Model-Switching: Dealing with Fluctuating Workloads in MLaaS* Systems

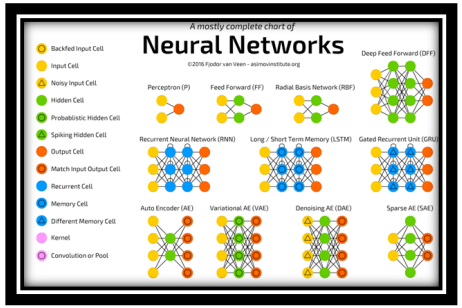
Jeff Zhang[1], Sameh Elnikety[2], Shuayb Zarar[2], Atul Gupta[2], Siddharth Garg[1]

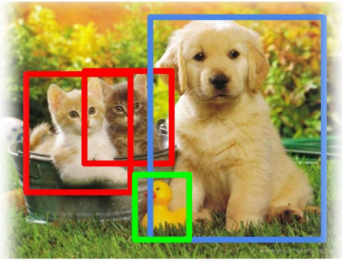
[1]New York University, [2]Microsoft

jeffjunzhang@nyu.edu

[*] Machine Learning-as-a-Service

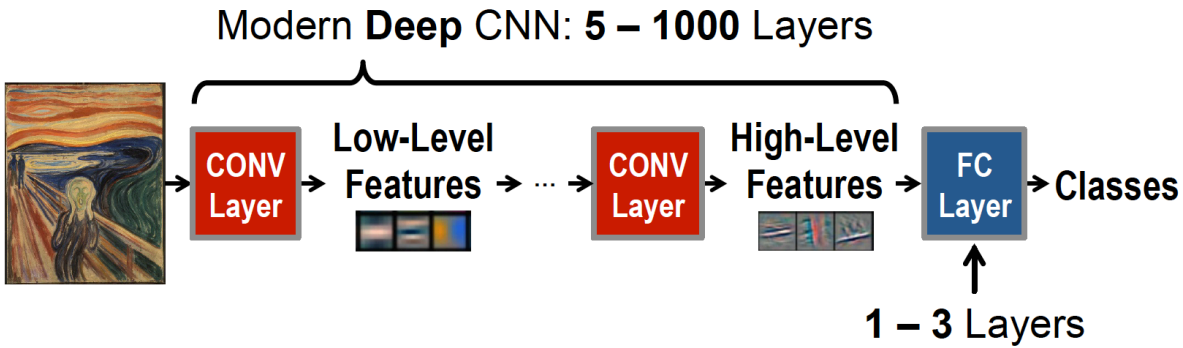
Deep-Learning Models are Pervasive



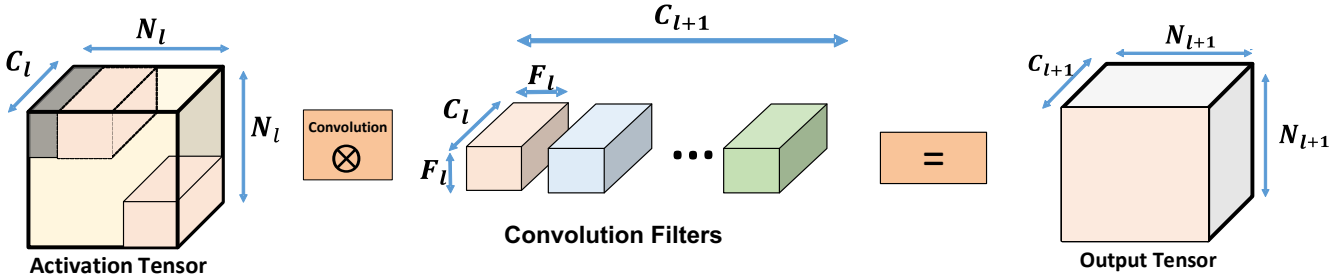


CAT, DOG, DUCK

Computations in Deep Learning

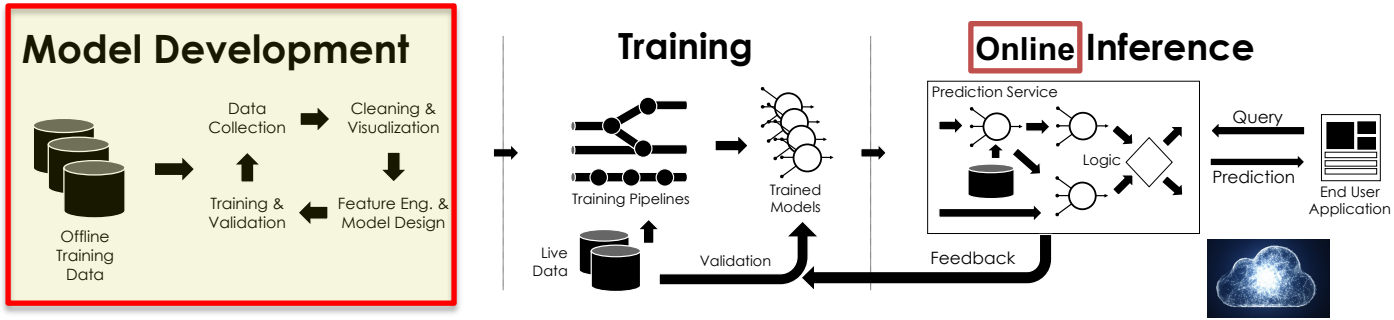


Convolutions account for **more than 90%** computation → dominate both **run-time** and **energy.***



Execution time factors: depth, activation/filter size in each layer

