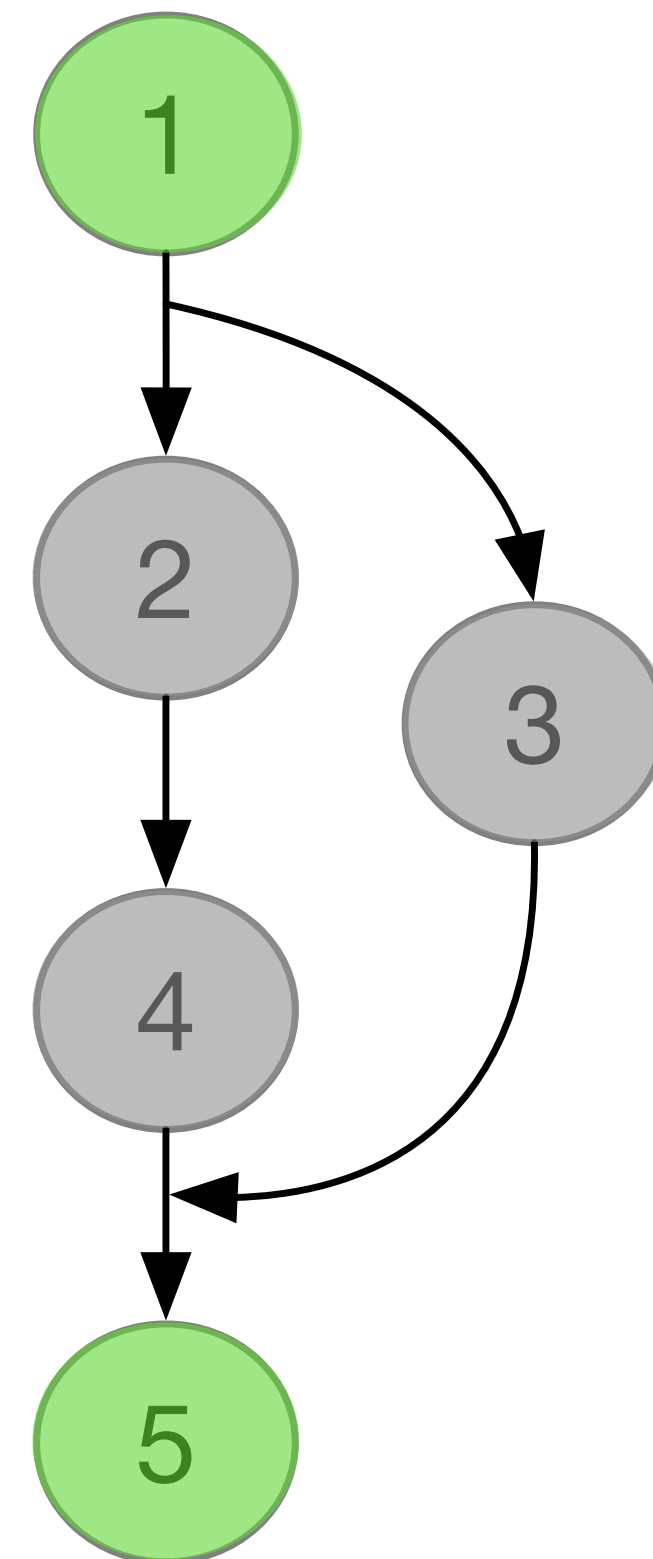
- cheapest ramps



Coverage matters more than complexity!

Apparate: Automatic integration for EEs

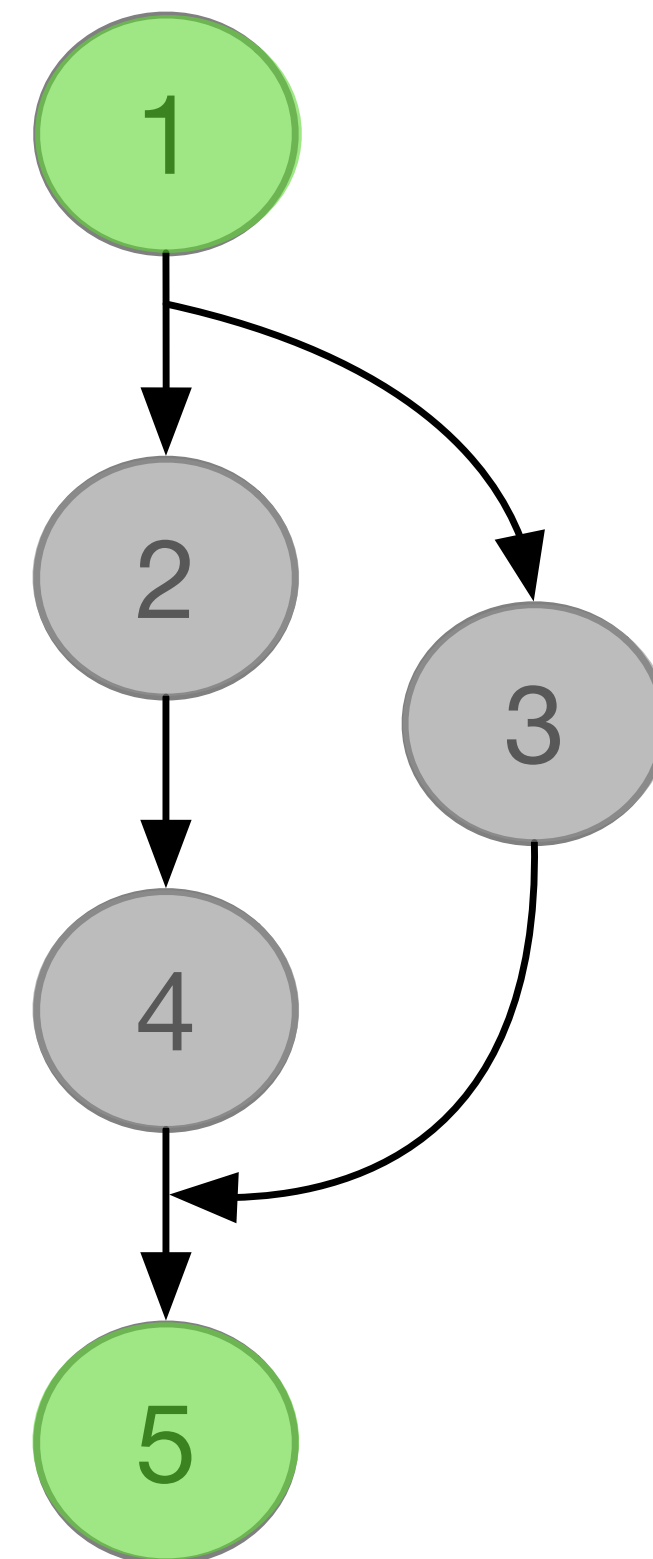
challenge: ramp location, architecture, and weights



Ramp weights

Apparate: Automatic integration for EEs

challenge: ramp location, architecture, and weights

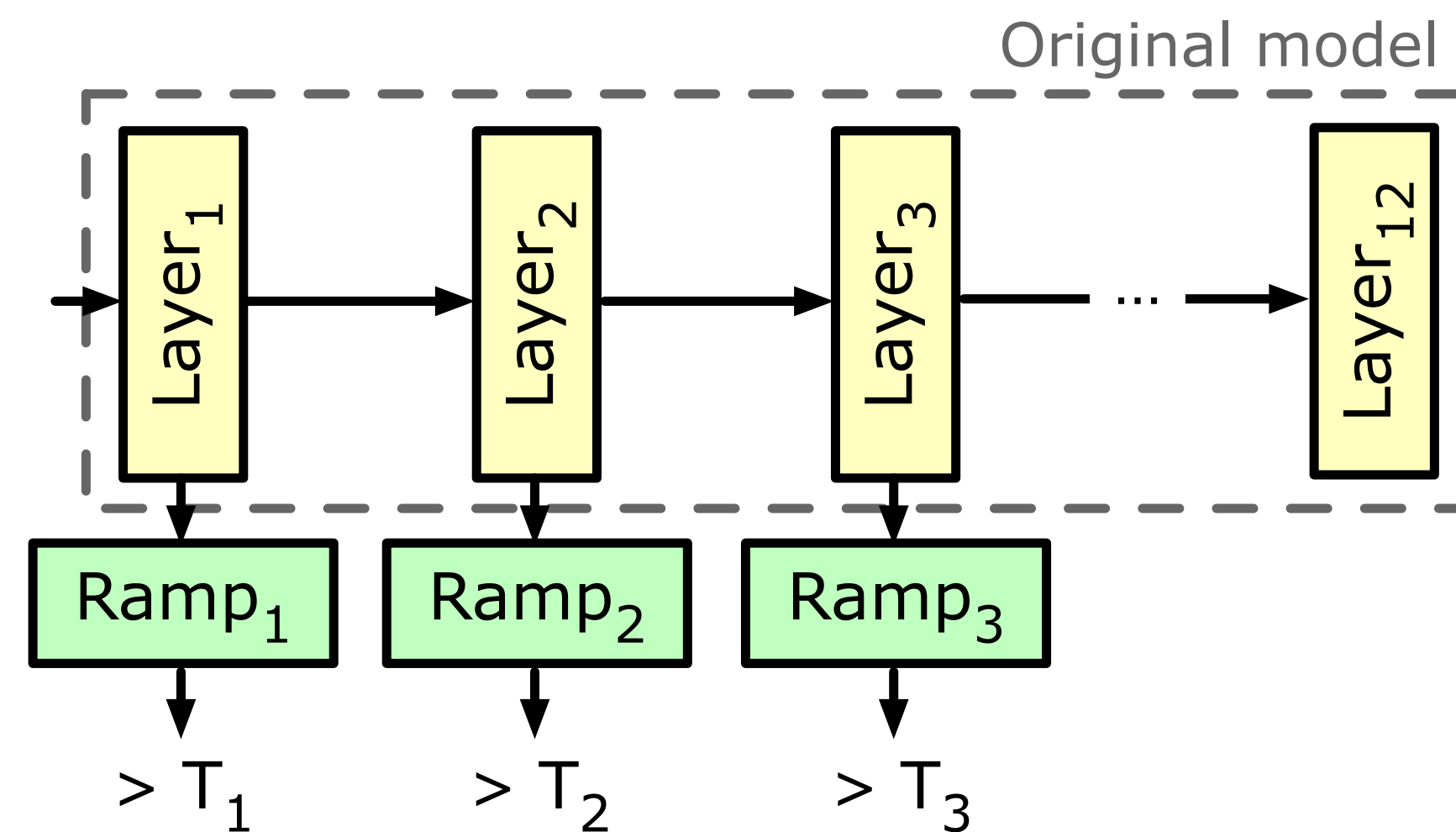


Ramp weights

- parallel training

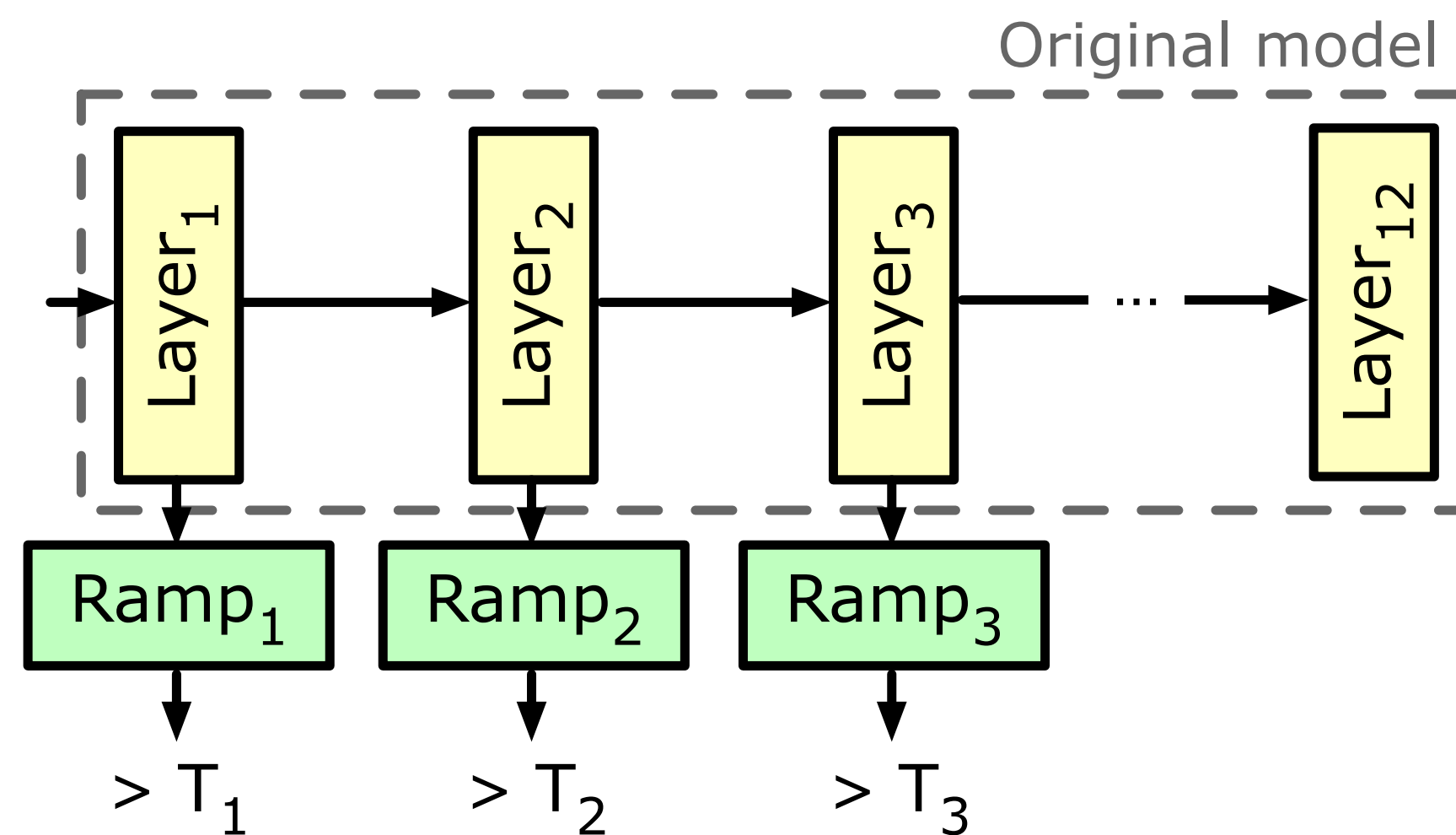
Apparate: Efficient Runtime Adaptation for EEs

challenge: massive space of EE configurations



Apparate: Efficient Runtime Adaptation for EEs

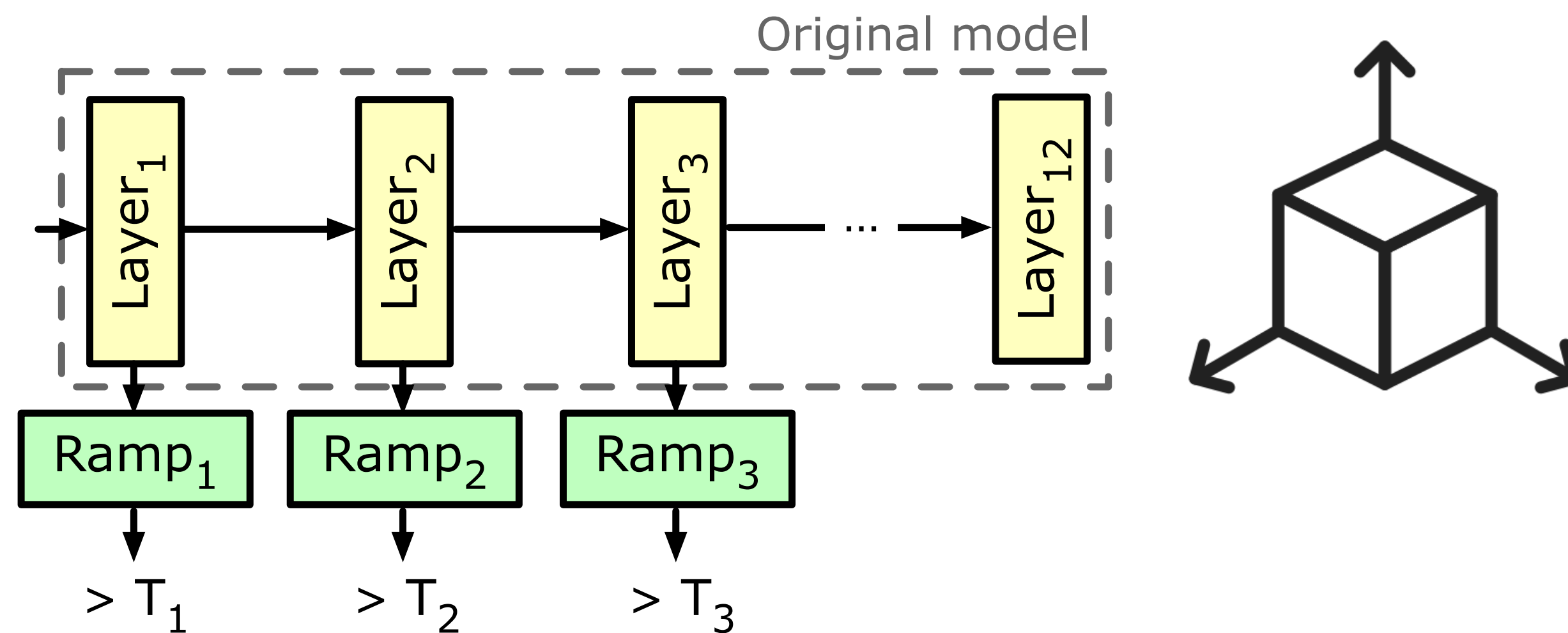
challenge: massive space of EE configurations



Minimizes latency while bounds accuracy loss (<1%) and ramp overhead (< 2%)

Apparate: Efficient Runtime Adaptation for EEs

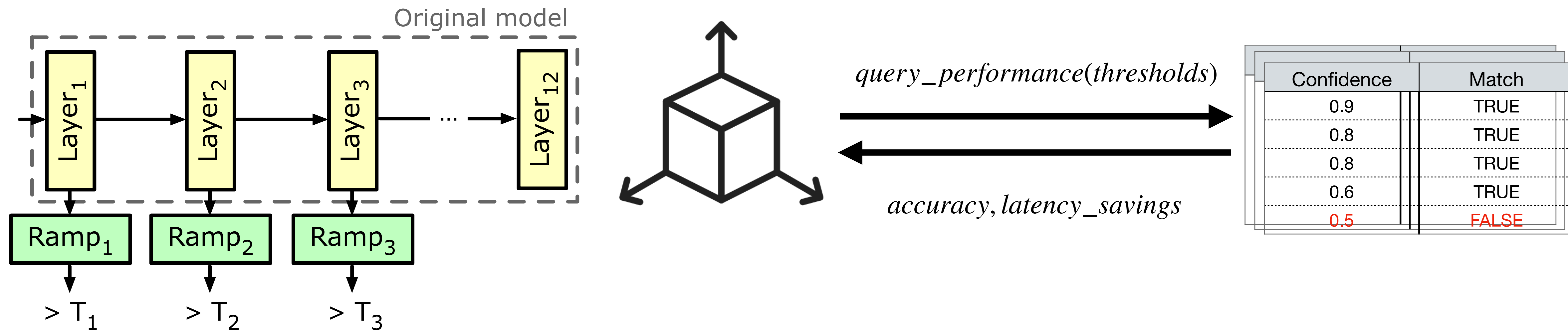
challenge: massive space of EE configurations



Minimizes latency while bounds accuracy loss ($< 1\%$) and ramp overhead ($< 2\%$)

Apparate: Efficient Runtime Adaptation for EEs

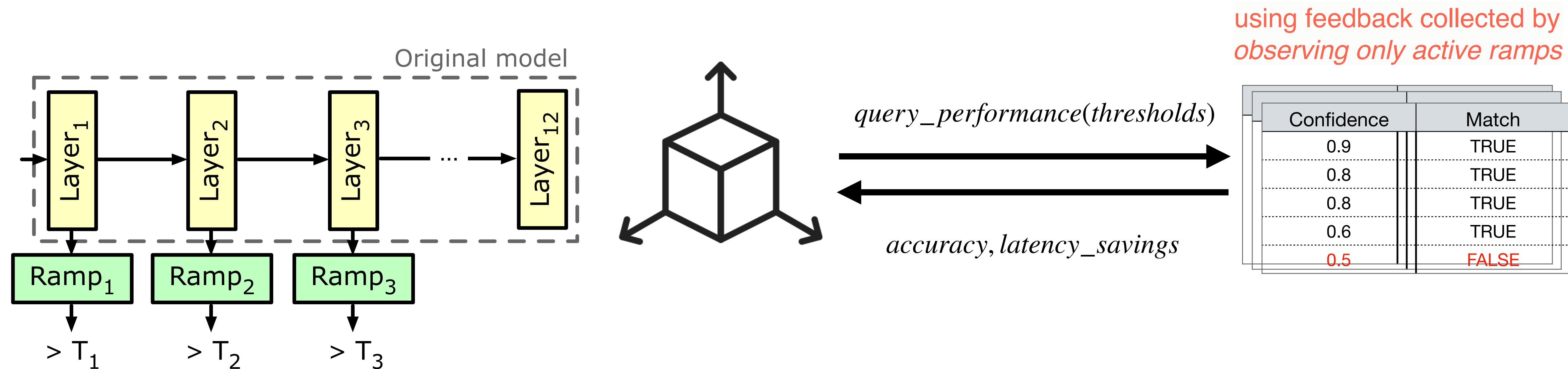
challenge: massive space of EE configurations



