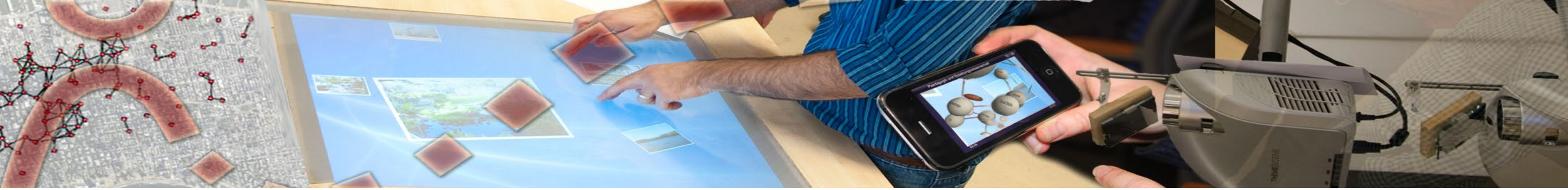


Sponge: Inference Serving with Dynamic SLOs Using In-Place Vertical Scaling

Kamran Razavi†, Saeid Ghafouri^, Max Mühlhäuser†, Pooyan Jamshidi*, Lin Wang‡

†University of Darmstadt, ^Queen Mary University of London,

*University of South Carolina, ‡Paderborn University

“More than 90% of data center compute for ML workload, is used by inference services”

Inference Serving Requirements

Highly Responsive!
(end-to-end latency guarantee)

Cost-Efficient!
(least resource consumption)

Inference Serving Requirements

Highly Responsive!
(end-to-end latency guarantee)

Cost-Efficient!
(least resource consumption)

Resource Scaling

In-place Vertical Scaling
(more responsive)

Horizontal Scaling
(more cost efficient)

Dynamic User -> Dynamic Network Bandwidths

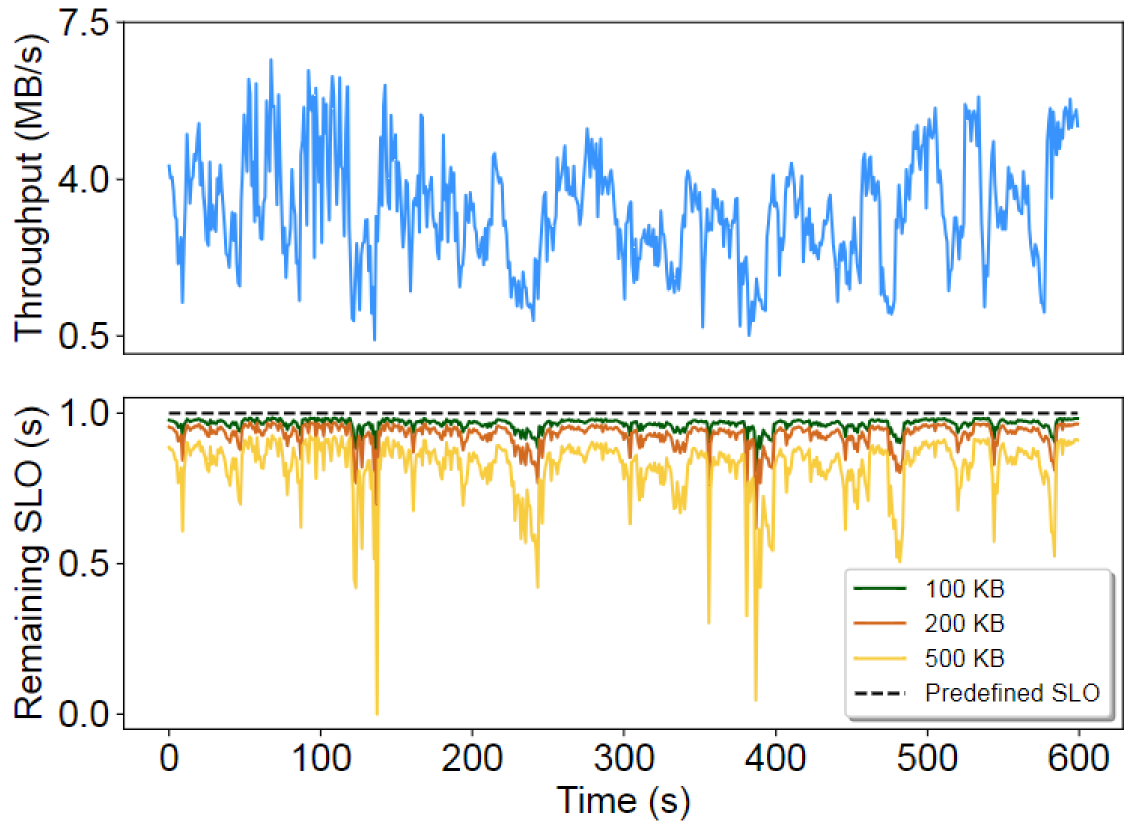
- Users move
  - Fluctuations in the network bandwidths
    - Reduced time-budget for processing requests