Machine Learning Lifecycle (Typical)



MLaaS

Model Development

- Prescribes model design, architecture and data processing

Training

- At scale on live data
- Retraining on new data and manage model versioning

Serving or Online Inference

- Deploys trained model into device, edge, or cloud

MLaaS: Challenges and Limitations

Maintain QoS under *dynamic workloads*.

Existing Solution

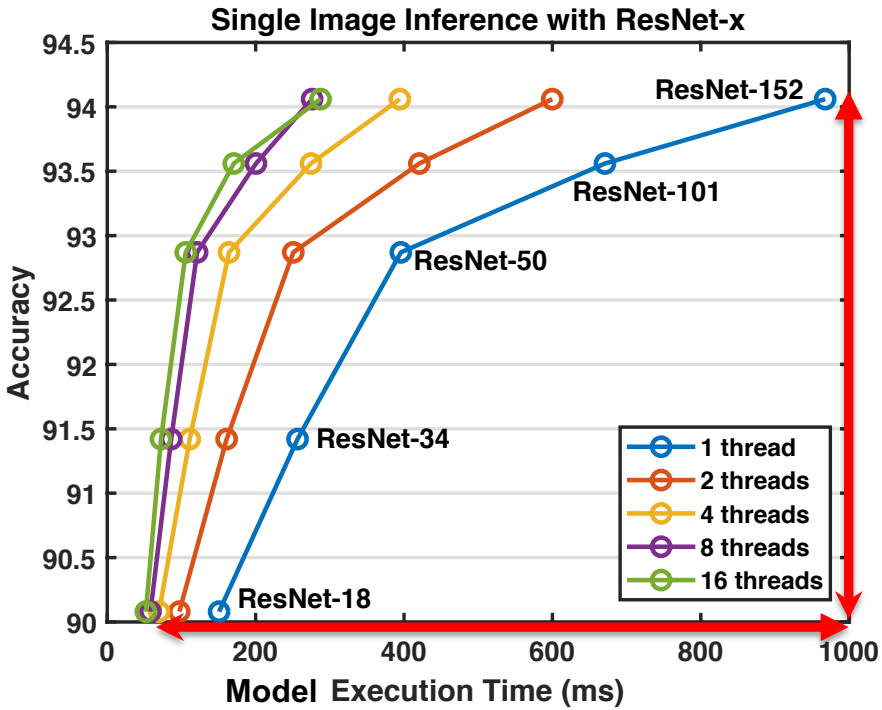
- Static model versioning
  - Tie each application to one specific model ***at run-time***
- In the event of load spikes:
  - *Prune requests* (new, low priority, near deadline *etc.*)
    - QoS violations, customer ☹
  - *Add "significant new capacity"* (autoscaling)
    - Not economically viable, provider ☹