Minimizes latency while bounds accuracy loss (<1%) and ramp overhead (< 2%)

Apparate: Efficient Runtime Adaptation for EEs

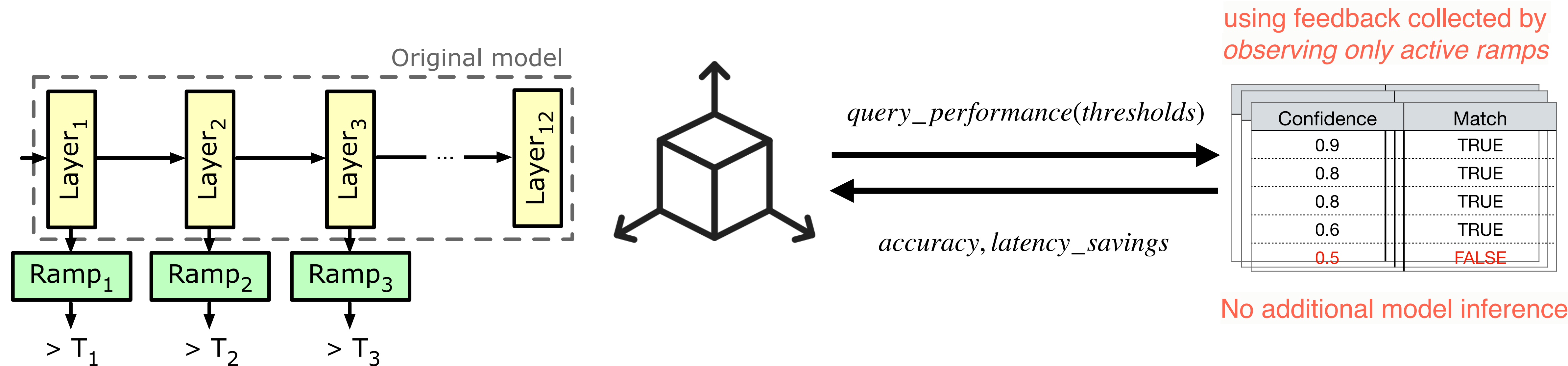
challenge: massive space of EE configurations



Minimizes latency while bounds accuracy loss (<1%) and ramp overhead (< 2%)

Apparate: Efficient Runtime Adaptation for EEs

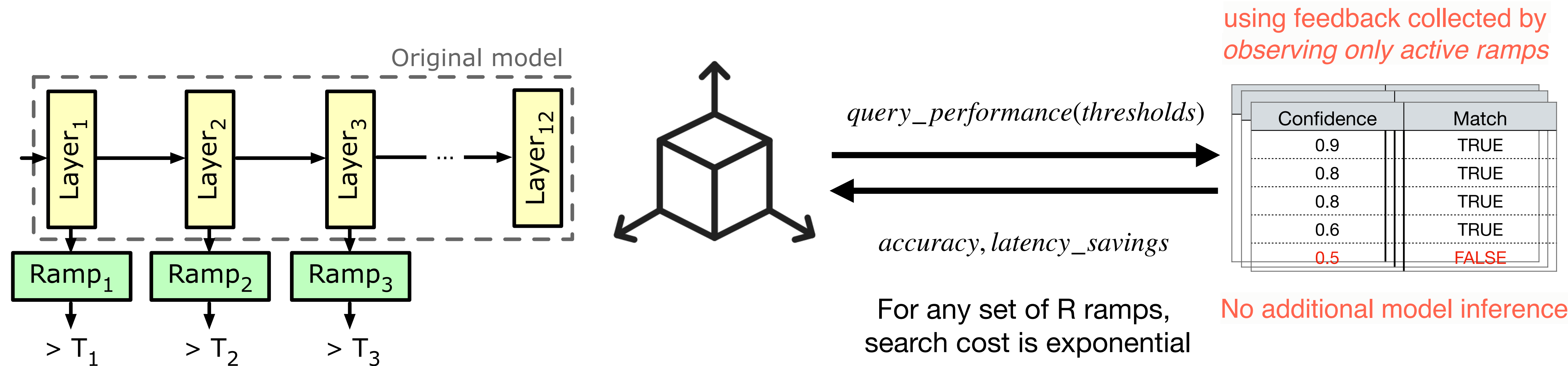
challenge: massive space of EE configurations



Minimizes latency while bounds accuracy loss (<1%) and ramp overhead (< 2%)

Apparate: Efficient Runtime Adaptation for EEs

challenge: massive space of EE configurations



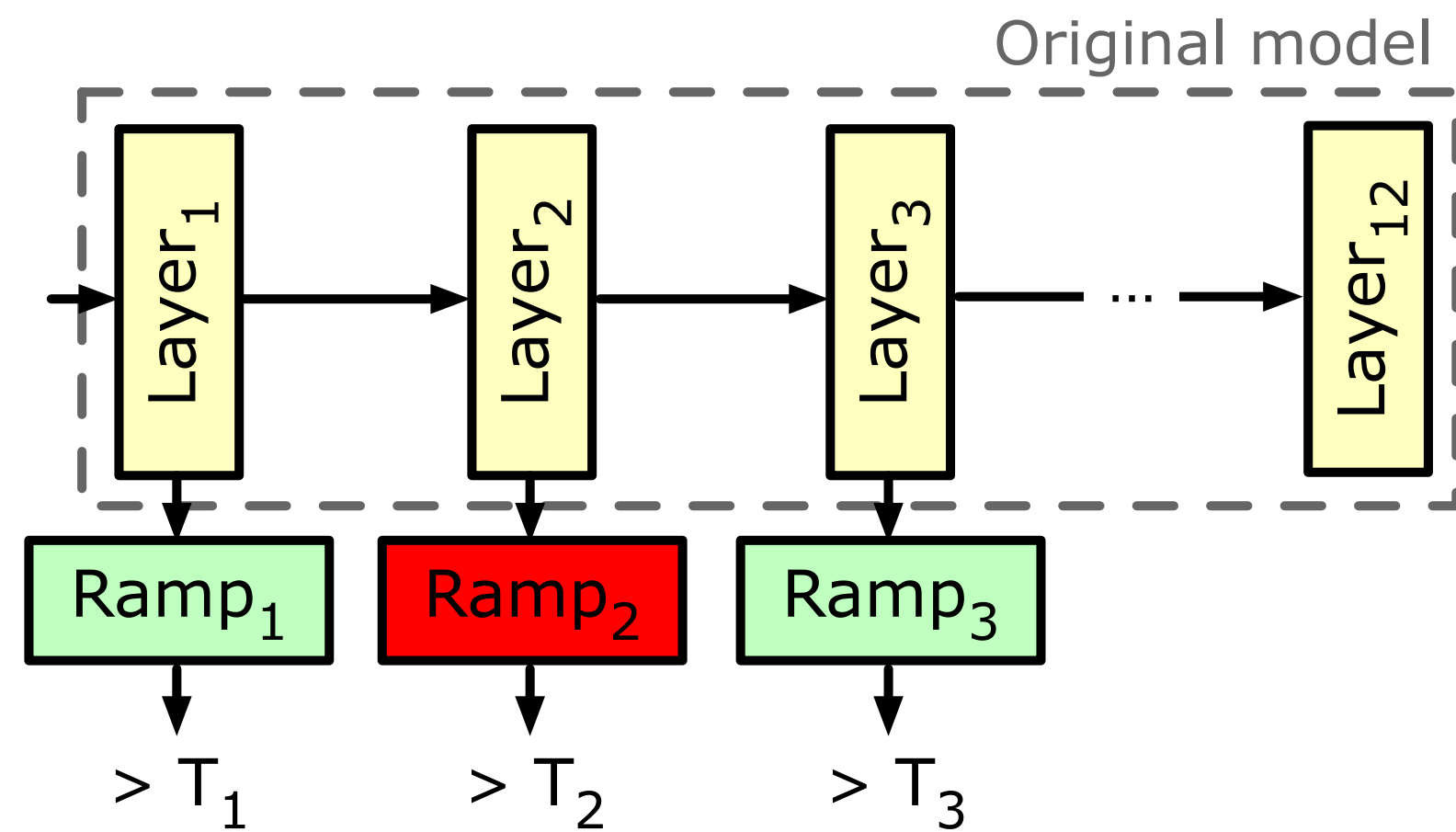
Minimizes latency while bounds accuracy loss (<1%) and ramp overhead (< 2%)

Apparate: Efficient Runtime Adaptation for EEs

insight 1: decouple tunable EE knobs

Apparate: Efficient Runtime Adaptation for EEs

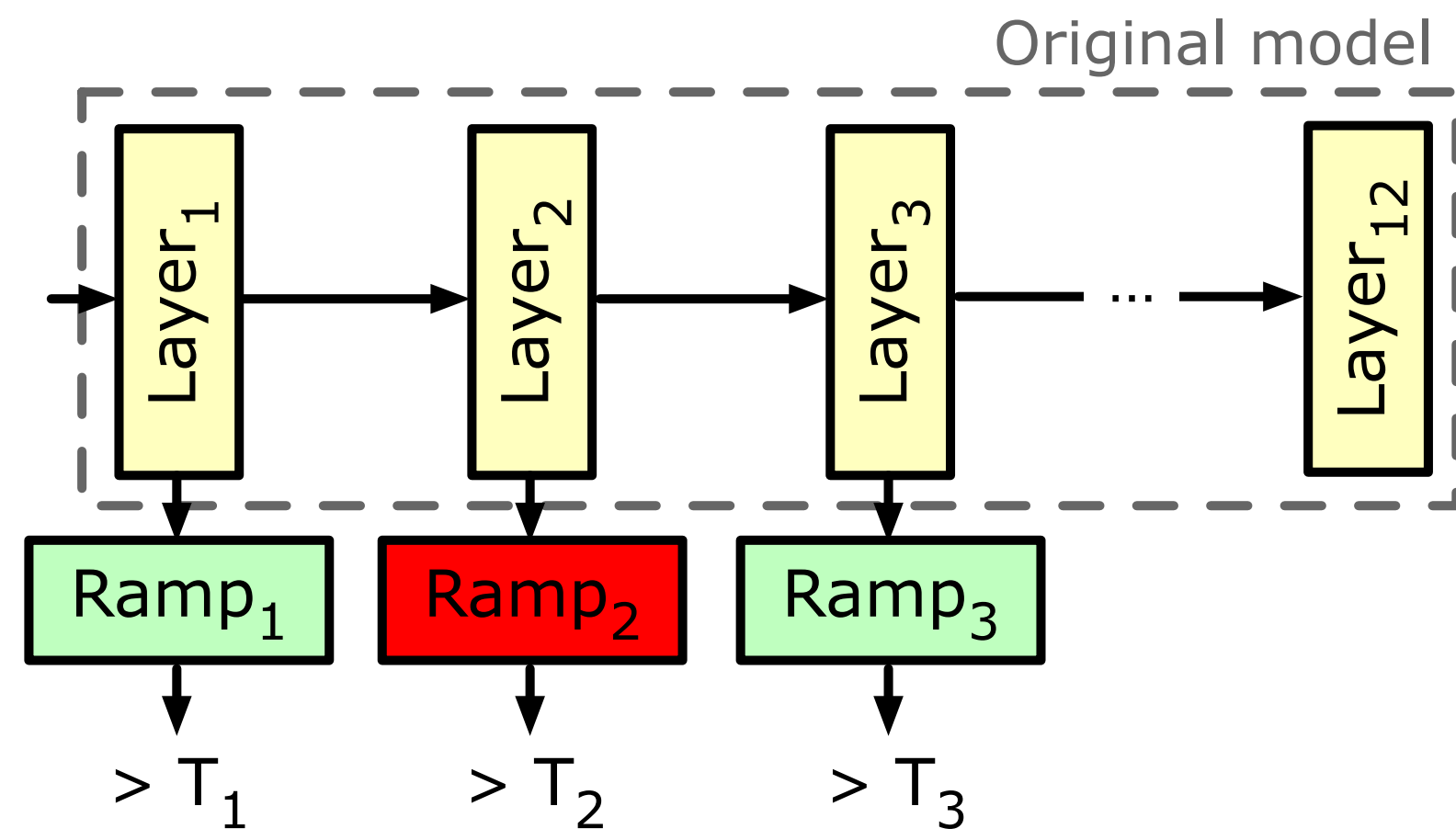
insight 1: decouple tunable EE knobs



- Threshold Tuning:
 - Cheap and controls accuracy

Apparate: Efficient Runtime Adaptation for EEs

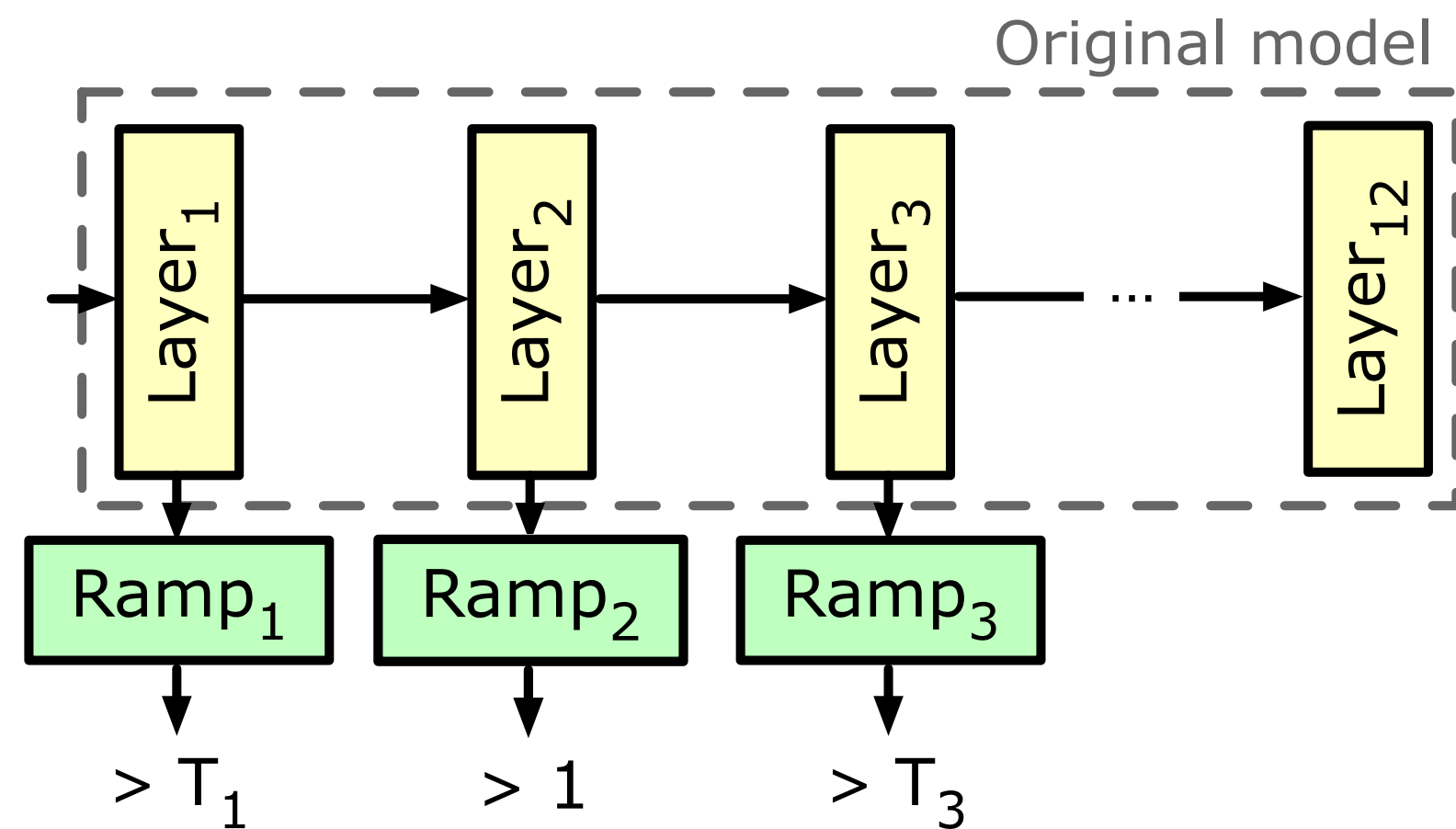
insight 1: decouple tunable EE knobs



- Threshold Tuning:
 - Cheap and controls accuracy
 - Adjusts immediately to bound **accuracy loss**