$$\overline{\underbrace{\hspace{4cm}}_{\text{network latency}}\;\underbrace{\hspace{4cm}}_{\text{processing latency}}}^{\text{SLO}}$$

Dynamic User -> Dynamic Network Bandwidths

- Users move
  - Fluctuations in the network bandwidths
    - Reduced time-budget for processing requests

$$\overbrace{\underbrace{\qquad\qquad\qquad\qquad}_{\text{network latency}}\underbrace{\qquad\qquad\qquad\qquad}_{\text{processing latency}}}^{\text{SLO}}$$



Inference Serving Requirements

Highly Responsive!
(end-to-end latency guarantee)

Cost-Efficient!
(least resource consumption)

Resource Scaling

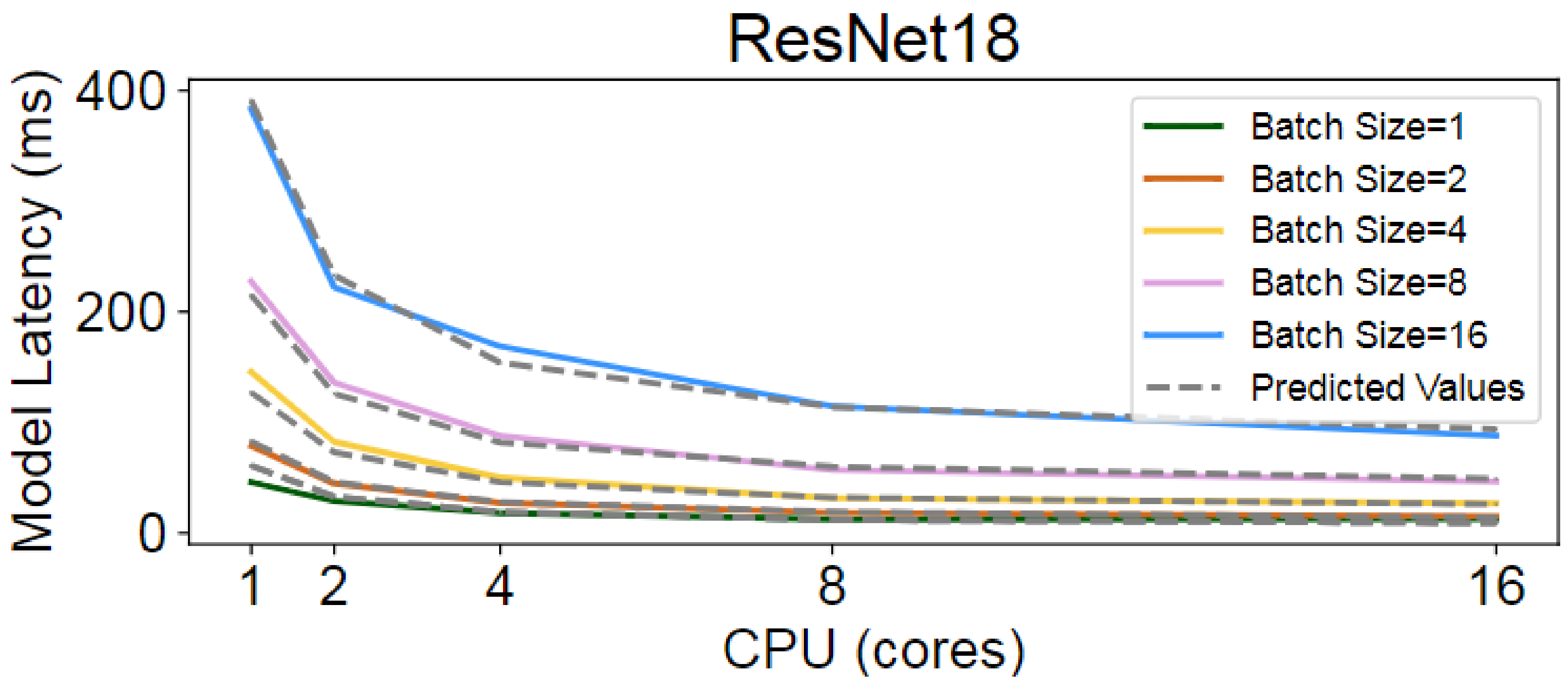
Sponge!