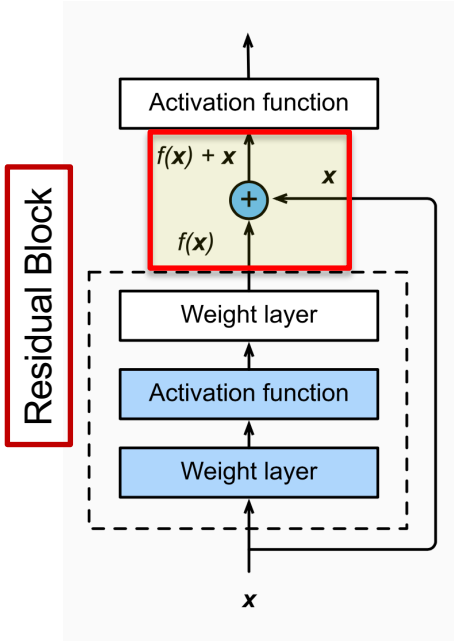
BRIEF March 28, 2020

Microsoft cloud use spikes 775%, testing capacity during coronavirus

- Service success rates dropped below 99.99%
- Teams suffered 2-hour outage in Europe
- Free offers and new subscriptions were limited

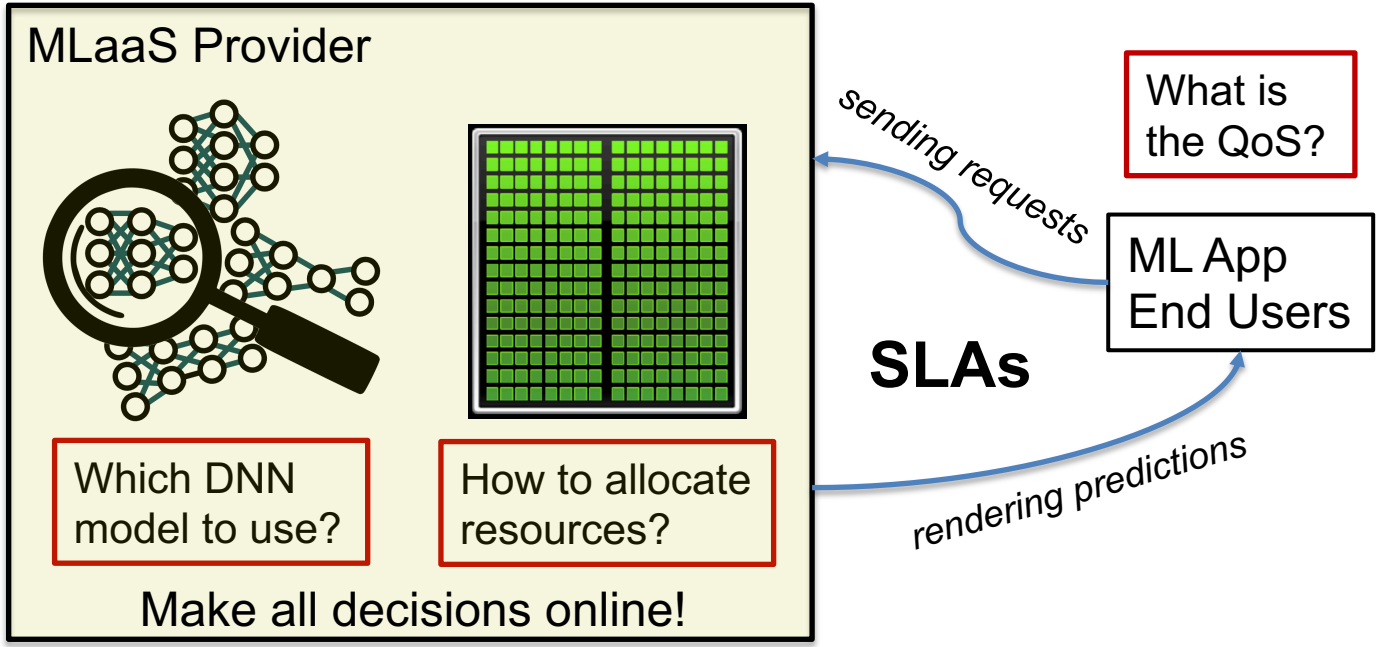
Opportunity: DNN Model Diversity





For the same application, many models can be trained with tradeoffs among:
Accuracy, Inference Time and Computation Cost (Parallelism)