Apparate: Efficient Runtime Adaptation for EEs

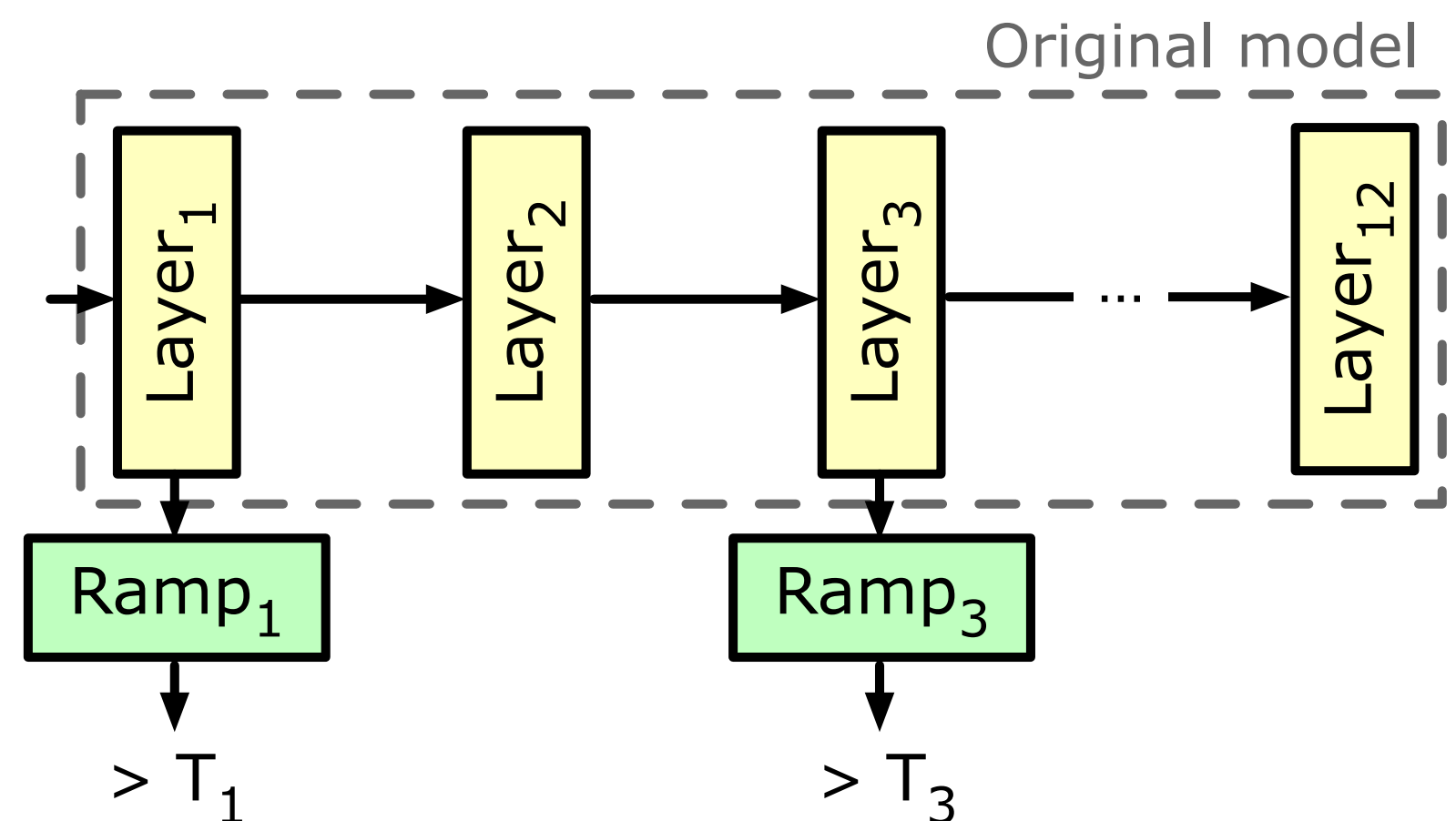
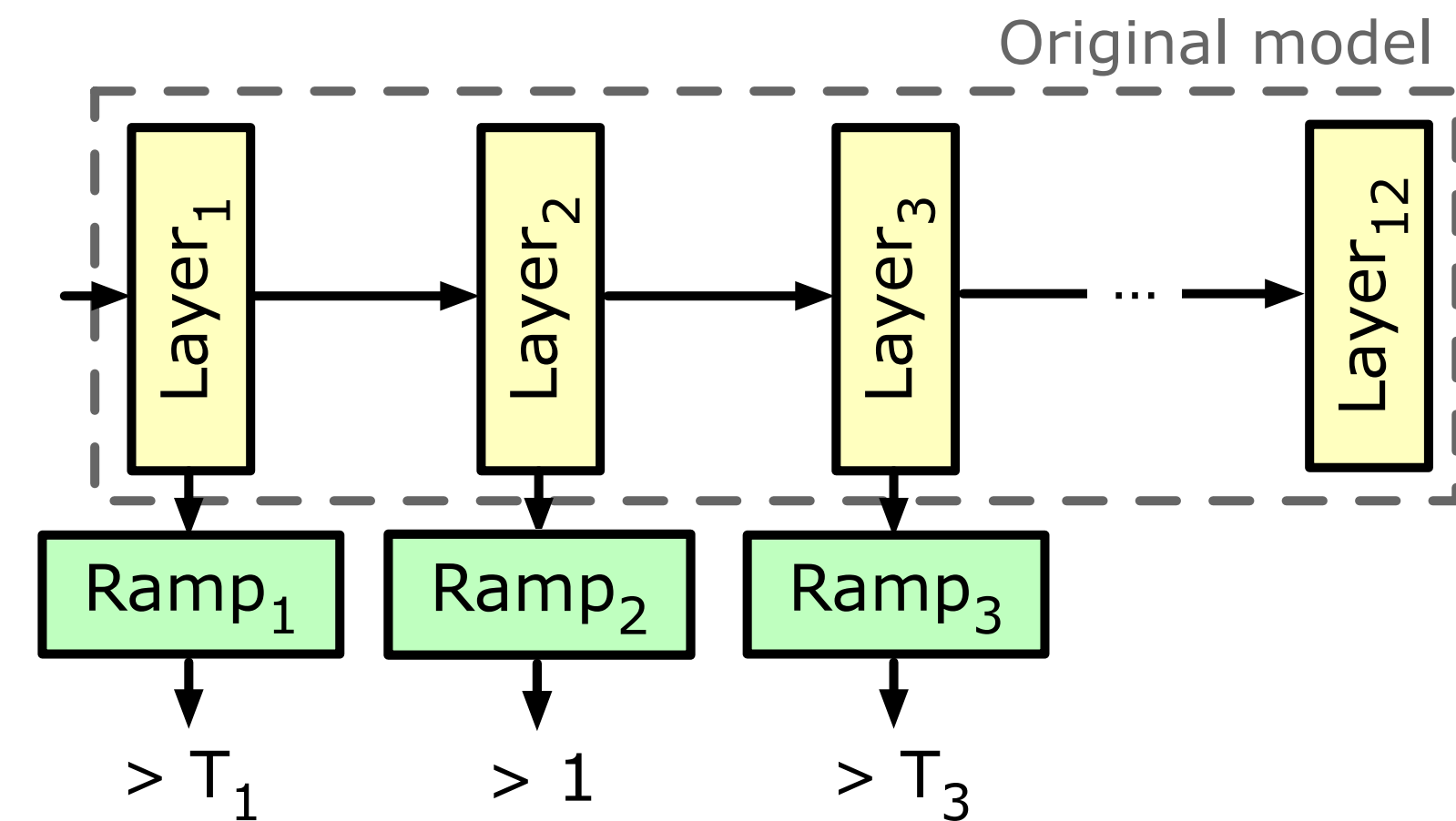
insight 1: decouple tunable EE knobs



- Threshold Tuning:
 - Cheap and controls accuracy
 - Adjusts immediately to bound **accuracy loss**

Apparate: Efficient Runtime Adaptation for EEs

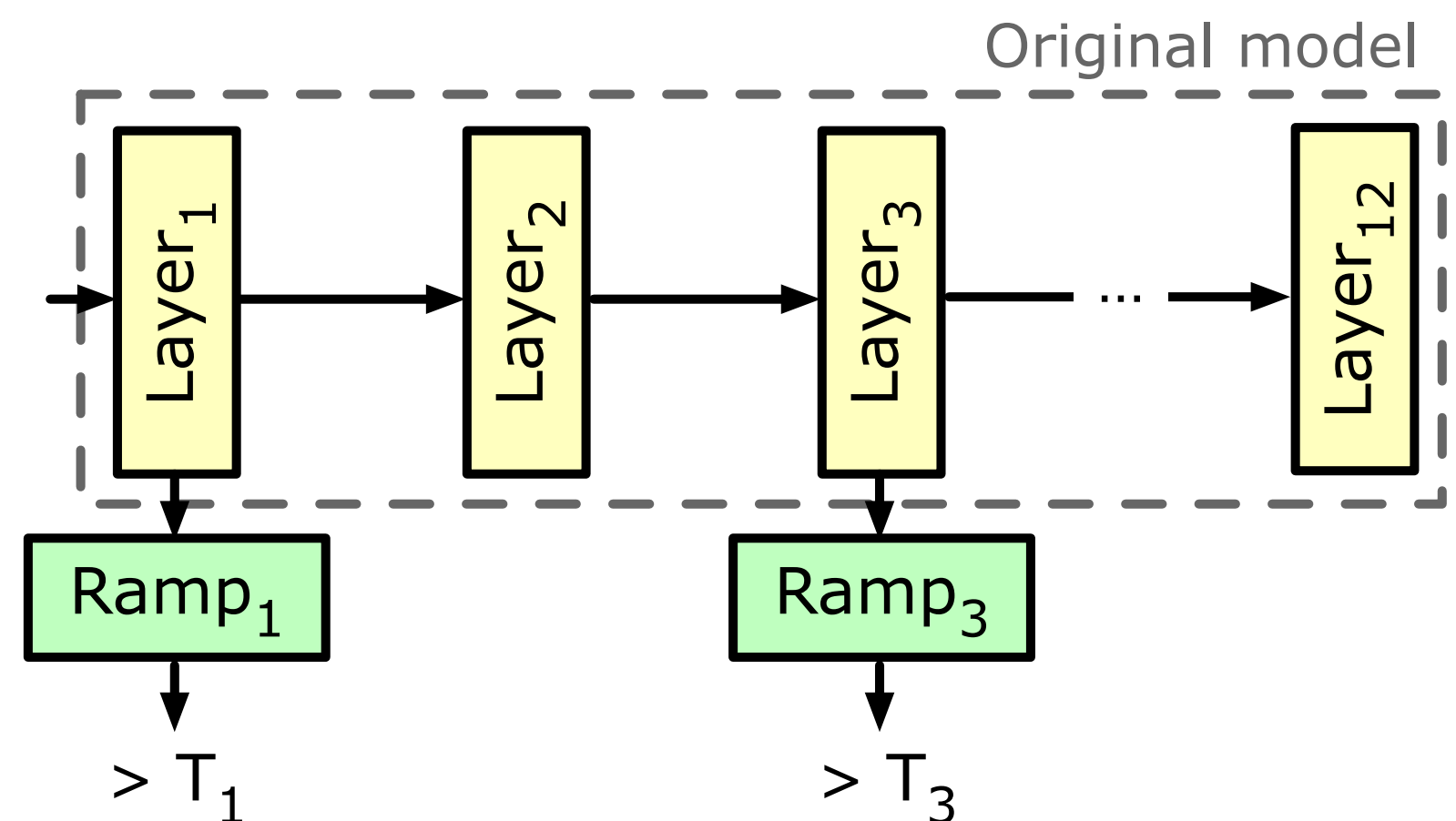
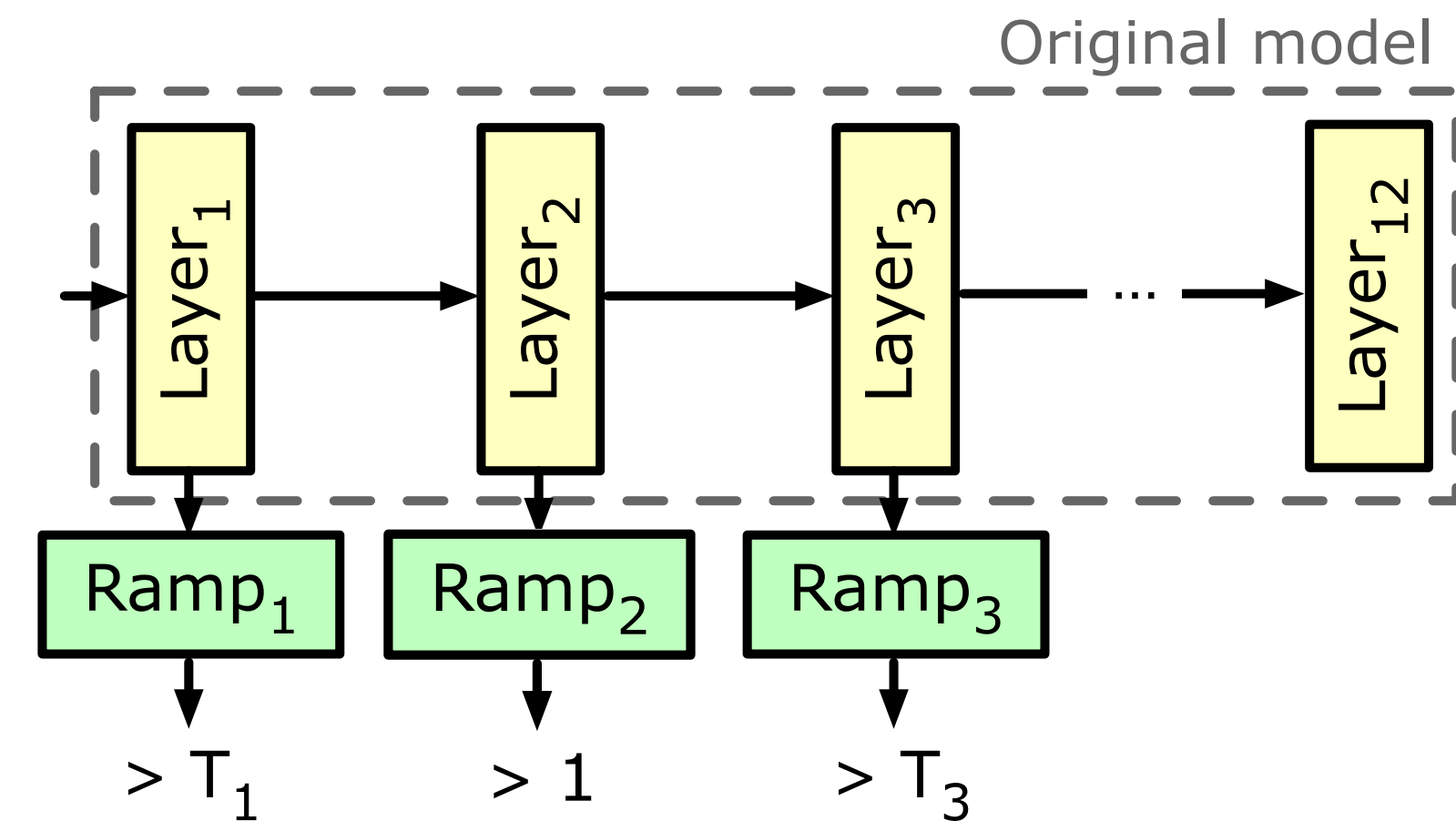
insight 1: decouple tunable EE knobs



- Threshold Tuning:
 - Cheap and controls accuracy
 - Adjusts immediately to bound **accuracy loss**
- Ramp Tuning:
 - Expensive and bounds latency
 - Adjusts periodically to optimize **latency savings**

Apparate: Efficient Runtime Adaptation for EEs

insight 1: decouple tunable EE knobs



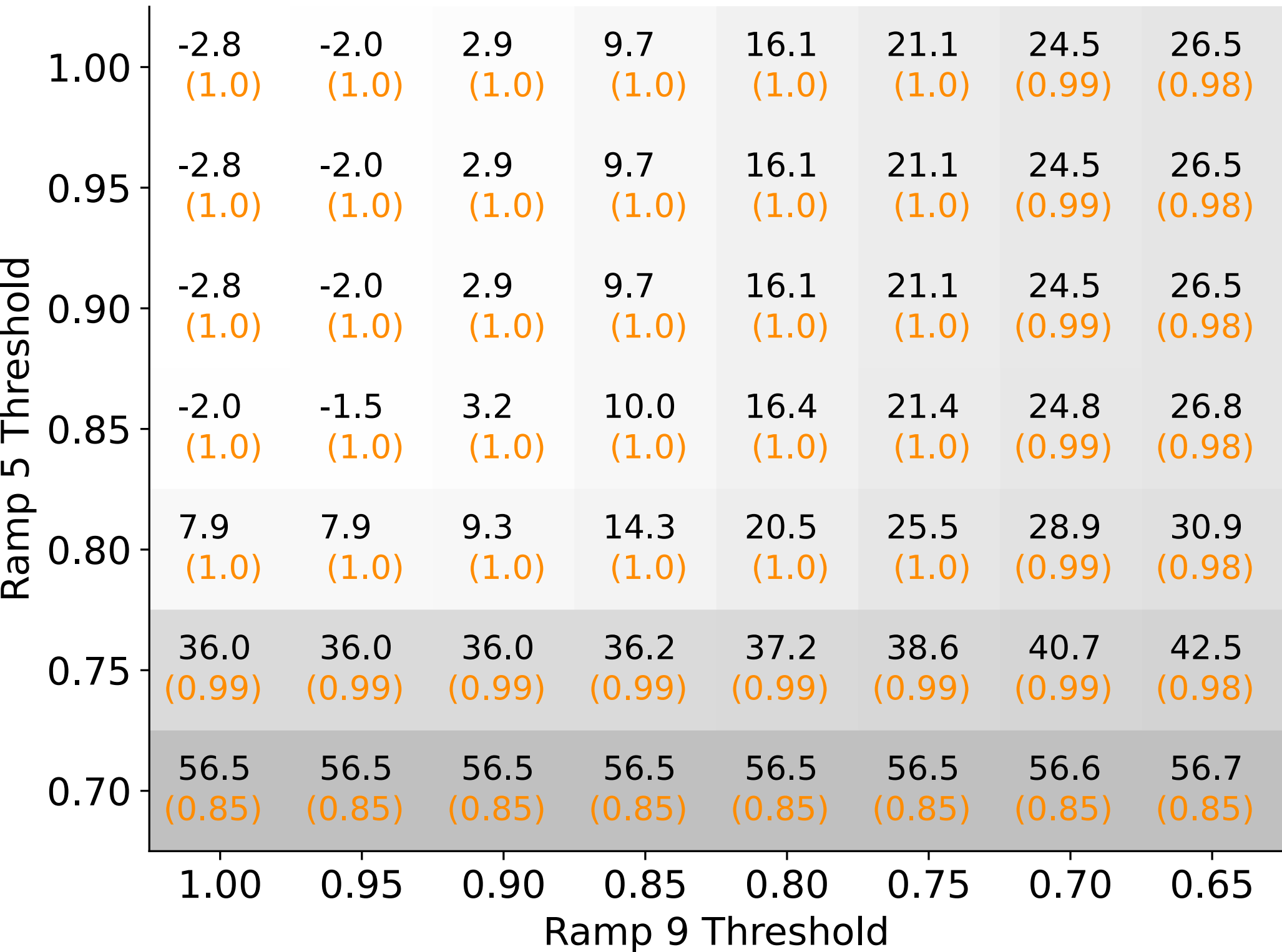
- Threshold Tuning:
 - Cheap and controls accuracy
 - Adjusts immediately to bound **accuracy loss**
- Ramp Tuning:
 - Expensive and bounds latency
 - Adjusts periodically to optimize **latency savings**
 - Multiple lightweight ramps reduce the impact of missing **highly effective ramps**

Apparate: Efficient Runtime Adaptation for EEs

insight 2: leverage EE properties for fast tuning

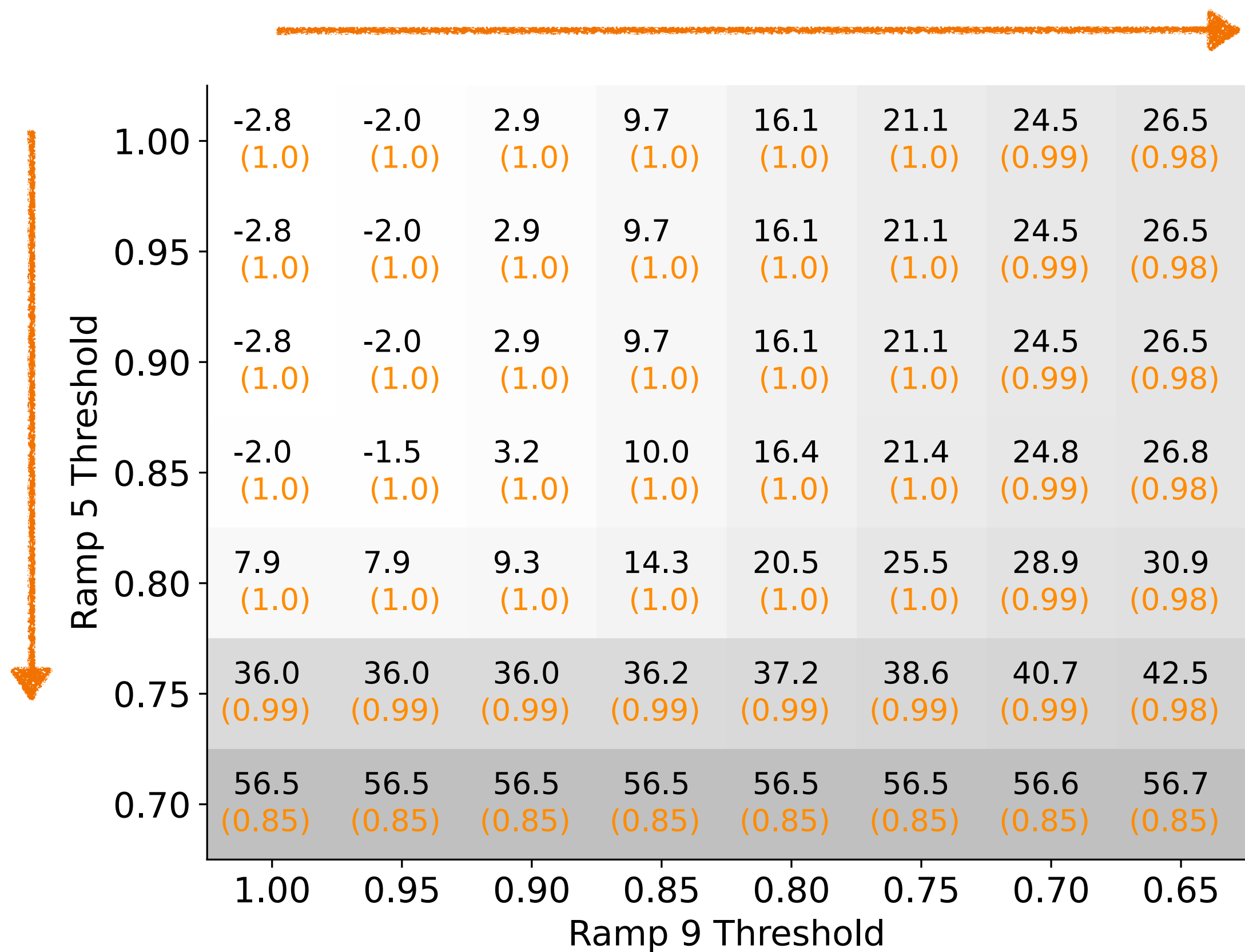
Apparate: Efficient Runtime Adaptation for EEs