In-place Vertical Scaling
(more responsive)

Horizontal Scaling
(more cost efficient)

Vertical Scaling DL Model Profiling

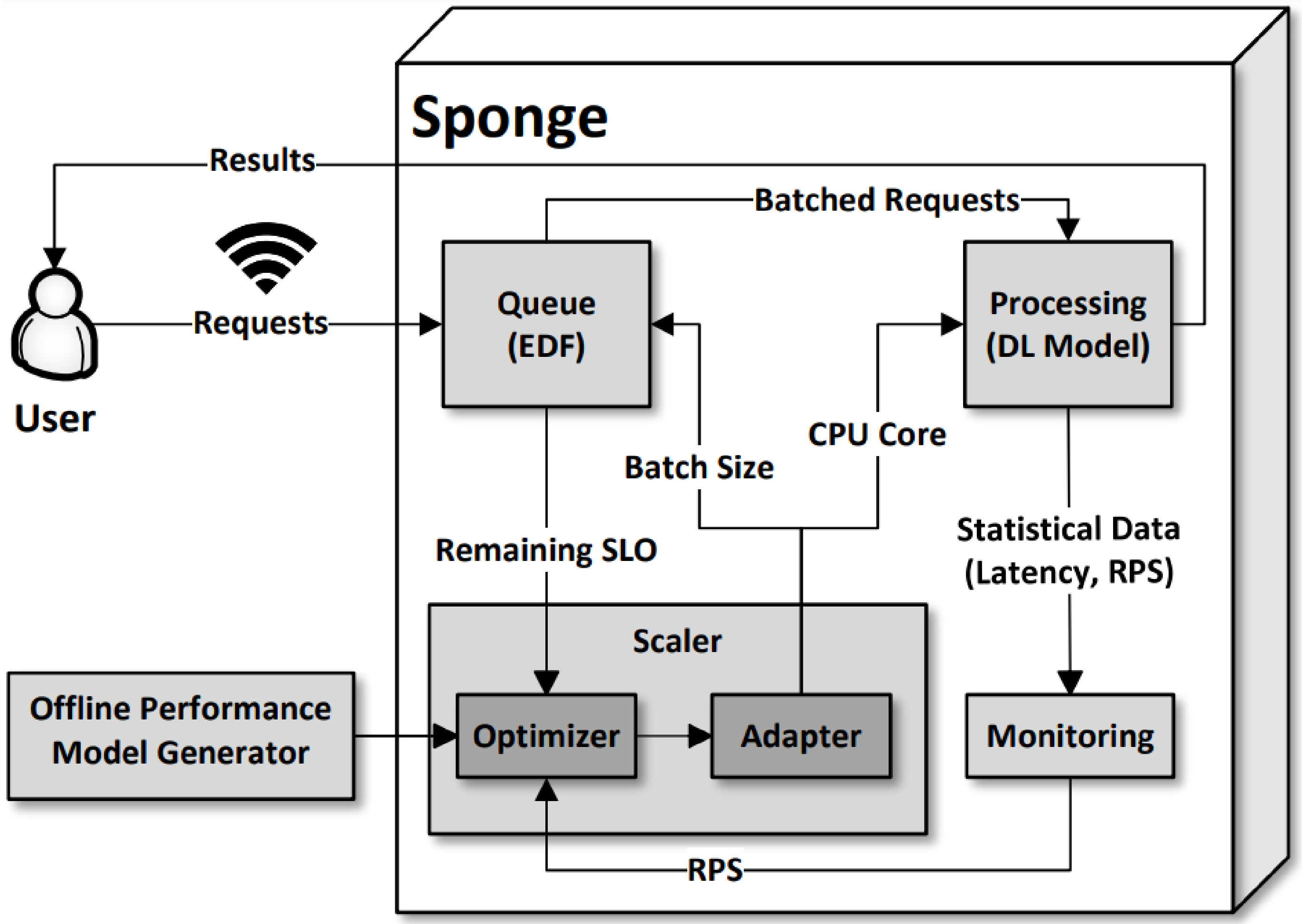
- How much resource should be allocated to a DL model?
  - Latency/batch size → linear relationship
  - Latency/CPU allocation → inverse relationship