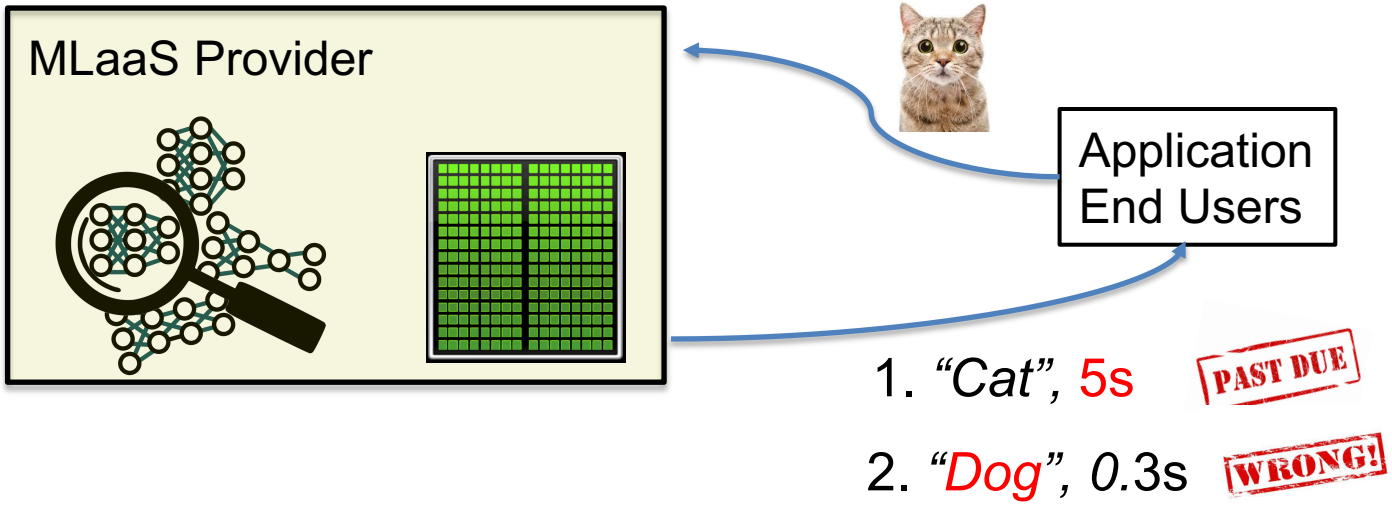
Questions in this Study



Typical SLA Objectives:
latency, throughput, cost,…

Assumption:
fluctuating workloads
fixed hardware capacity

What Do Users Care About?



From the users' perspective, deadline misses and incorrect predictions are equally bad:

- User can a*lways* meet deadline by guessing randomly

Quick *and* correct predictions!

A New Metric for MLaaS

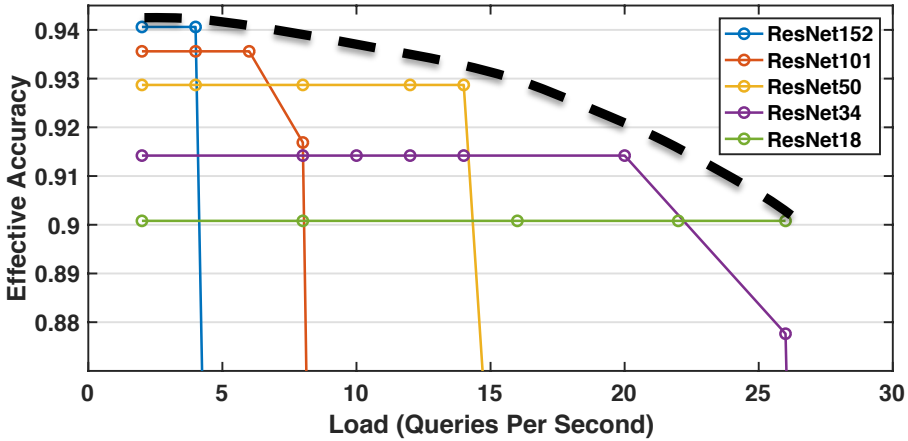
Effective Accuracy (a^{eff}) :