insight 2: leverage EE properties for fast tuning



Apparate: Efficient Runtime Adaptation for EEs

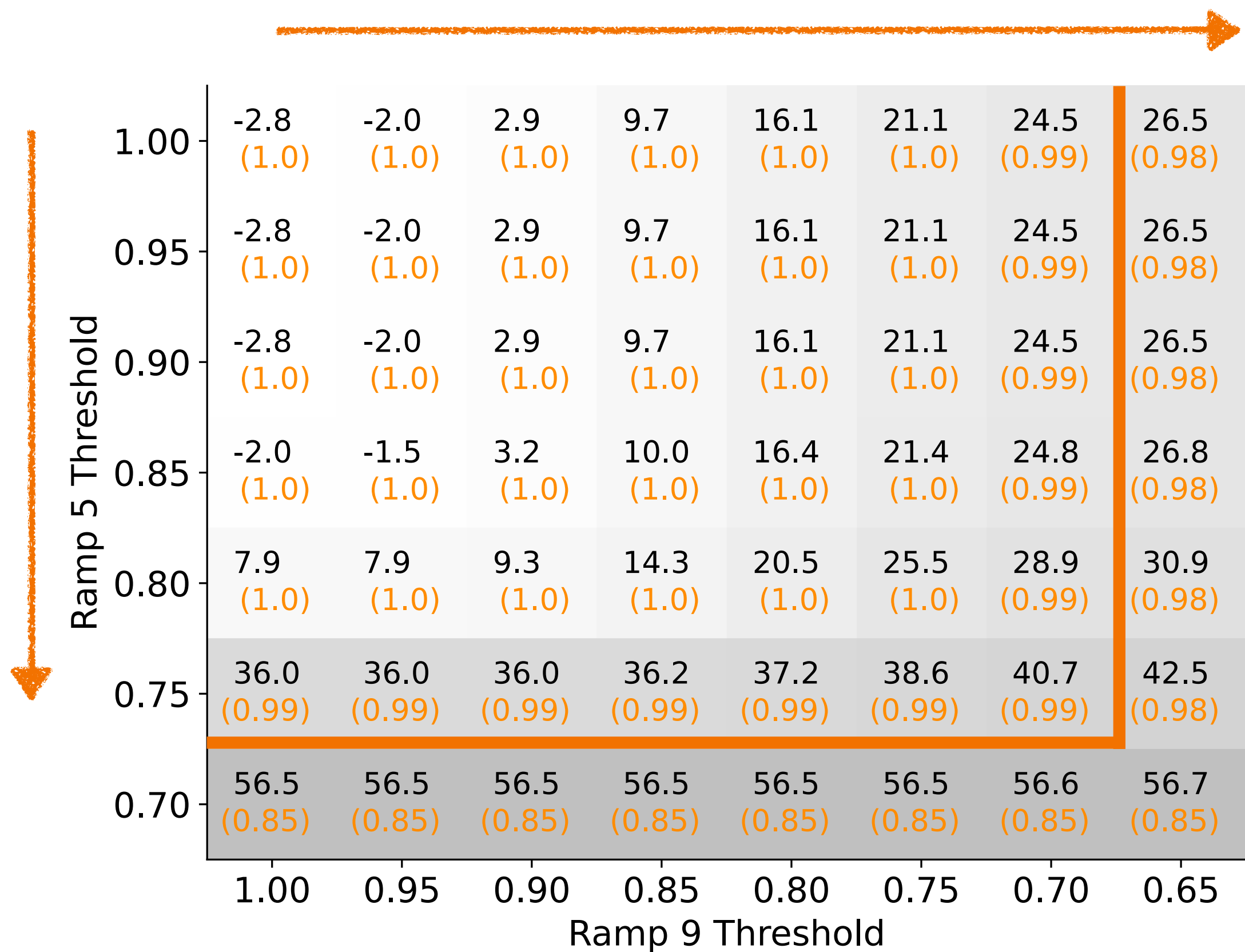
insight 2: leverage EE properties for fast tuning



Lower thresholds indicate higher latency savings and lower accuracy

Apparate: Efficient Runtime Adaptation for EEs

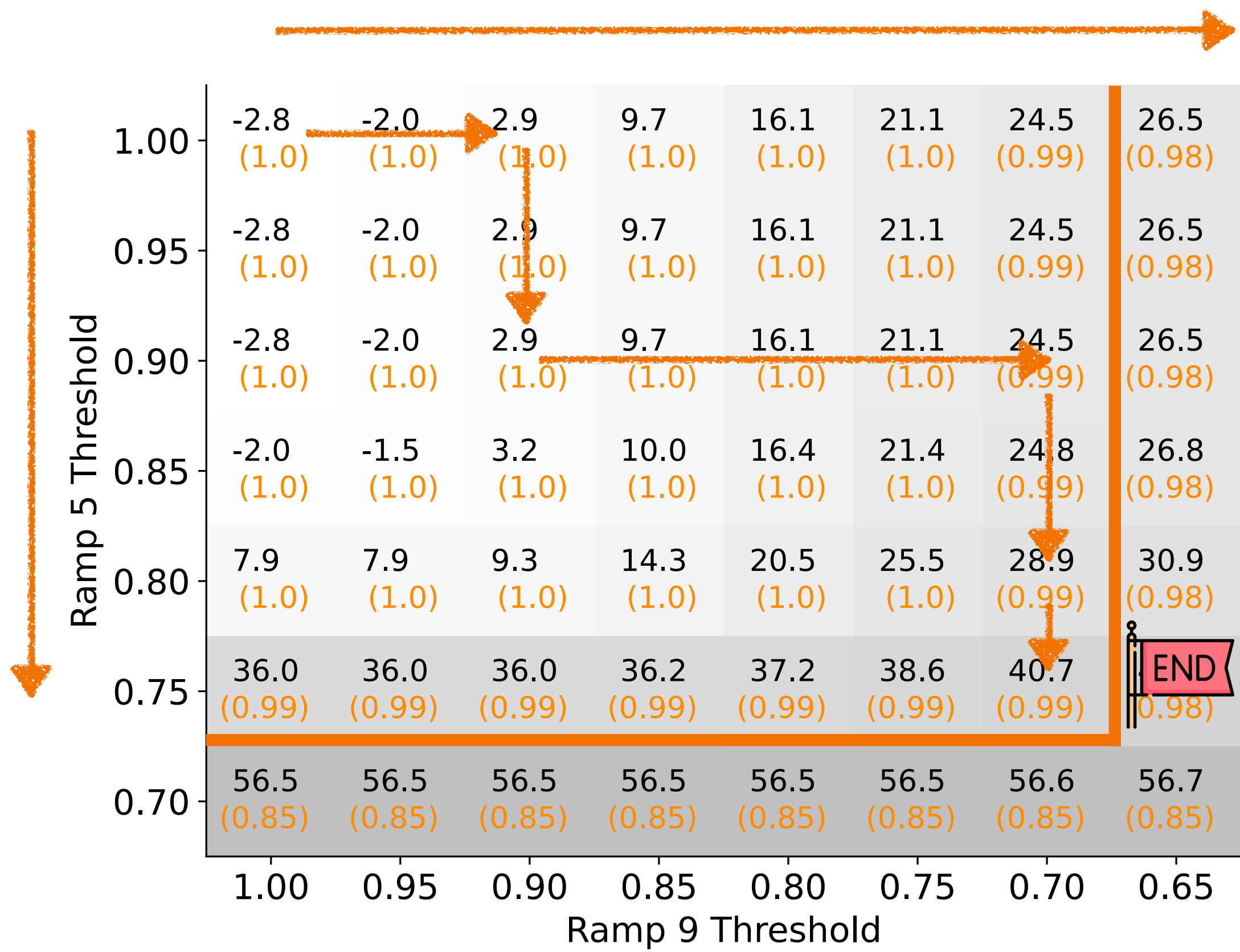
insight 2: leverage EE properties for fast tuning



Lower thresholds indicate higher latency savings and lower accuracy

Apparate: Efficient Runtime Adaptation for EEs

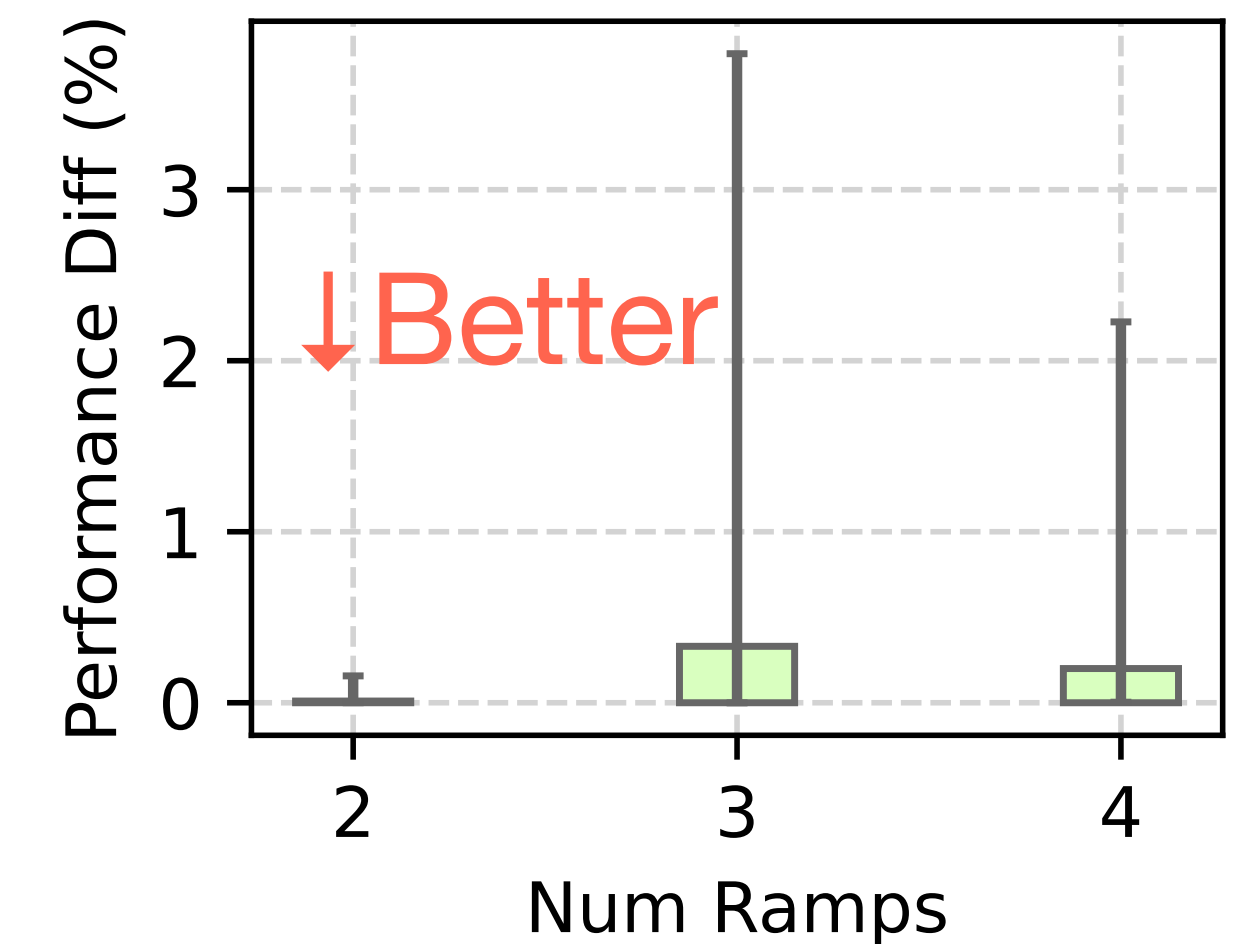
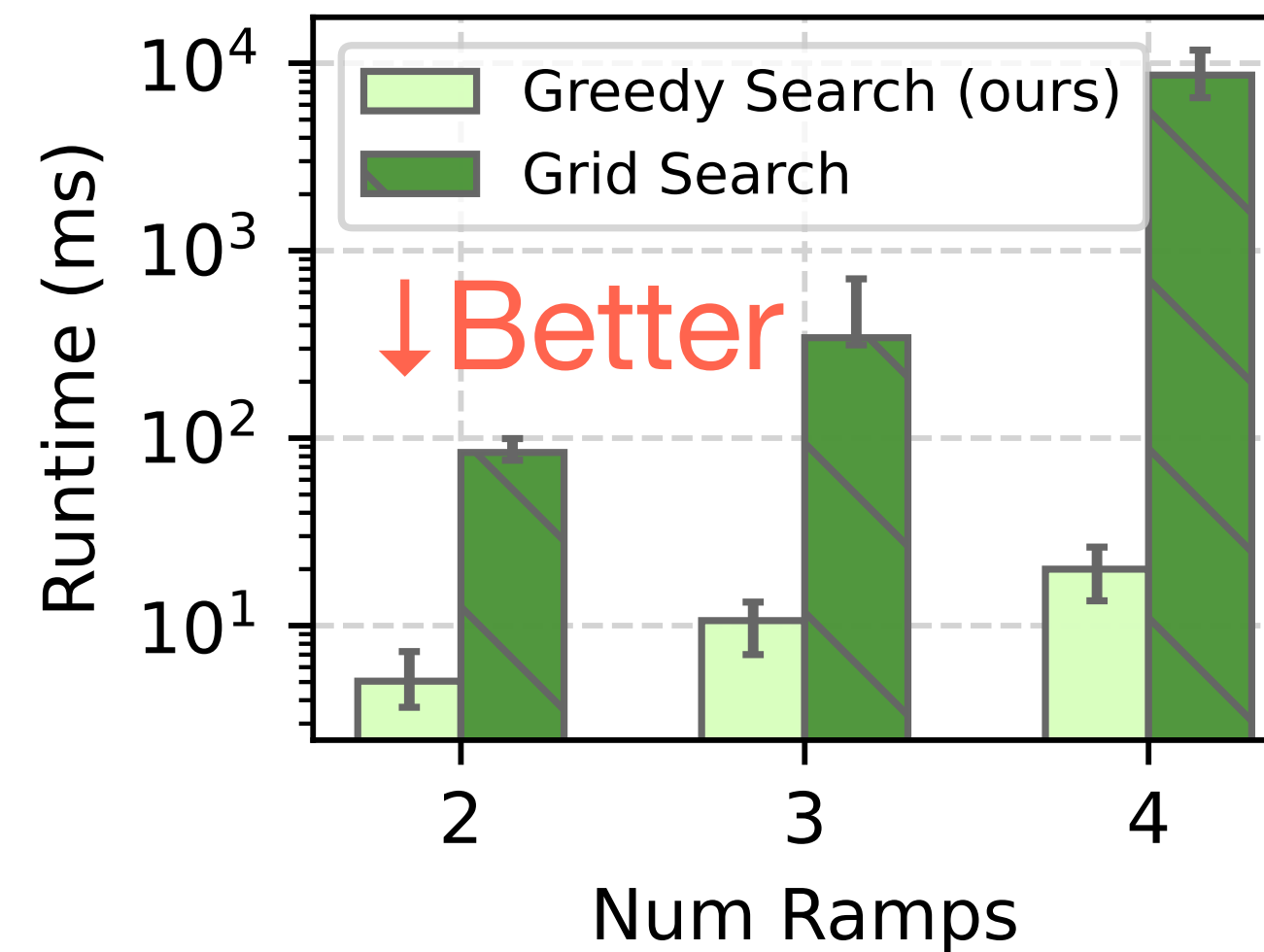
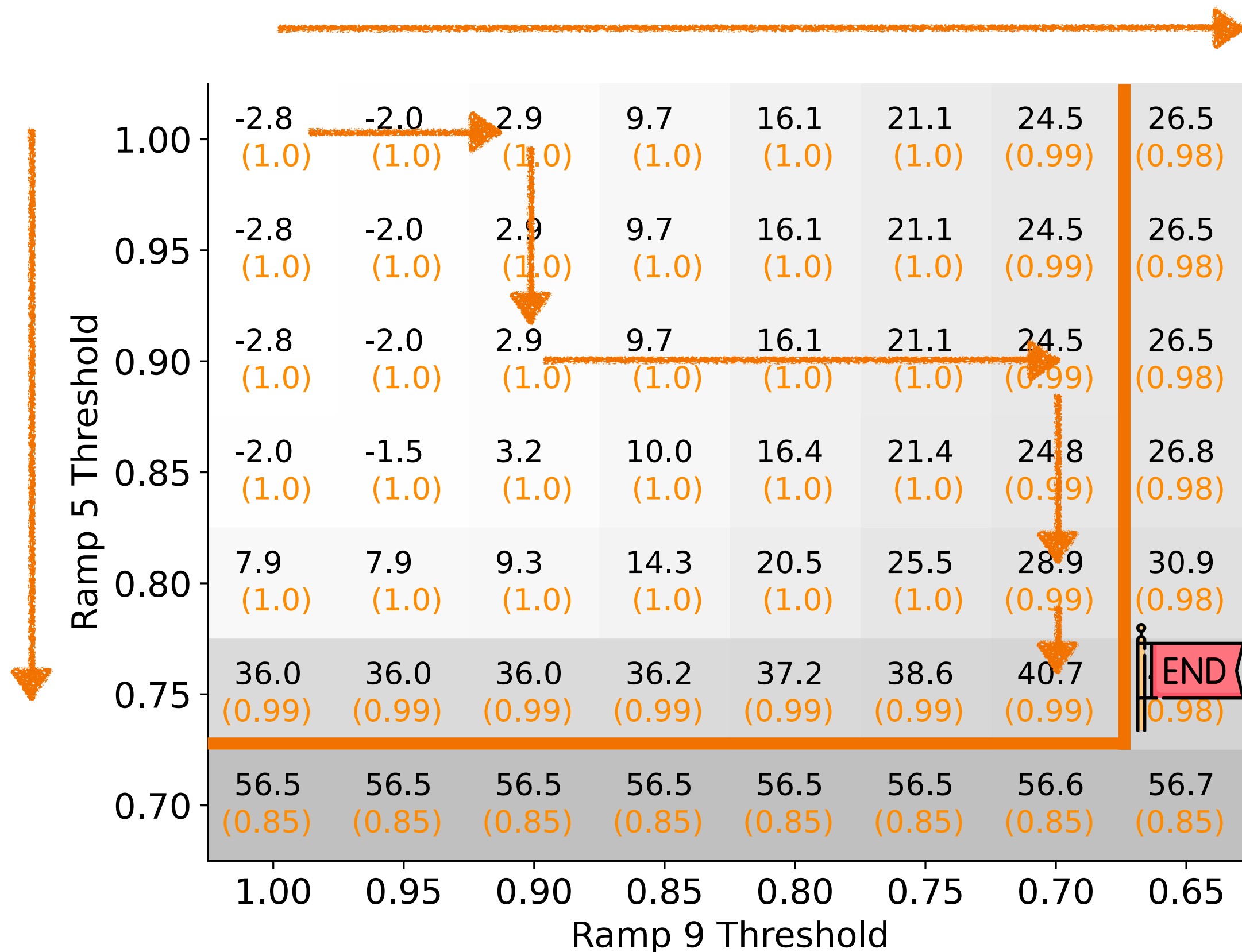
insight 2: leverage EE properties for fast tuning