System Design

3 design choices:

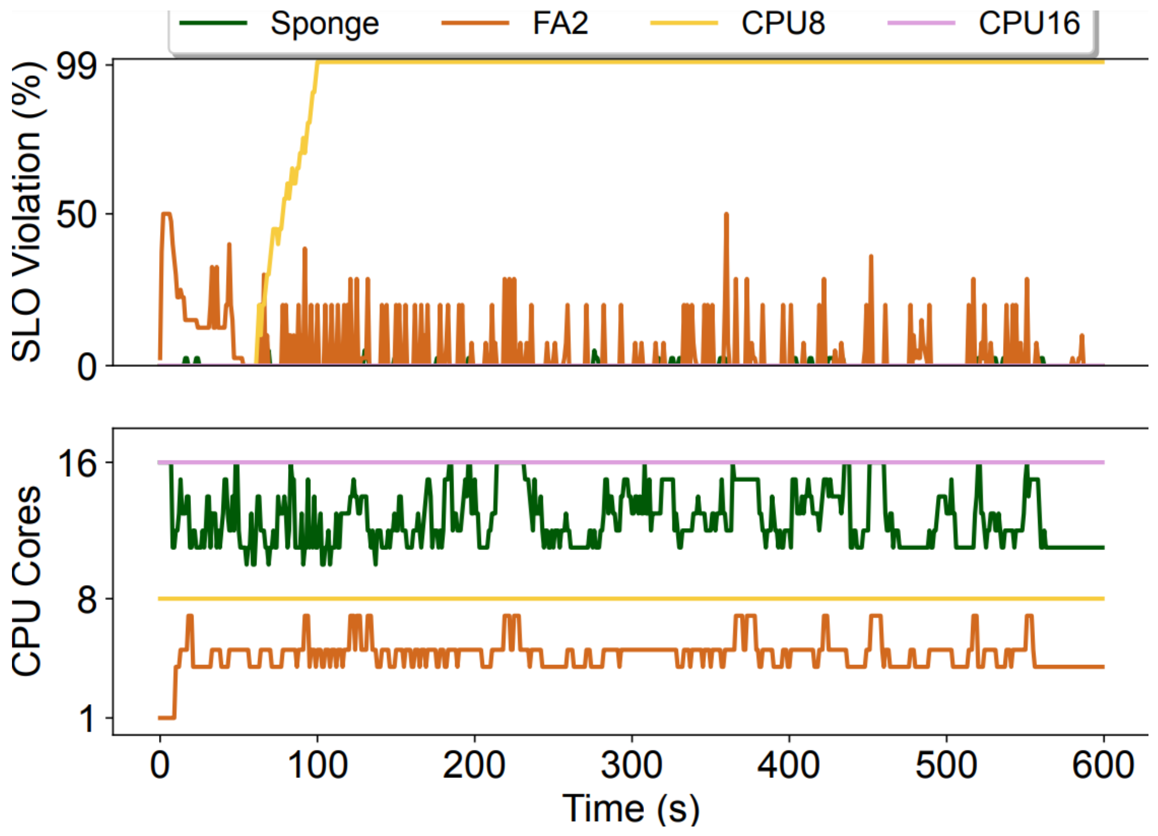
1. In-place vertical scaling
   - Fast response time
2. Request reordering
   - High priority requests
3. Dynamic batching
   - Increase system utilization