the fraction of correct predictions *within deadline* (*D*)

$$a^{eff} = p^{D,\lambda} \times a$$

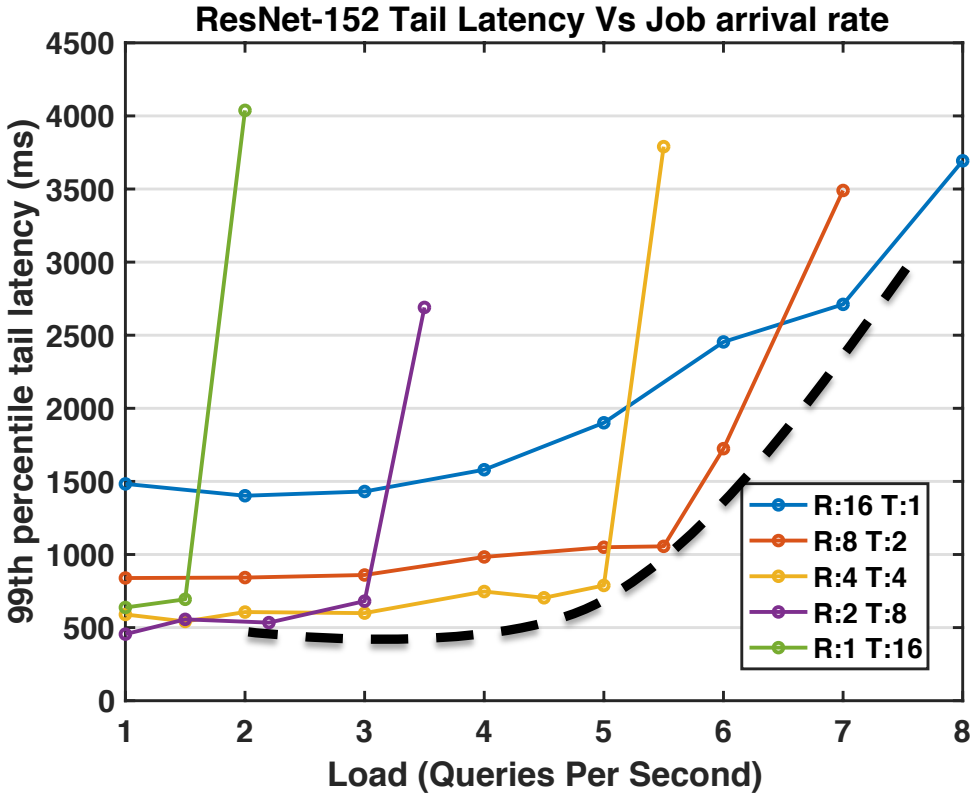
(λ: load, a: baseline accuracy)