Lower thresholds indicate higher latency savings and lower accuracy

Apparate: Efficient Runtime Adaptation for EEs

insight 2: leverage EE properties for fast tuning

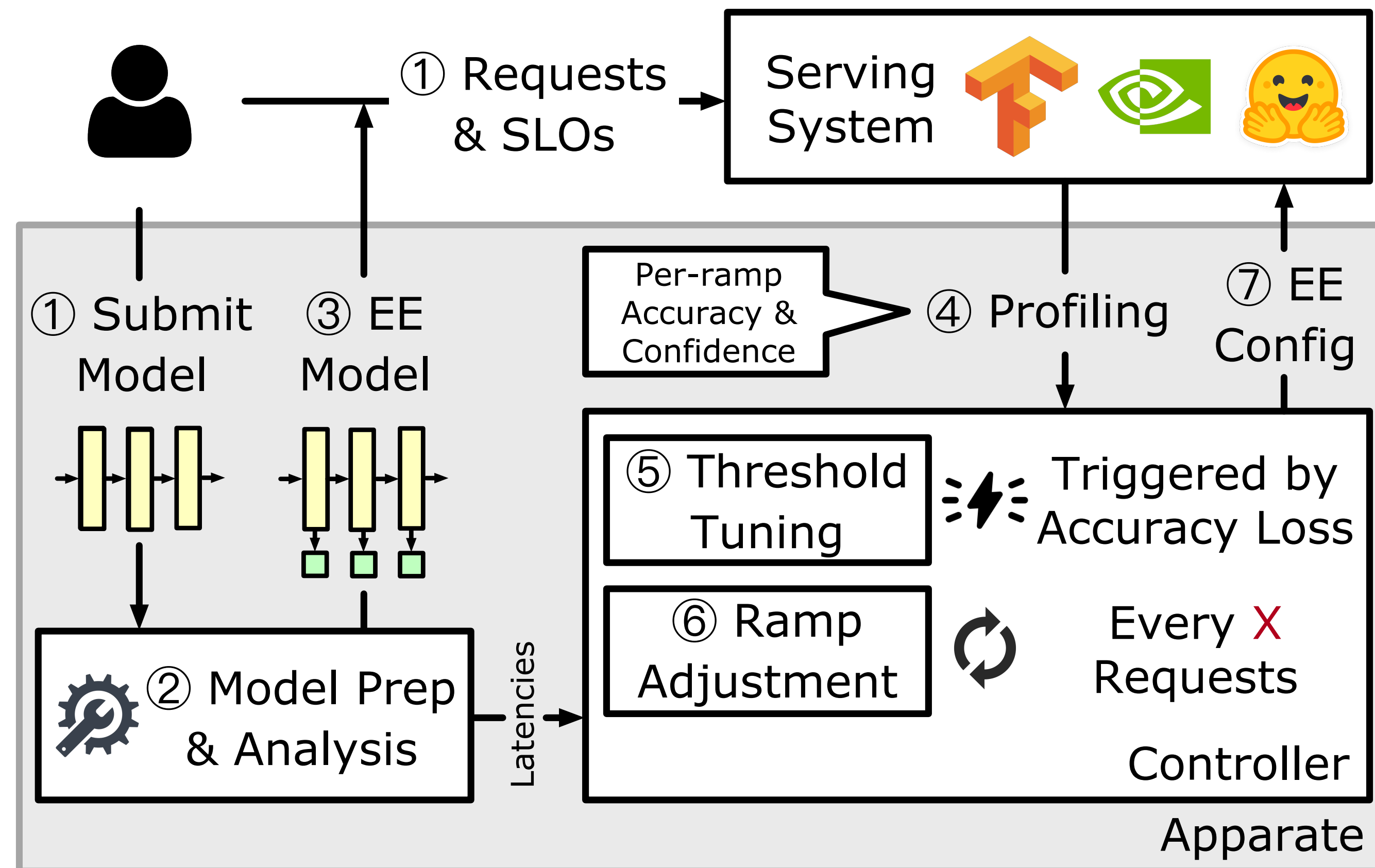


Threshold tuning is quick and achieves near-optimal latency savings

Lower thresholds indicate higher latency savings and lower accuracy

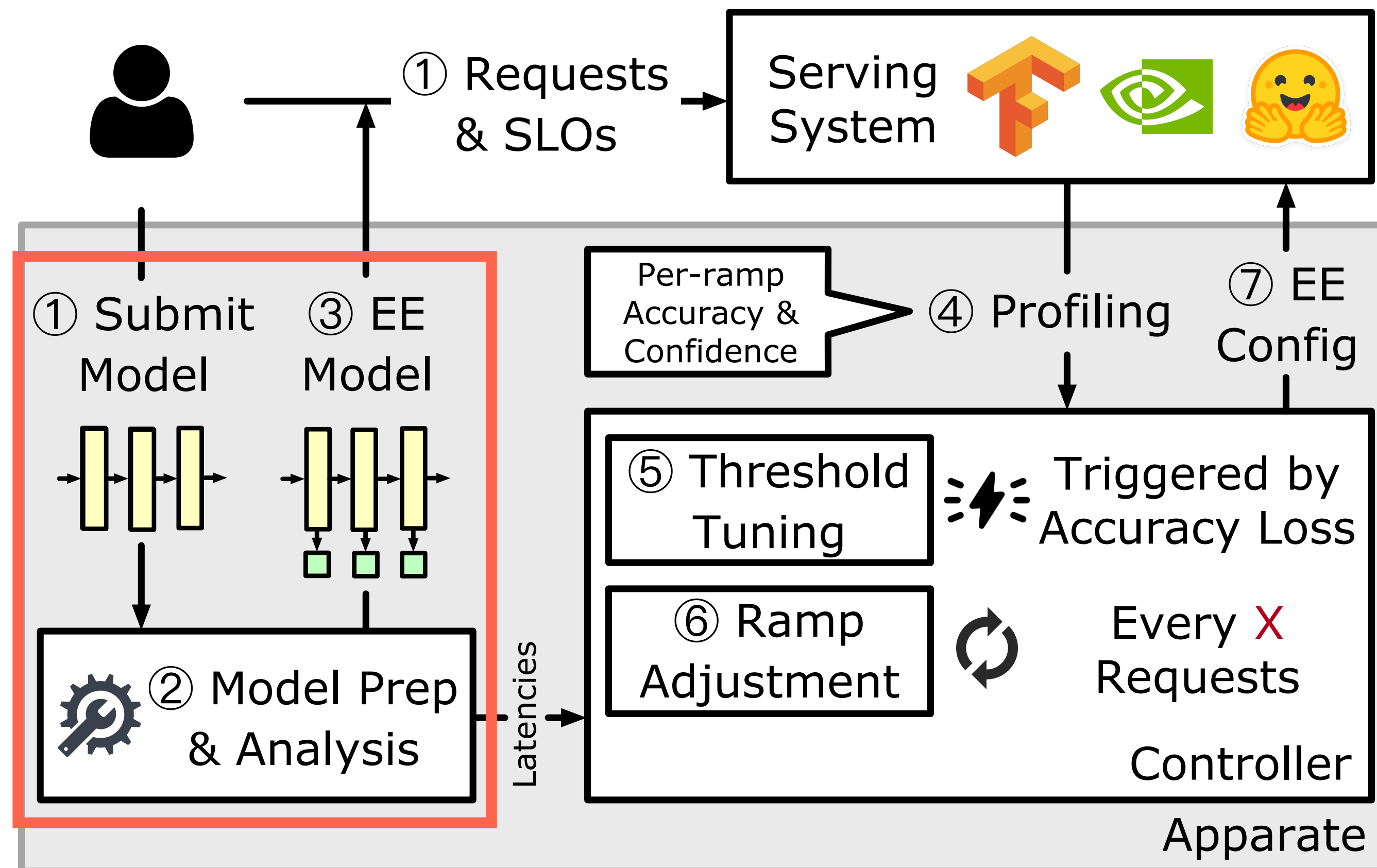
Apparate

the first system that automatically integrates and manages EEs for ML inference



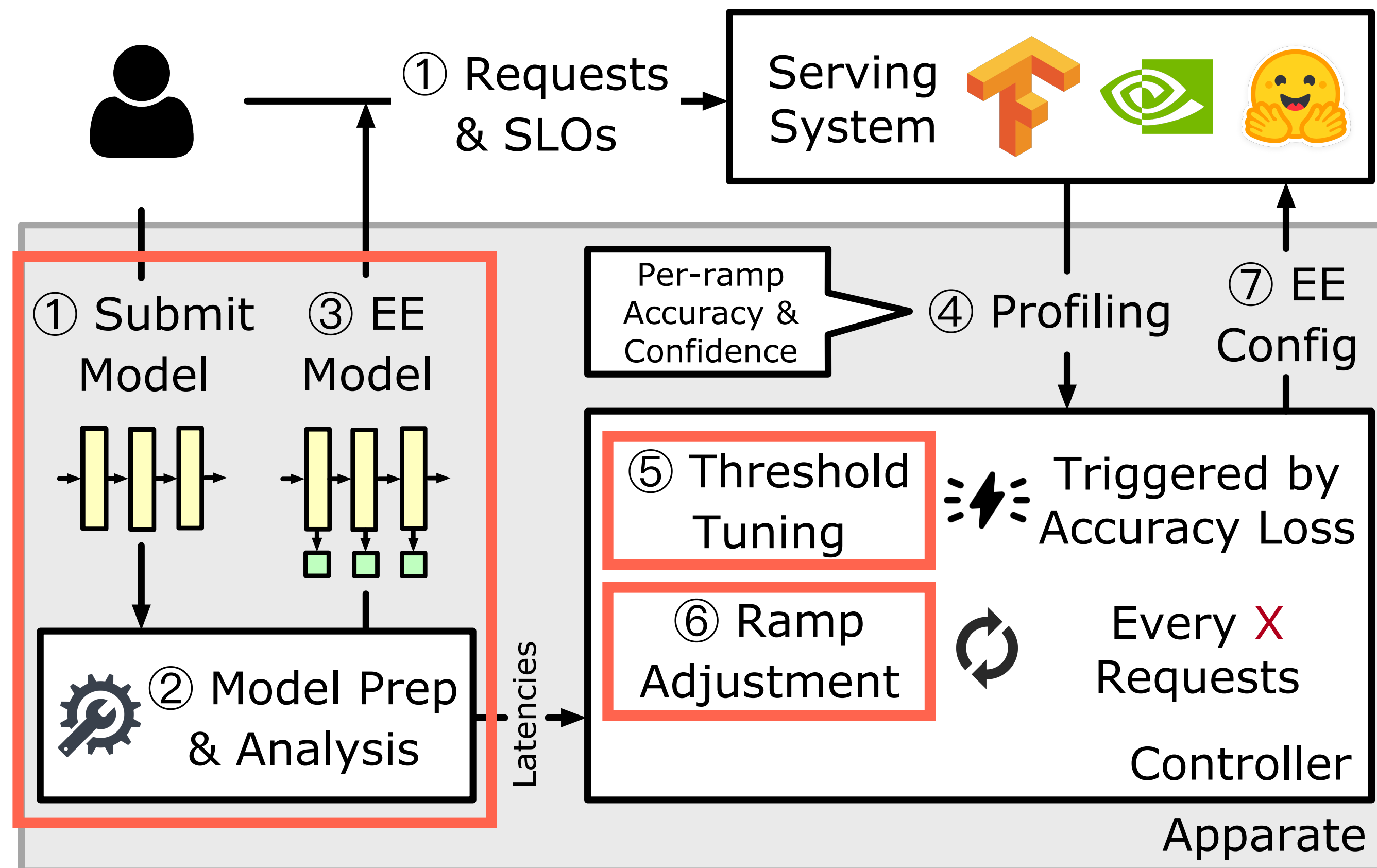
Apparate

the first system that automatically integrates and manages EEs for ML inference



Apparate

the first system that automatically integrates and manages EEs for ML inference



Evaluation

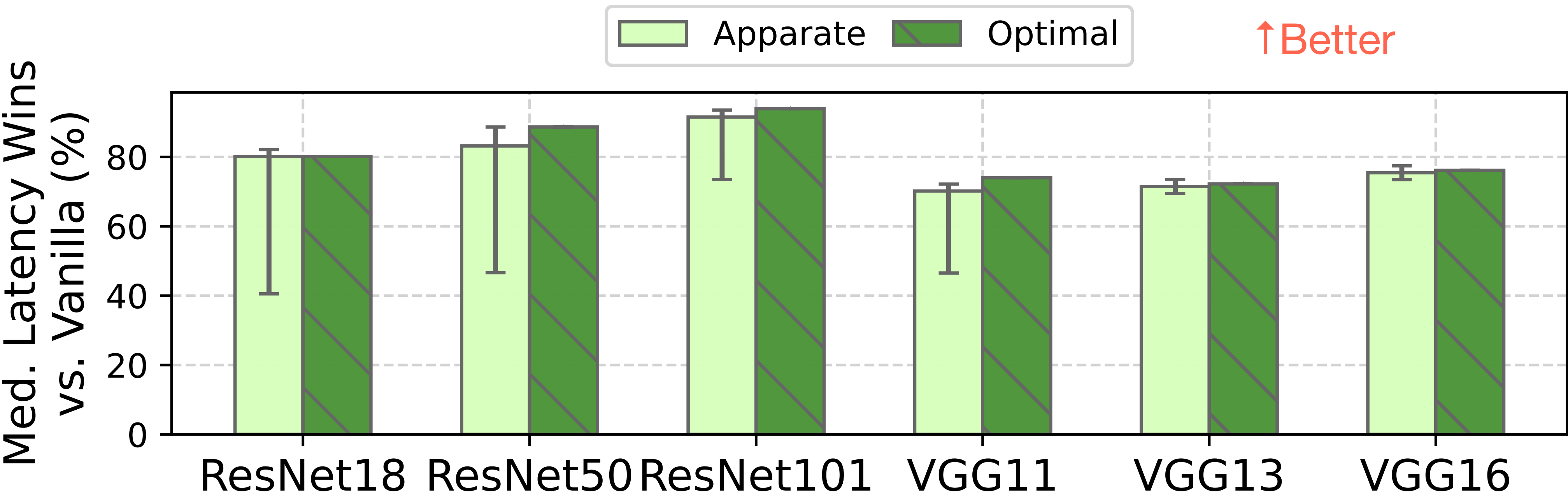
- How does Apparate compare with vanilla model inference?
 - Non-generative workloads
 - Generative workloads
- How does Apparate compare with baselines
- How does Apparate perform under different SLOs?
- How does Apparate perform under different latency/acc budget?
- What's the runtime overhead?
- ...