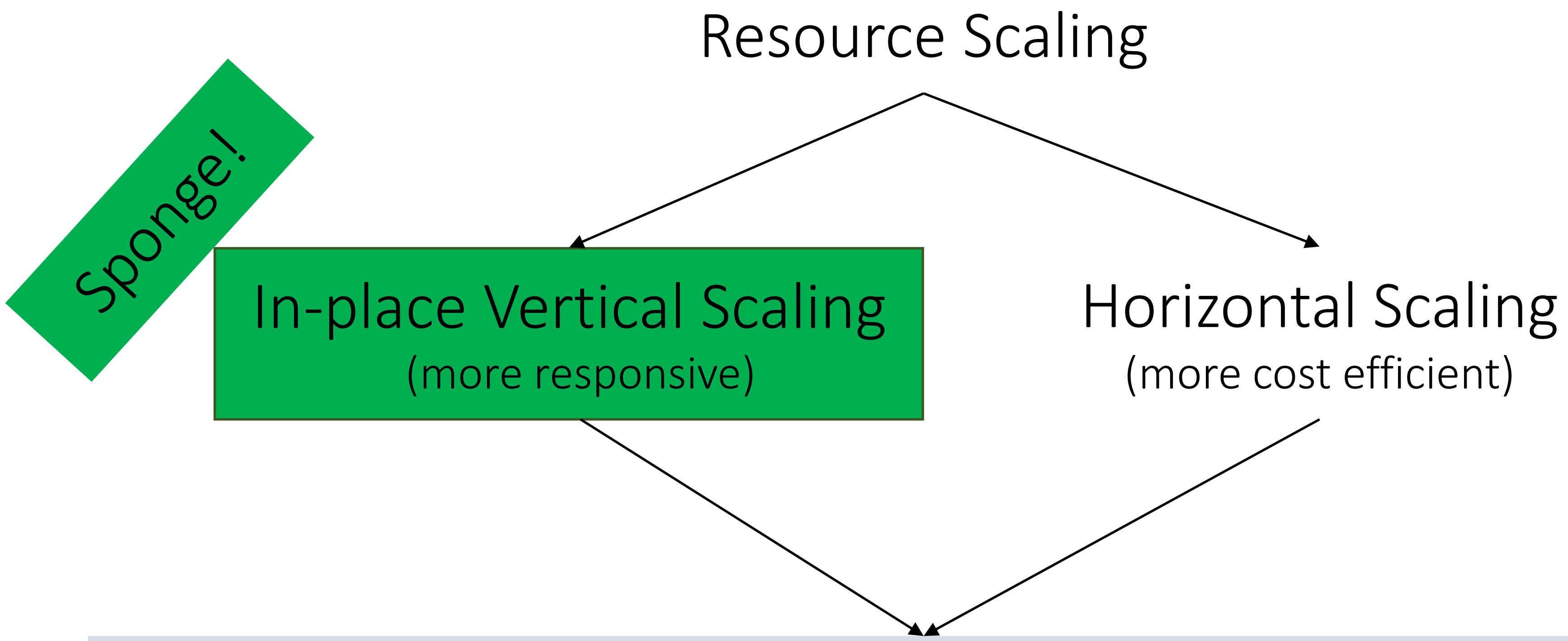
Evaluation

SLO guarantees (99th percentile) with up to 20% resource save up compared to static resource allocation.

Sponge source code:
https://github.com/saeid93/sponge



Future Directions



How can both scaling mechanisms be used jointly under a dynamic workload to be responsive and cost efficient while guaranteeing SLOs?