Likelihood of meeting deadline



No single DNN works best at all load levels

Characterizing DNN Parallelism



R: Replicas
T: Threads

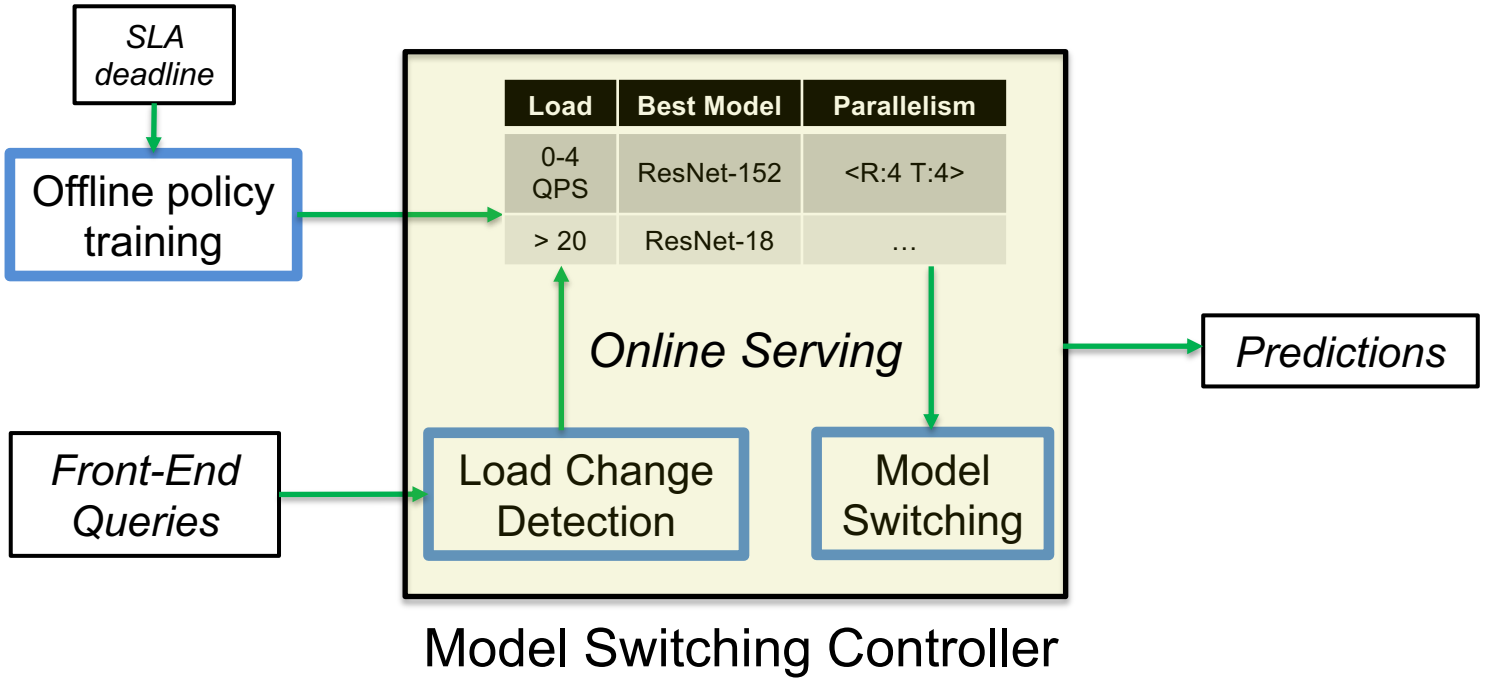
Fixed Capacity:
16 threads

End to End
Query Latency

As load increases, additional replicas help more than threads.