Evaluation

Performance on CV workloads

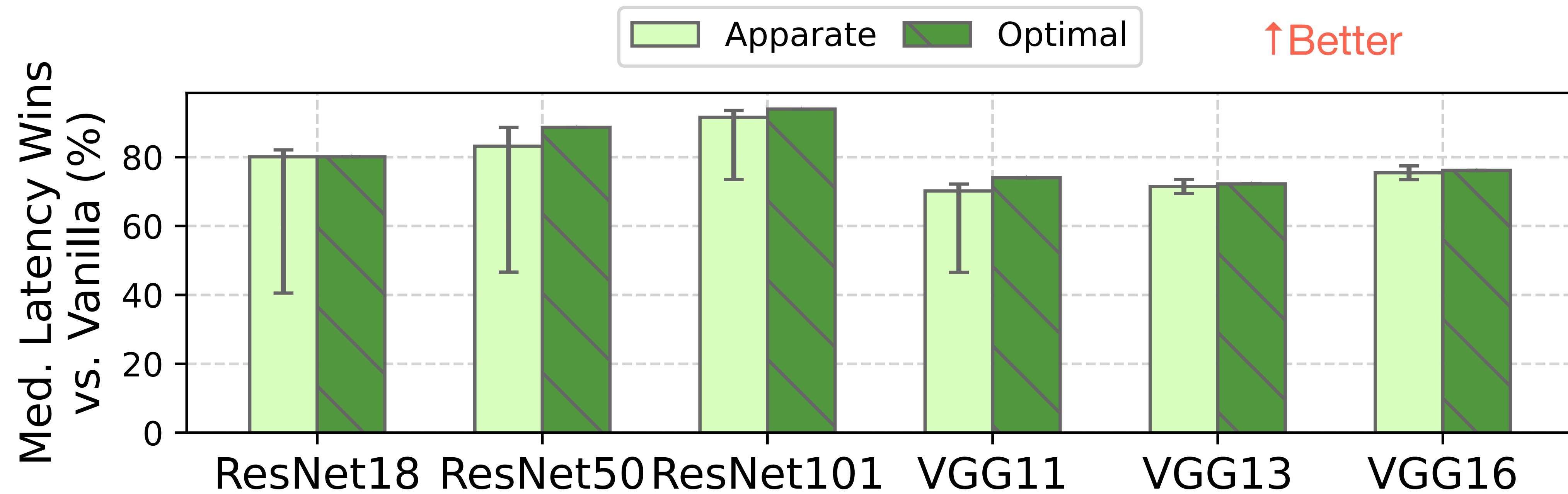
Evaluation

Performance on CV workloads



Evaluation

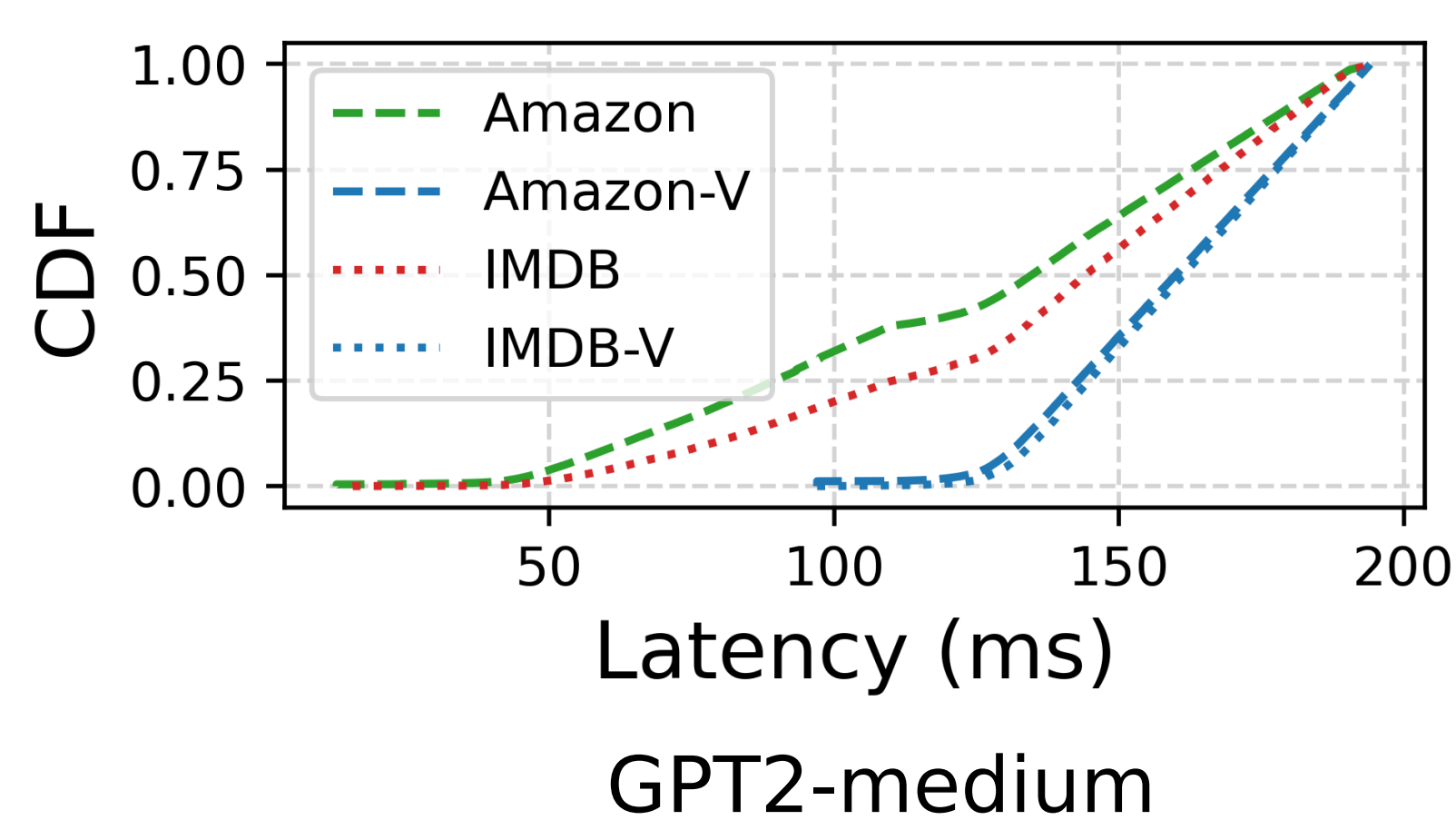
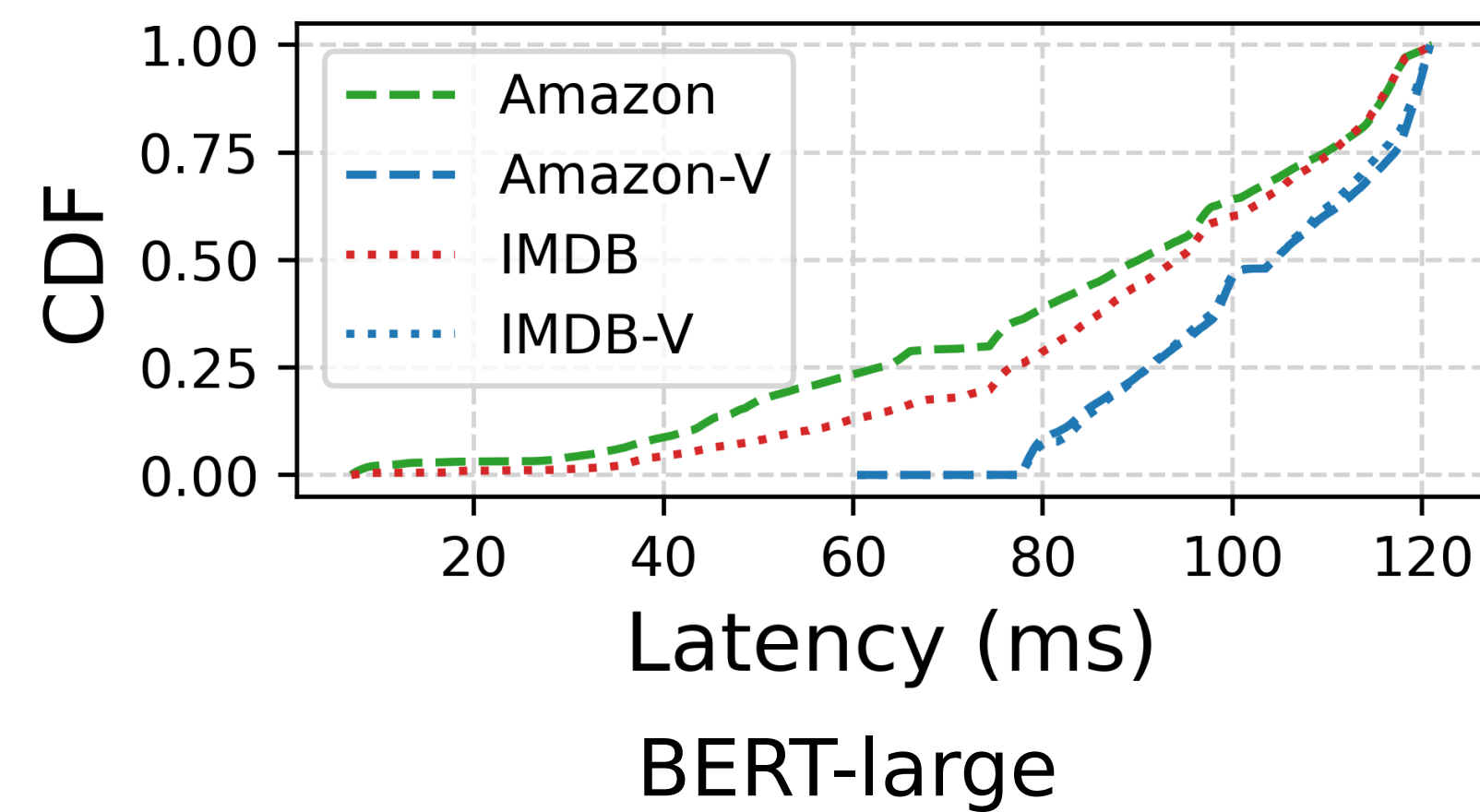
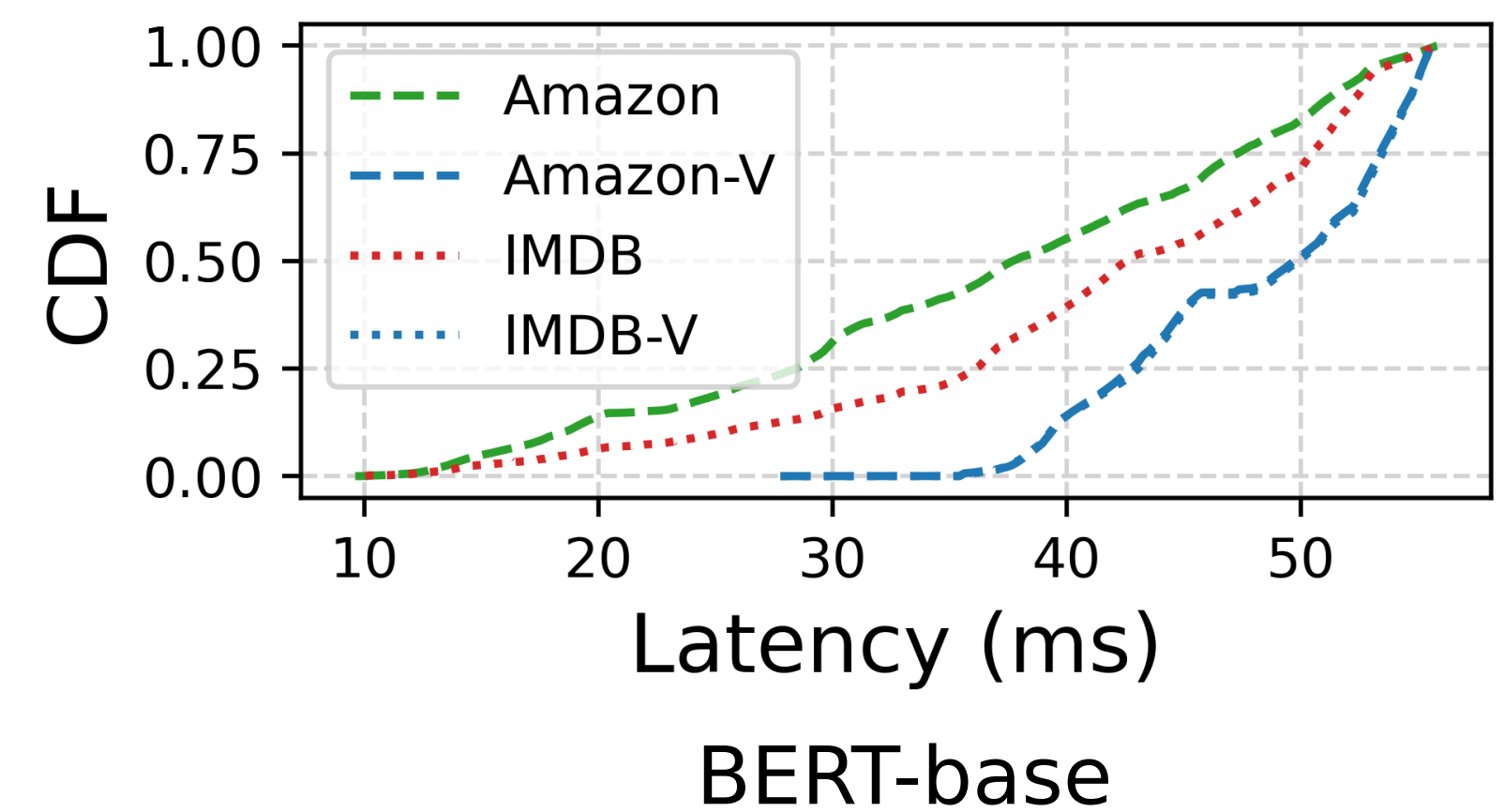
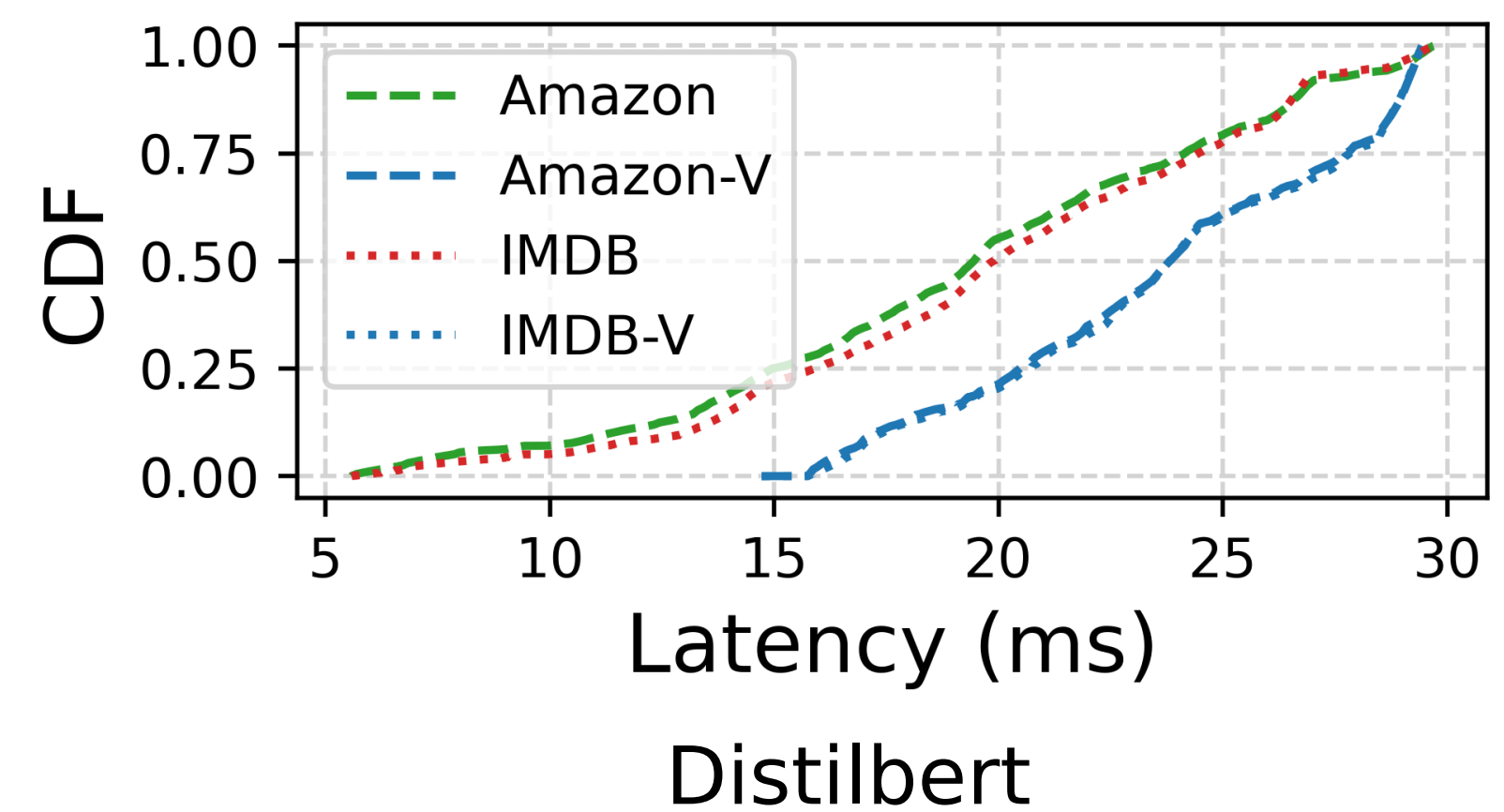
Performance on CV workloads



Up to 94% lower median latency than vanilla, and is close to optimal

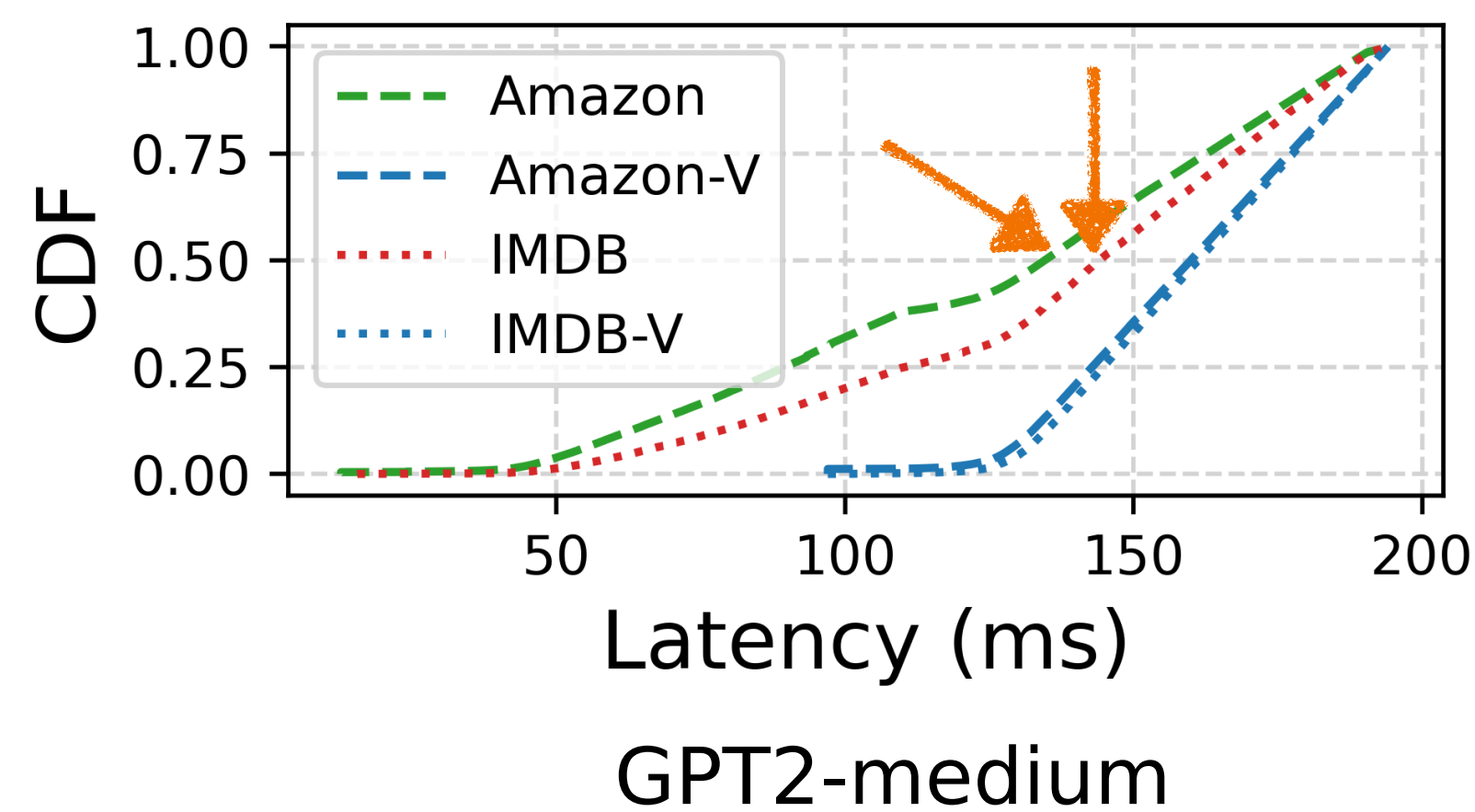
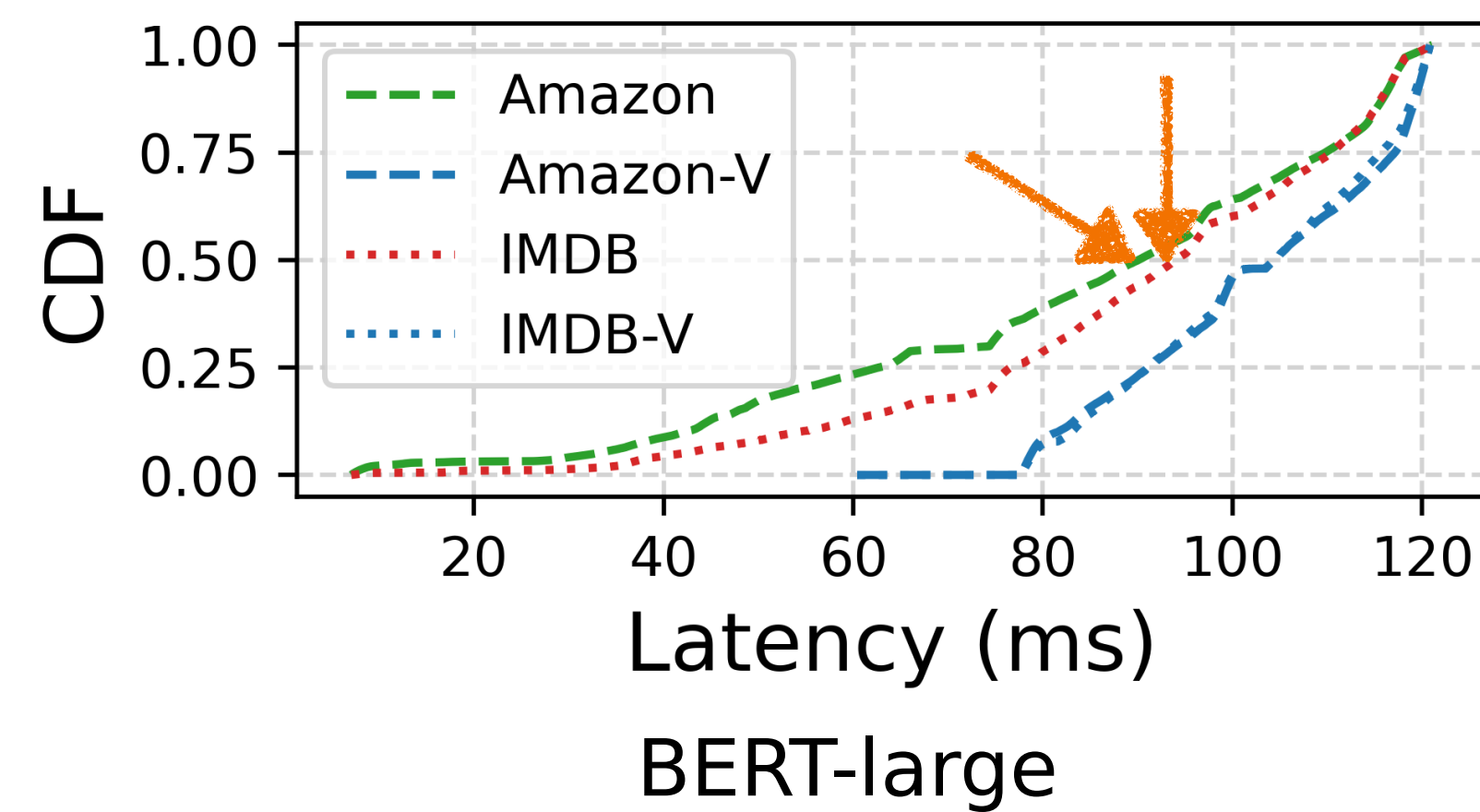
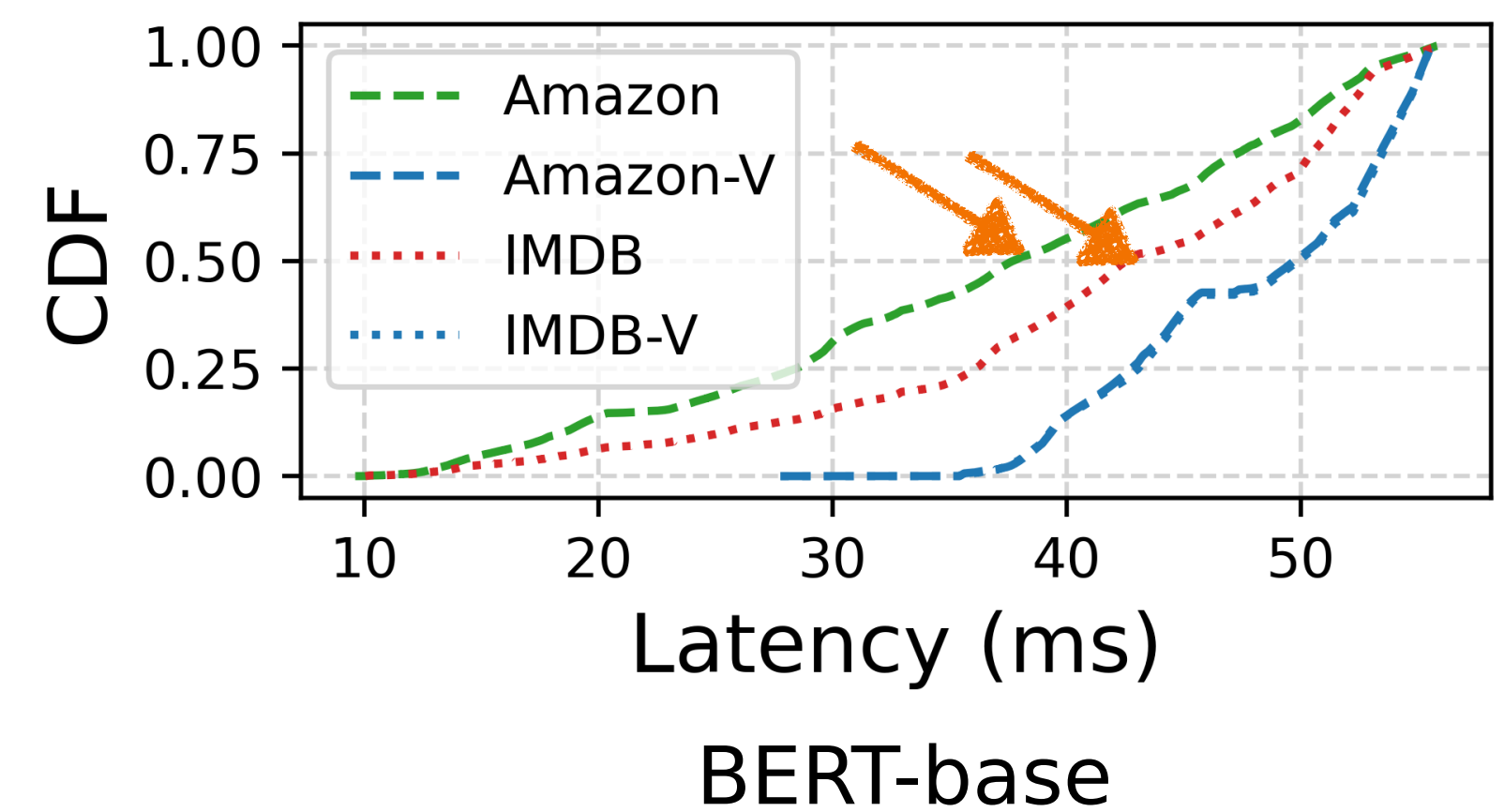
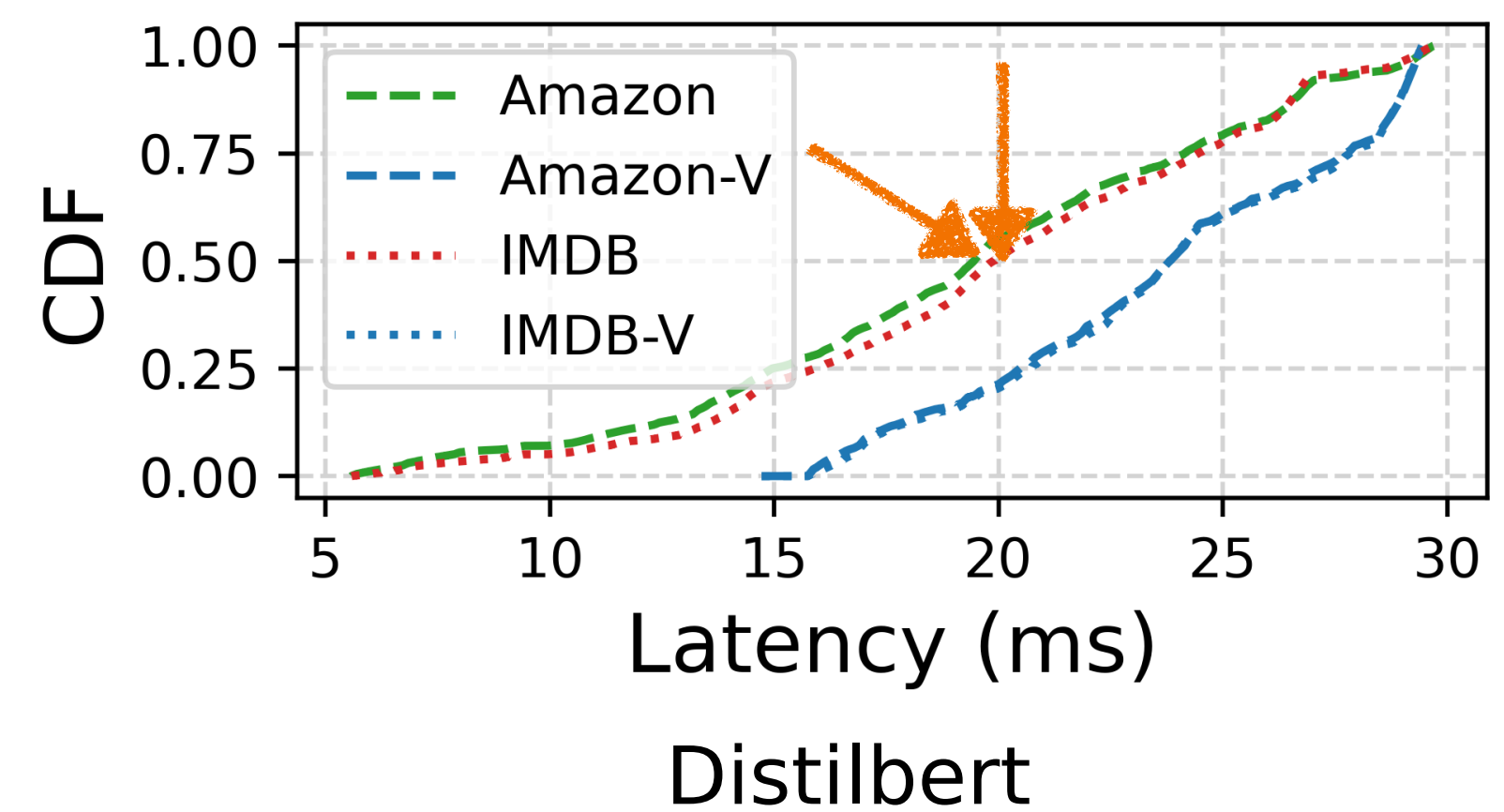
Evaluation