Online Model Switching Framework

Model that exhibits best effective accuracy is a function of load

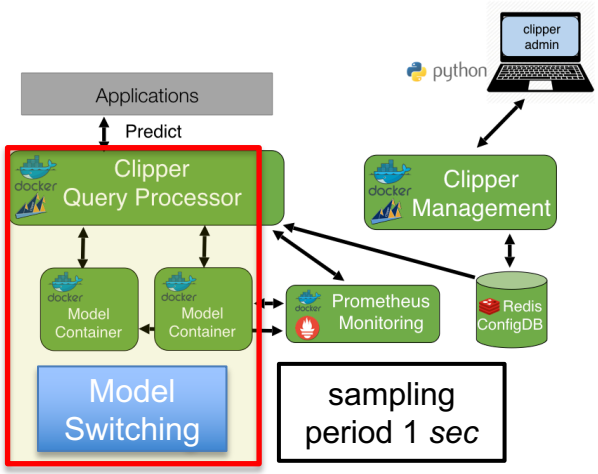


Model Switching Controller

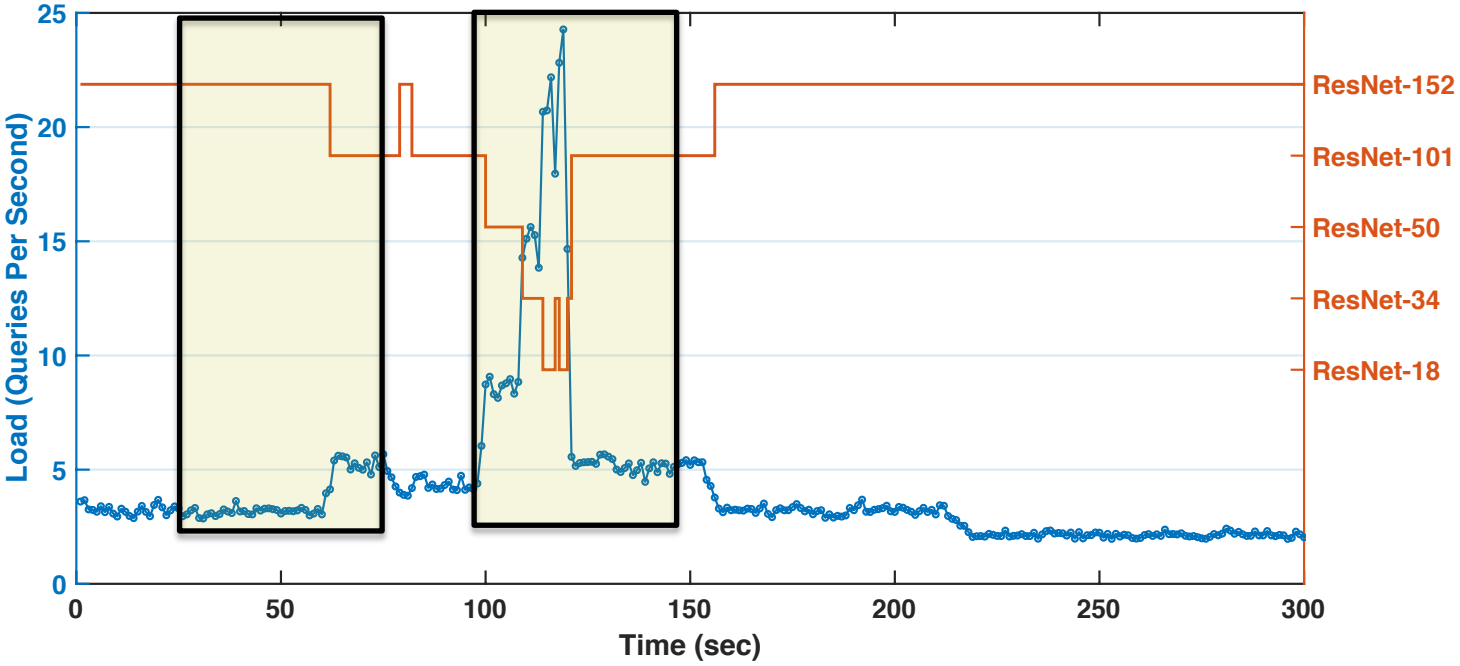
Dynamically select best model (effective accuracy) based on load

Experimental Setup

- Built on top of *Clipper*, an open-source containerized model-serving framework (caching, adaptive batching disabled)
- Deployed PyTorch pretrained ResNet models on ImageNet (*R*: 4 *T*: 4)
- Two dedicated Azure VMs:
  - Server: 32 vCPUs + 128GB RAM
  - Client: 8vCPUs + 32GB RAM
- Markov Model based load generator
  - Open system model
  - Poisson inter-arrivals

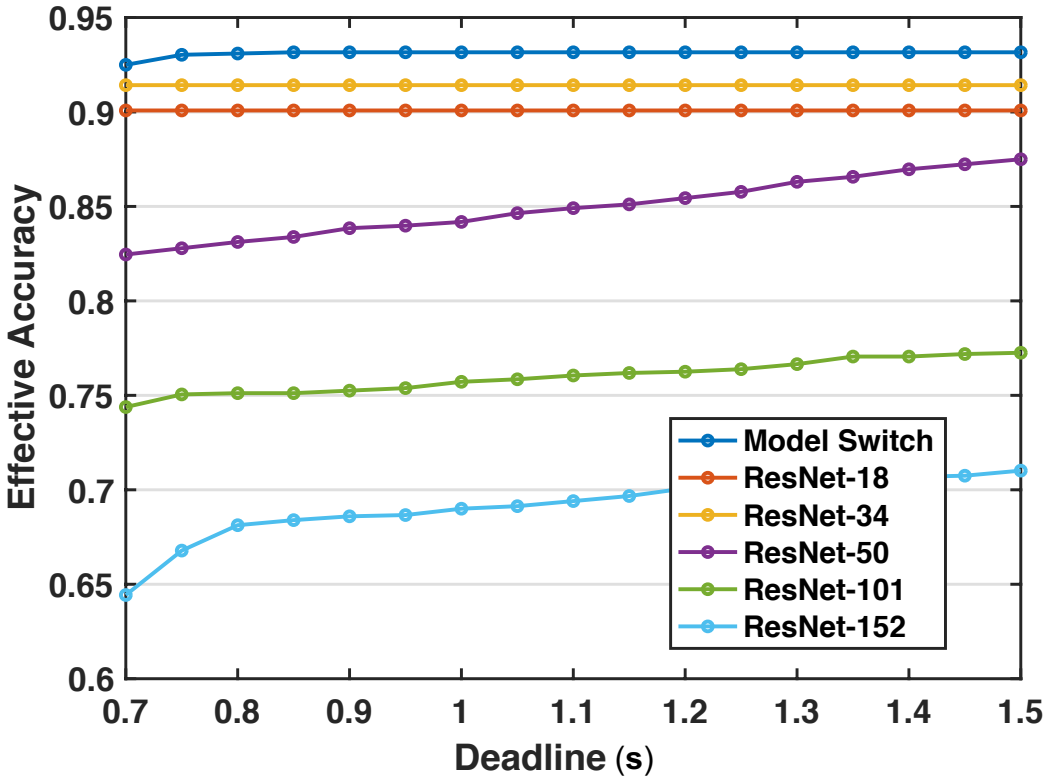


Evaluation: Automatic Model-Switching



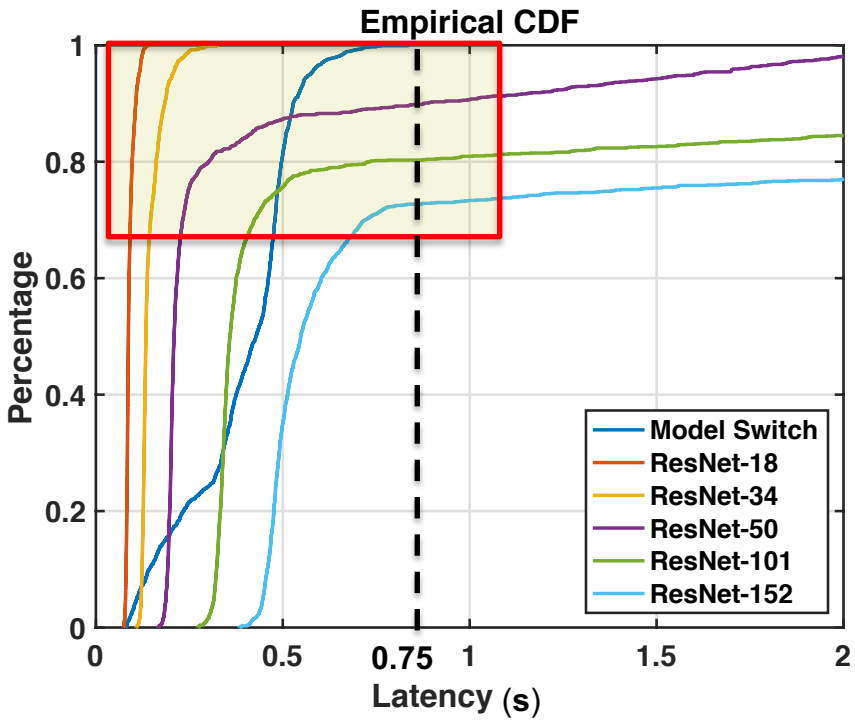
Model-Switching can quickly adapt to load spikes.

Evaluation: Effective Accuracy



Model-Switching achieves pareto-optimal effective accuracy.

Evaluation: Tail Latency



SLA deadline:
750 ms

Model-Switching tradeoffs deadline slack for accuracy.

Thank You and Questions

Model-Switching: Manage Fluctuating Workloads in MLaaS Systems

Jeff Zhang
jeffjunzhang@nyu.edu

Discussion and Future Work

- How to prepare a pool of models for each application?
  - Neural Architecture Search, Multi-level Quantization
- Current approach pre-deploys all (20) candidate models
  - Cold start time (ML): tens of seconds
  - RAM overheads: currently 11.8% of the total 128 GB RAM
- Reinforcement learning based controller for model switching
  - Account for job queue status, system load, current latency
  - Offline training free
- Integrate with existing MLaaS techniques
  - Batching, caching, autoscaling *etc.*
- Exploit availability of heterogenous computing resources
  - CPU, GPU, TPU, FPGA