Performance on NLP workloads



Evaluation

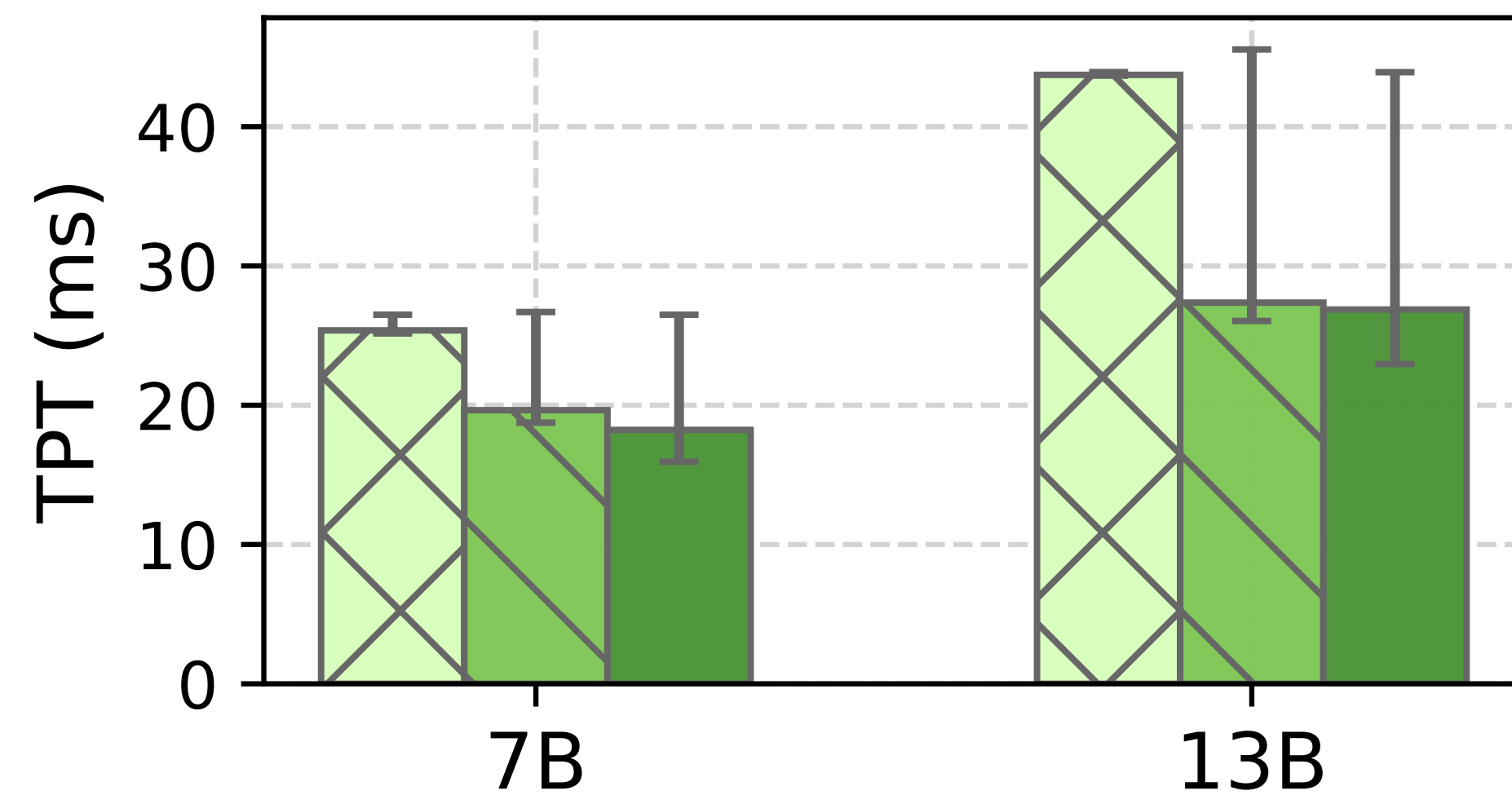
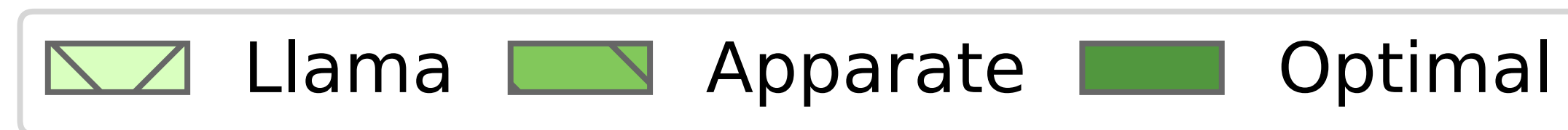
Performance on NLP workloads



Apparate lowers median latencies by 10-24%

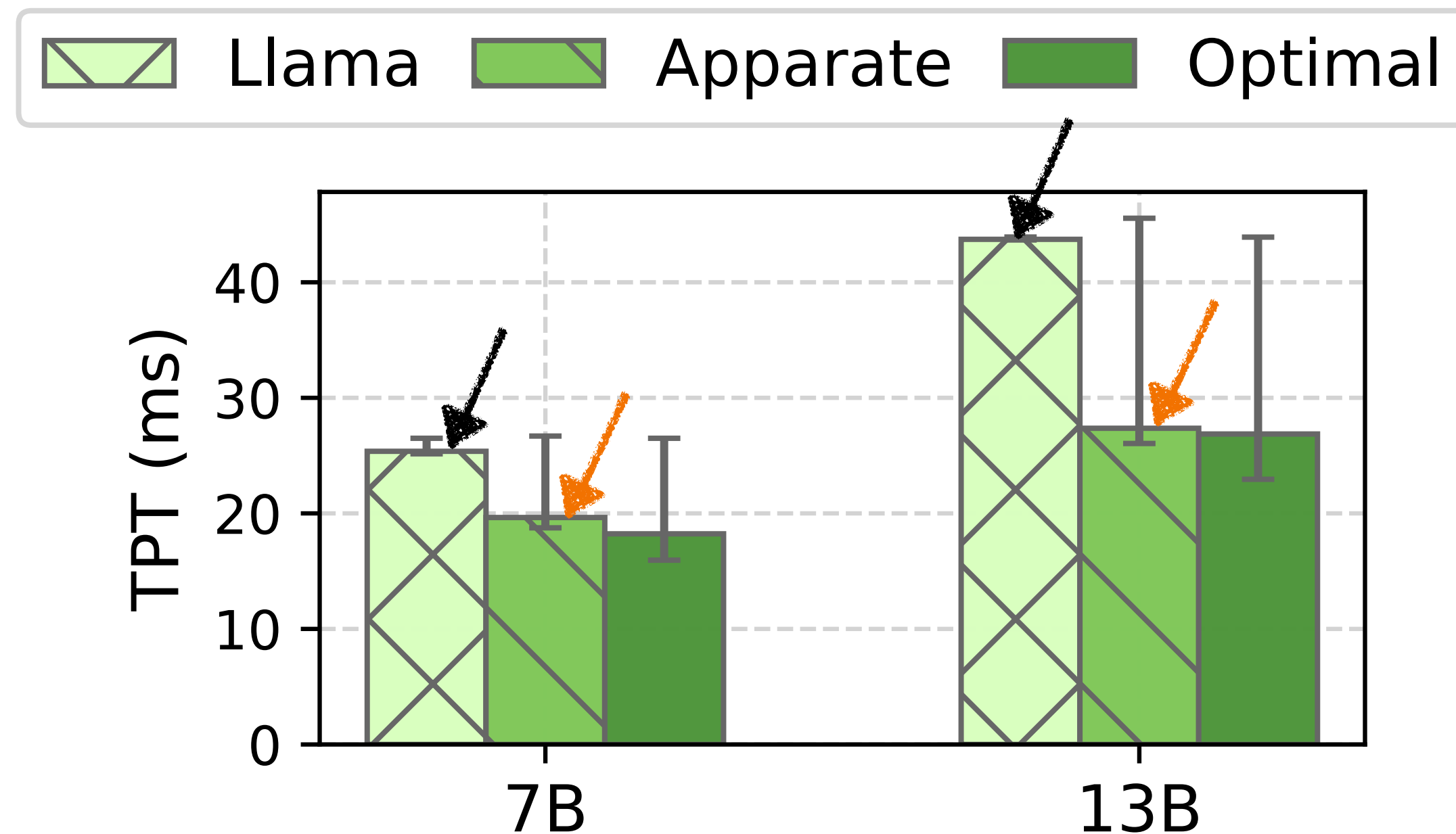
Evaluation

Performance on generative workloads



Evaluation

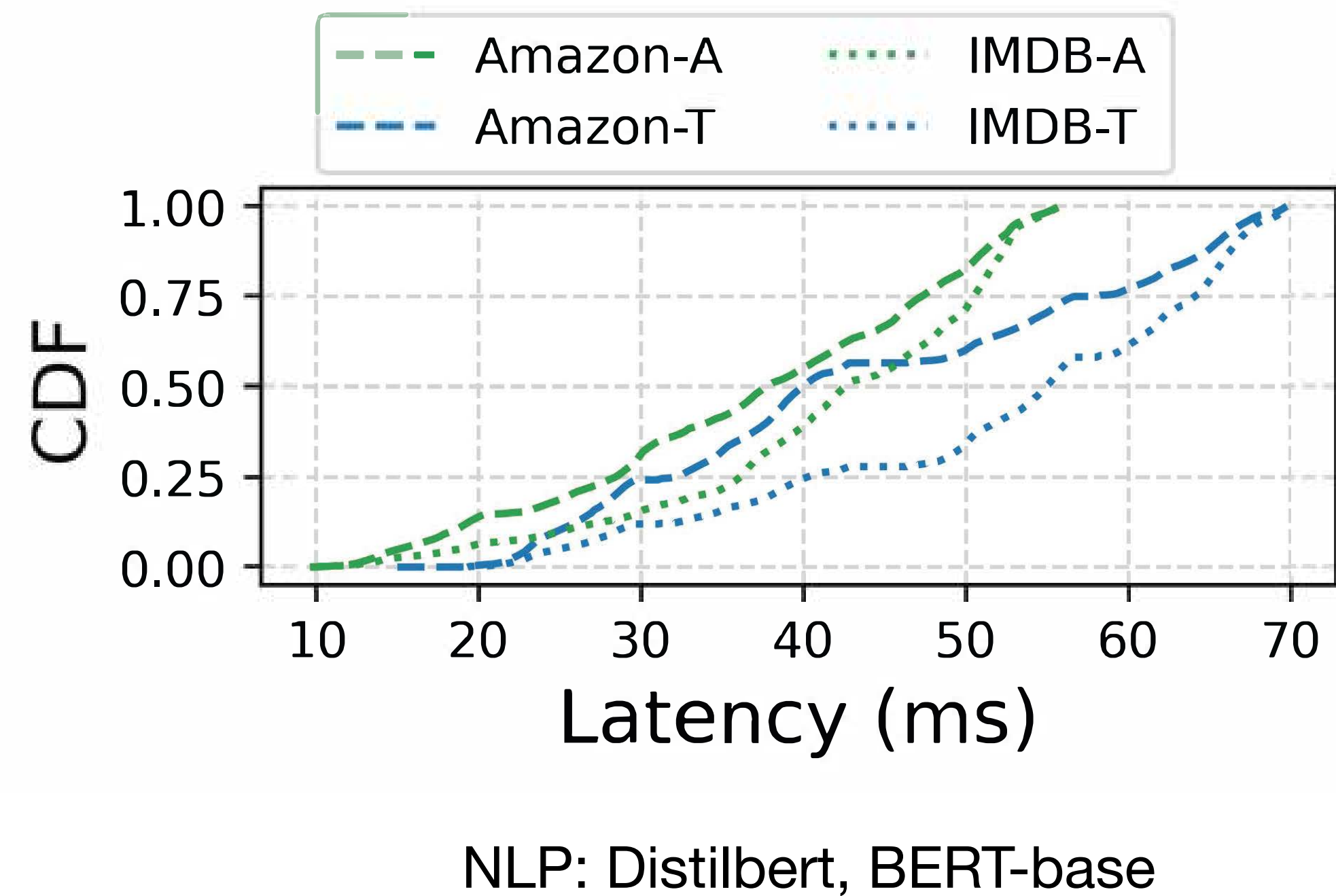
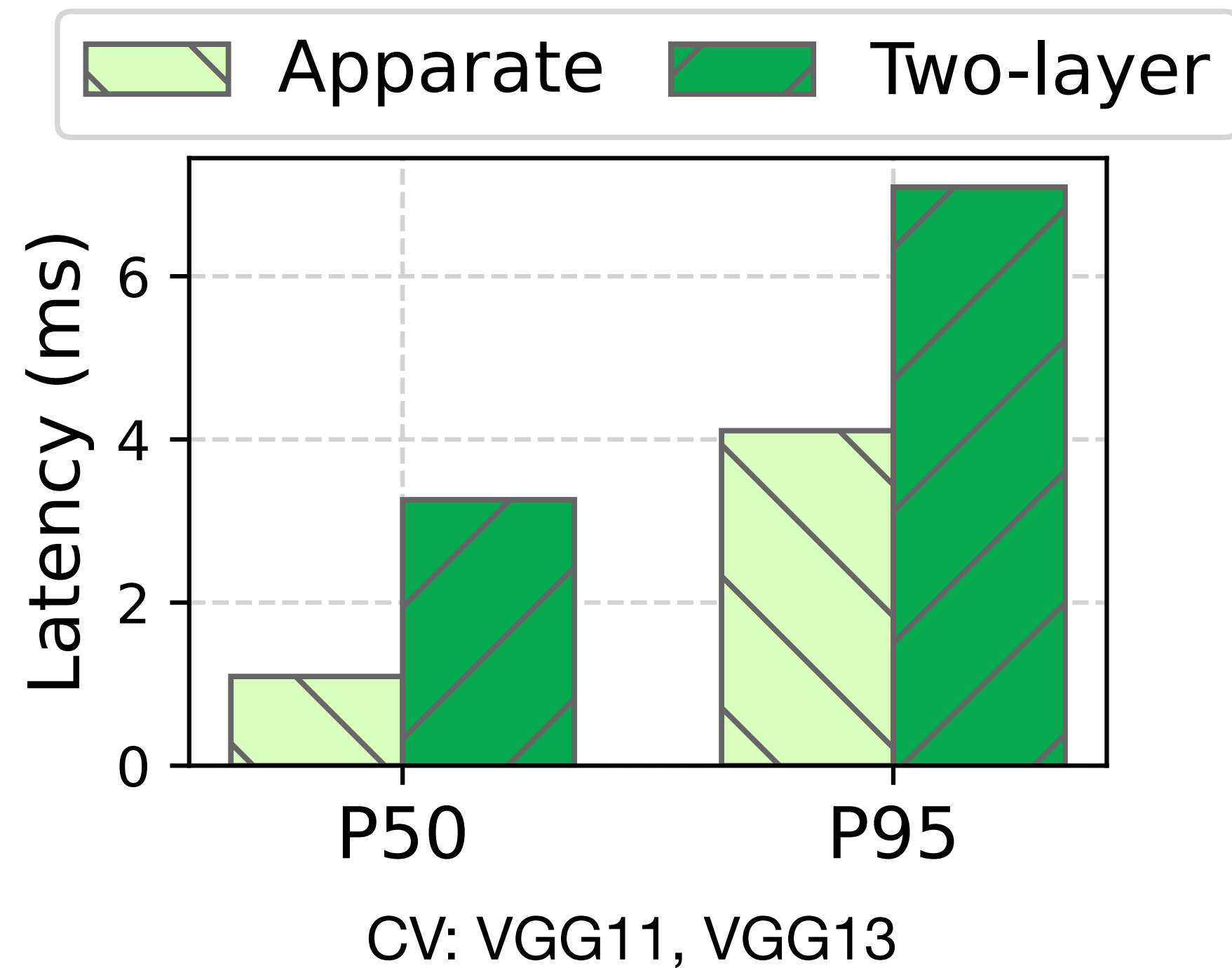
Performance on generative workloads



Apparate lowers median TPT by 22–37% compared with vanilla Llama models

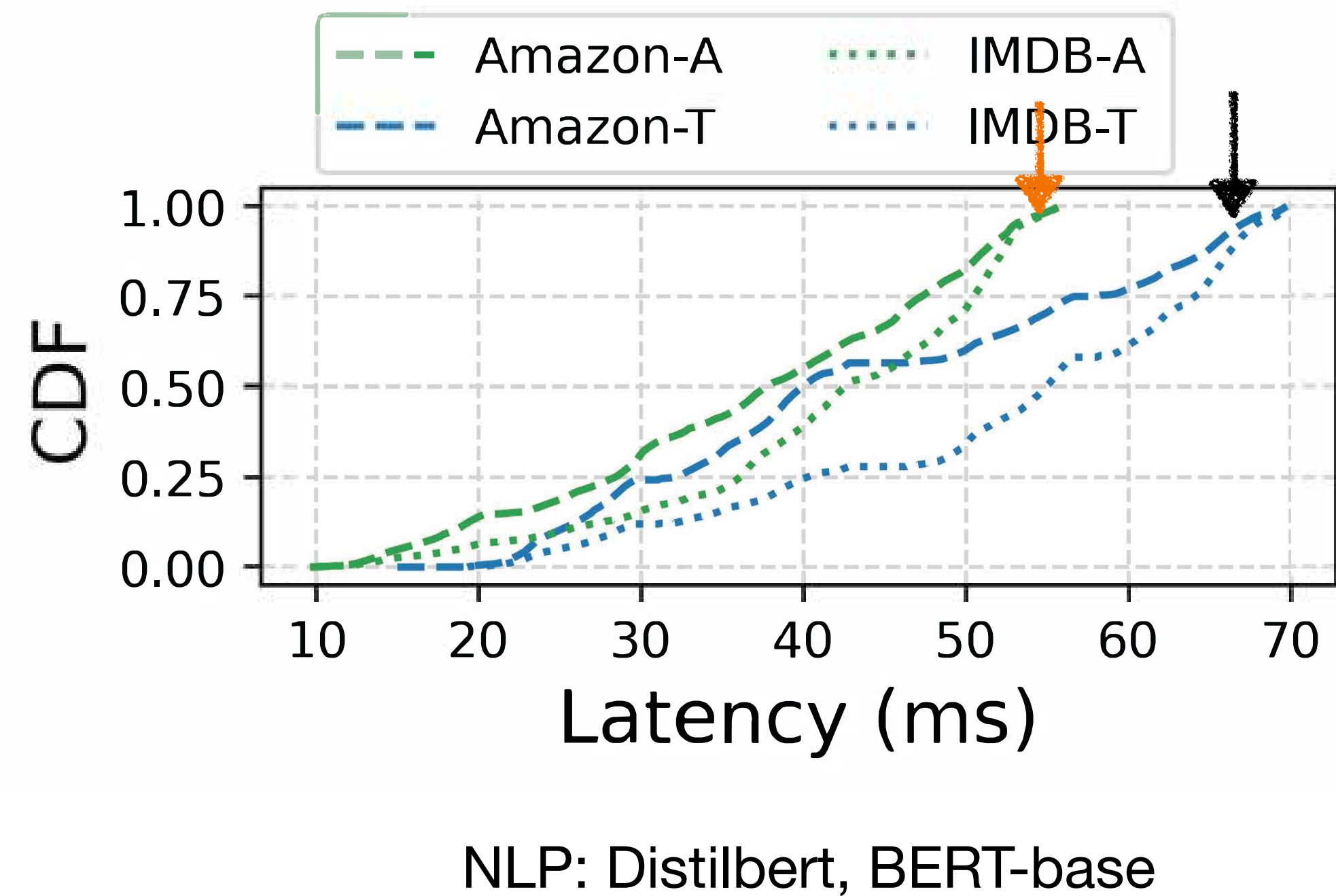
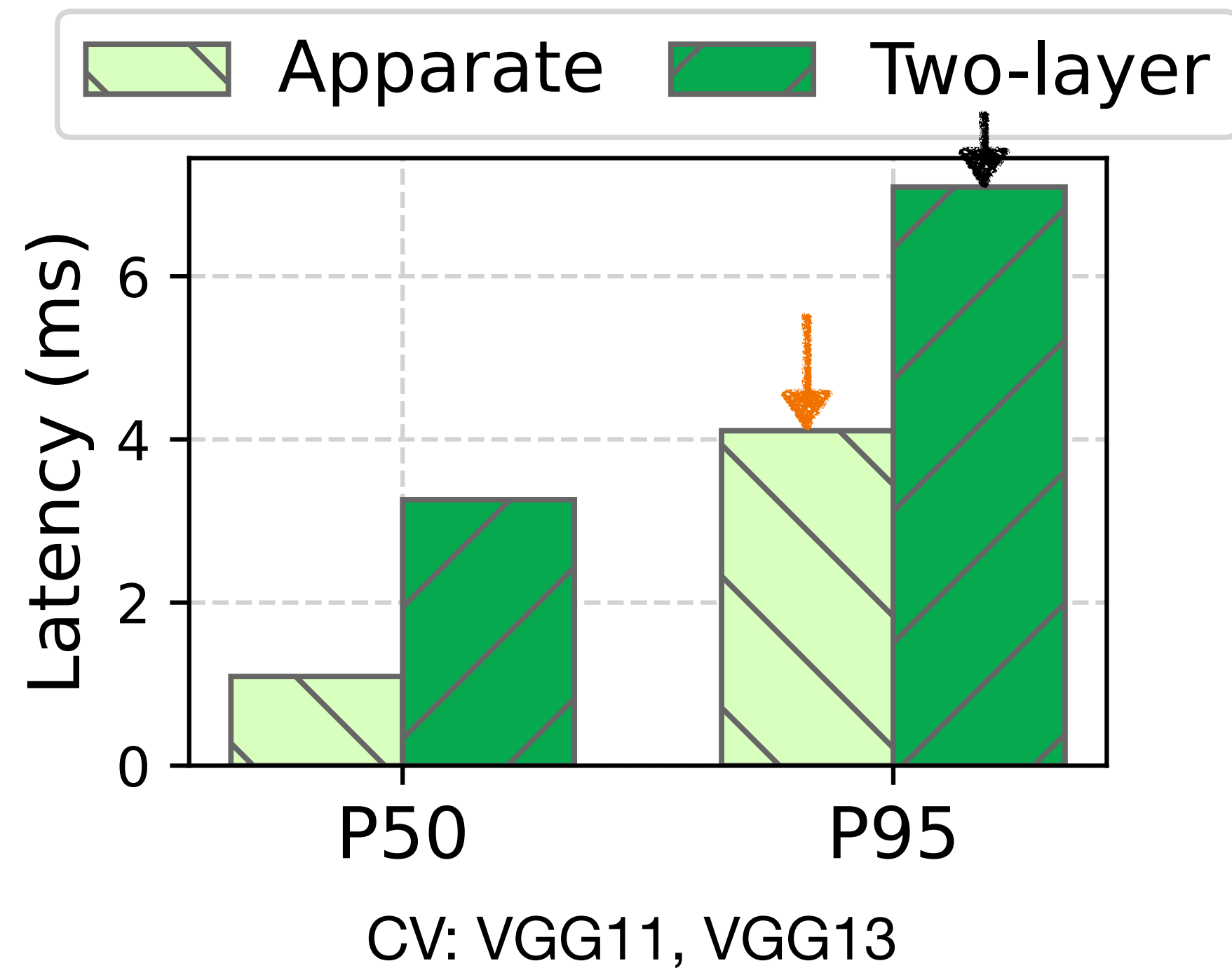
Evaluation

Compare Apparate with two layer inference systems



Evaluation

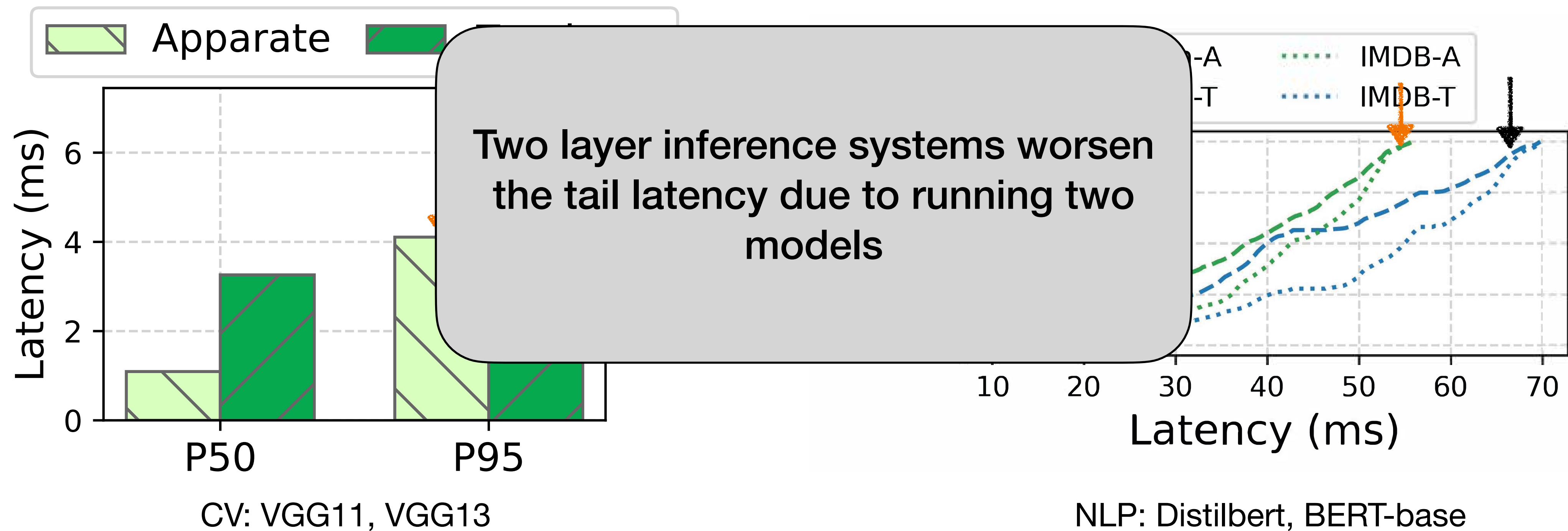
Compare Apparate with two layer inference systems



Apparate delivers 21-42% lower P95 latencies compared to these baselines

Evaluation

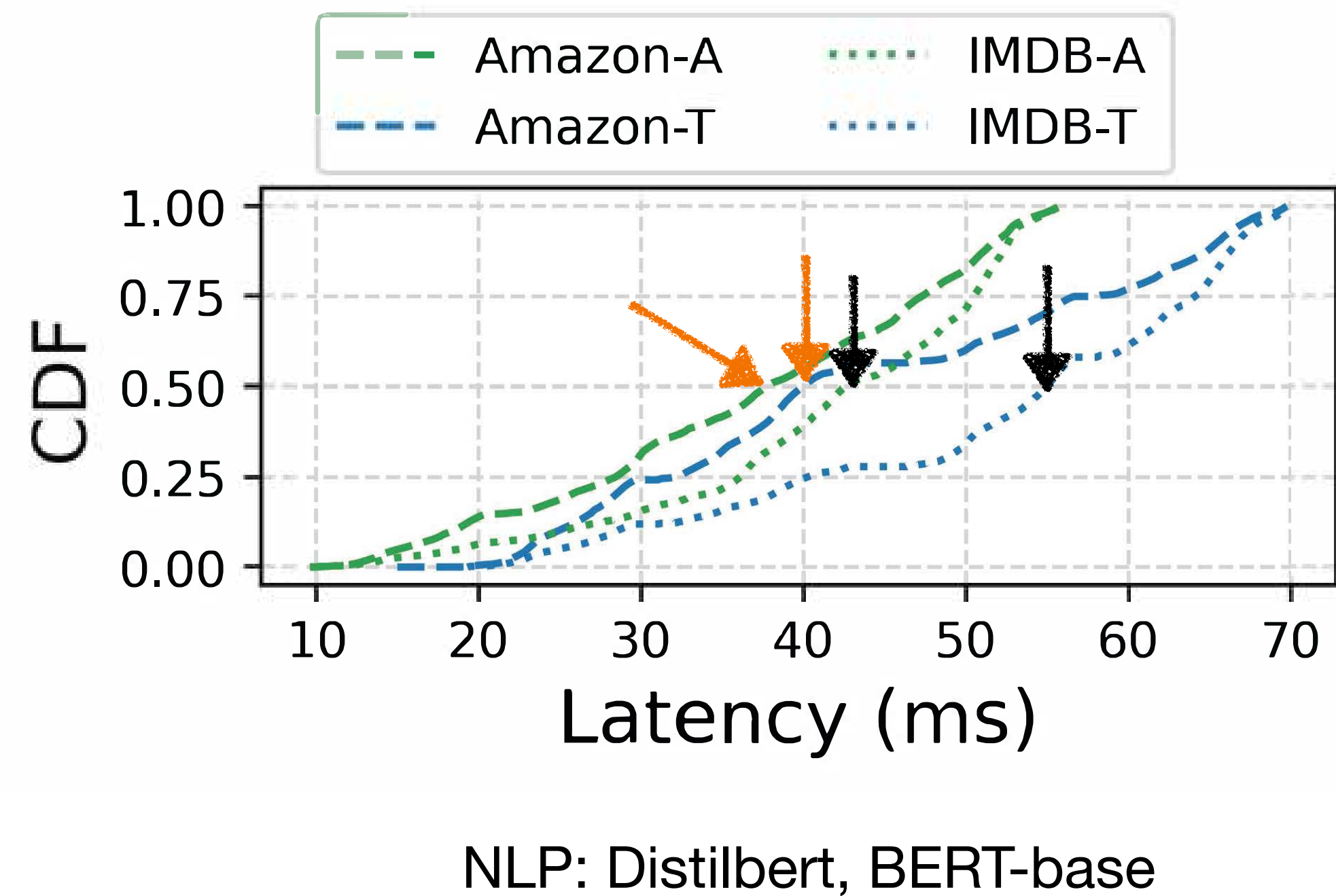
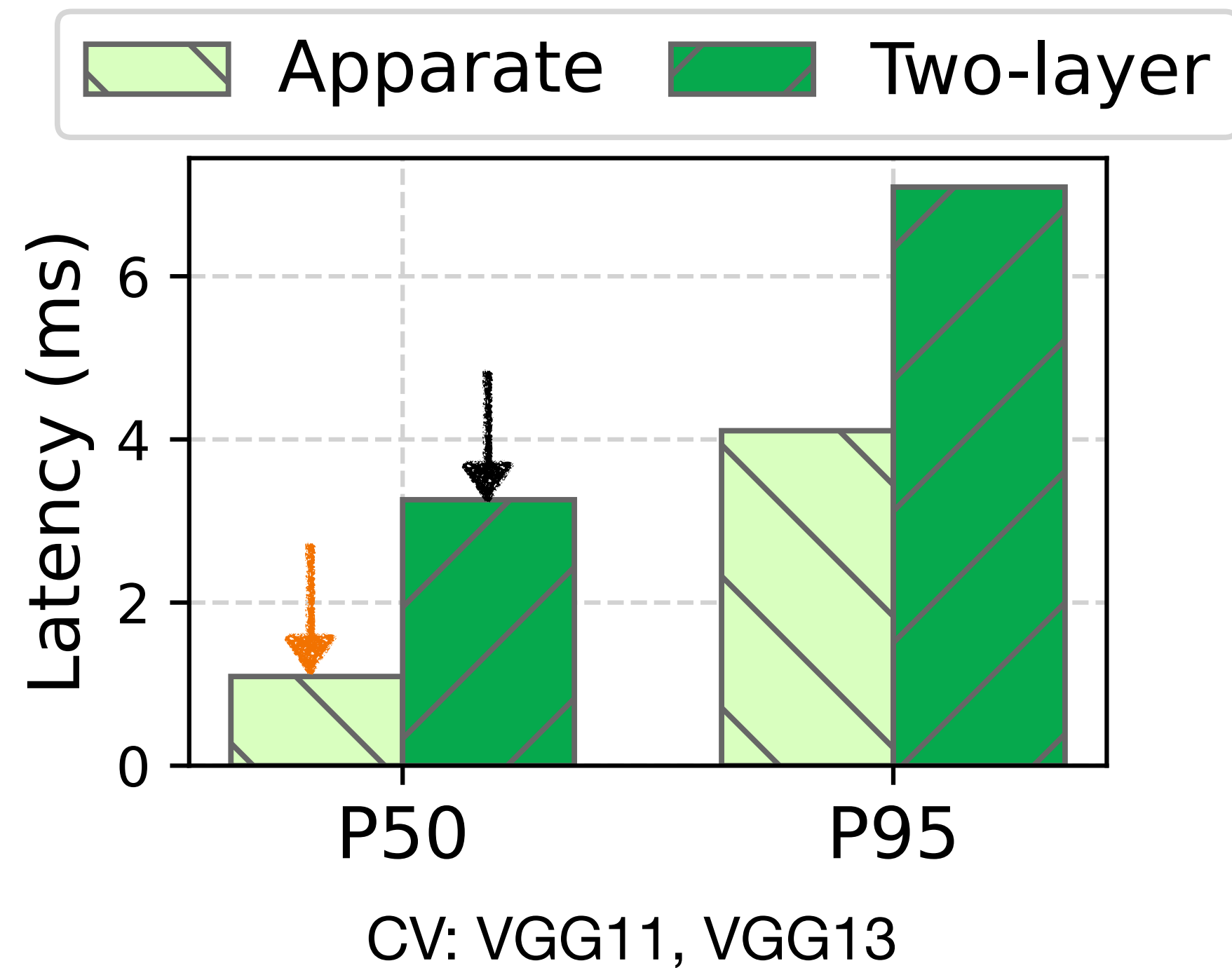
Compare Apparate with two layer inference systems



Apparate delivers 21-42% lower P95 latencies compared to these baselines

Evaluation

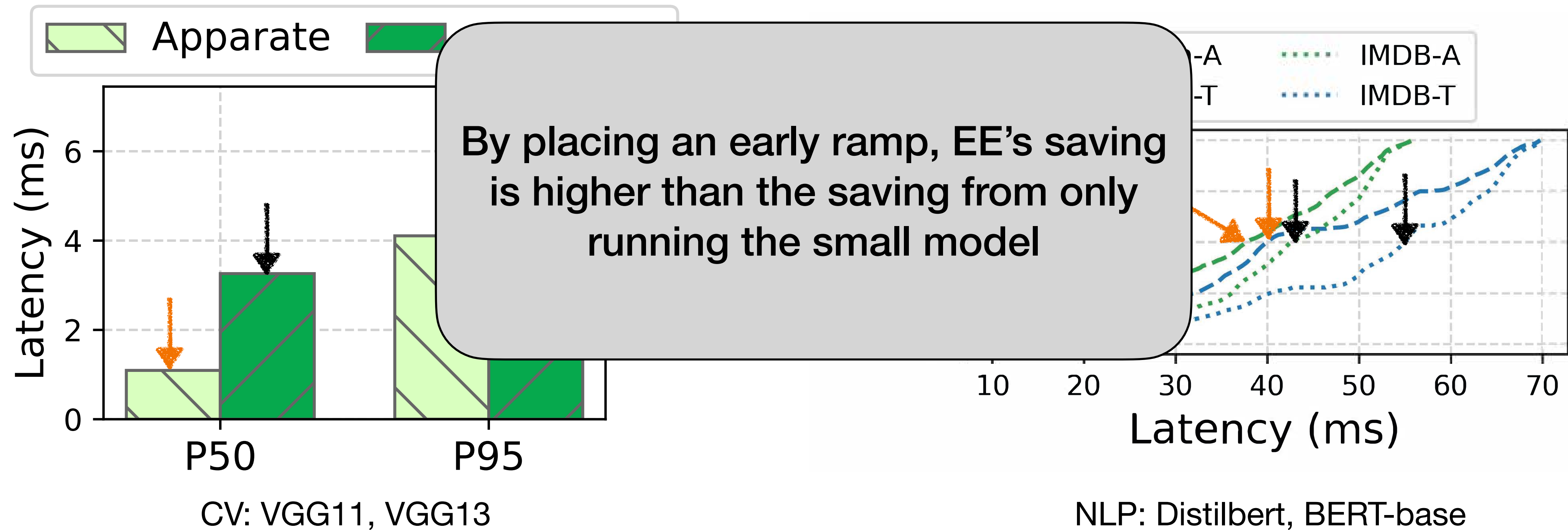
Compare Apparate with two layer inference systems



Apparate delivers up to 66% lower median latencies compared to these baselines

Evaluation

Compare Apparate with two layer inference systems



Apparate delivers up to 66% lower median latencies compared to these baselines

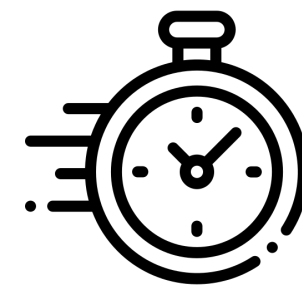
Apparate

The First Runtime Management System for Early Exit Networks

- Injects EEs for user-provided models without requiring manual effort or expertise
- Repurposes EEs for rapid results with continuous feedback for online adaptation
- Reduces per-request latency while meeting accuracy and overhead constraint



Accuracy



Latency



Throughput

Source code available at <https://github.com/dywsjtu/apparate>