Systems support for ML: It's all about the choices

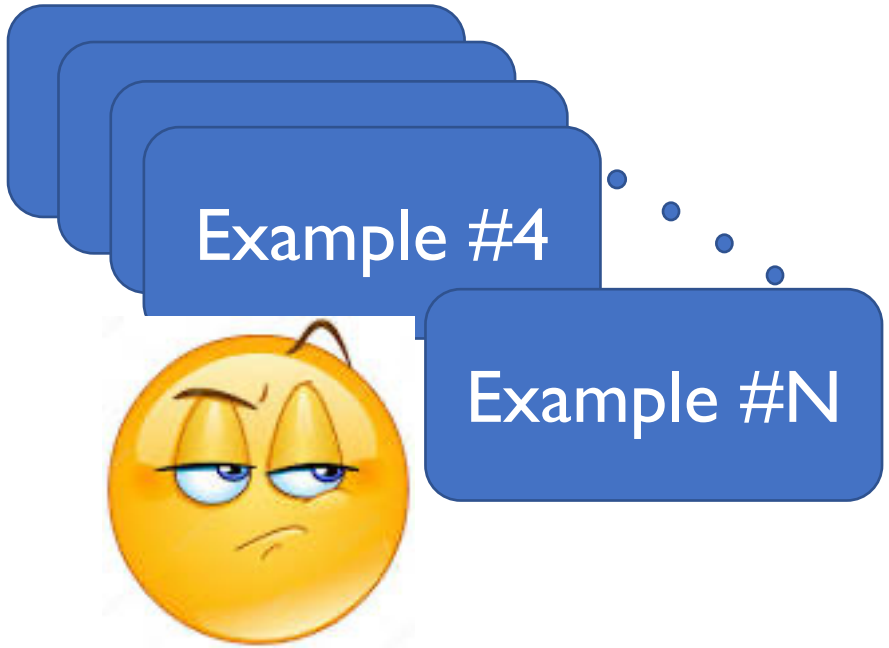
Neeraja J. Yadwadkar
UT Austin, ECE
Nov 8th, 2022

With collaborators from Stanford and UT Austin

What is ML?

$$1 - \frac{1}{1+\sqrt{3}} + \frac{1}{1-\sqrt{3}}$$
$$= 1 - \frac{1-\sqrt{3}}{(1+\sqrt{3})(1-\sqrt{3})} + \frac{1+\sqrt{3}}{(1-\sqrt{3})(1+\sqrt{3})}$$
$$= 1 - \frac{1-\sqrt{3}}{1^2-(\sqrt{3})^2} + \frac{1+\sqrt{3}}{1^2-(\sqrt{3})^2}$$
$$= 1 - \frac{1-\sqrt{3}}{1-3} + \frac{1+\sqrt{3}}{1-3}$$
$$= 1 - \frac{1-\sqrt{3}}{-2} + \frac{1+\sqrt{3}}{-2}$$
$$= 1 + \frac{1-\sqrt{3}}{2} - \frac{1+\sqrt{3}}{2}$$
$$= \frac{2+1-\sqrt{3}-1-\sqrt{3}}{2}$$
$$= \frac{2-2\sqrt{3}}{2}$$
$$= \frac{2(1-\sqrt{3})}{2}$$
$$= 1-\sqrt{3}$$

✓ 10



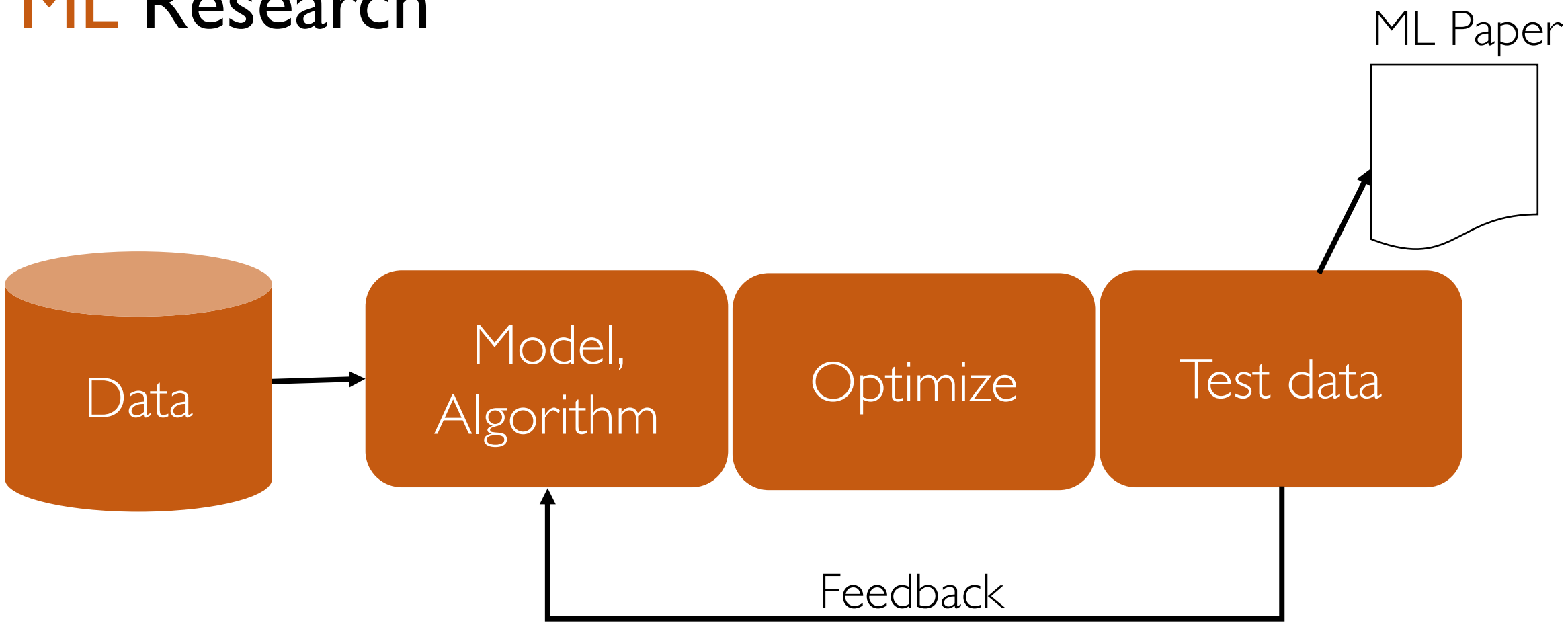
A dumb algorithm with lots and lots of data beats a clever one with a modest amount of it!*

* A Few Useful Things to Know About Machine Learning, CACM'12

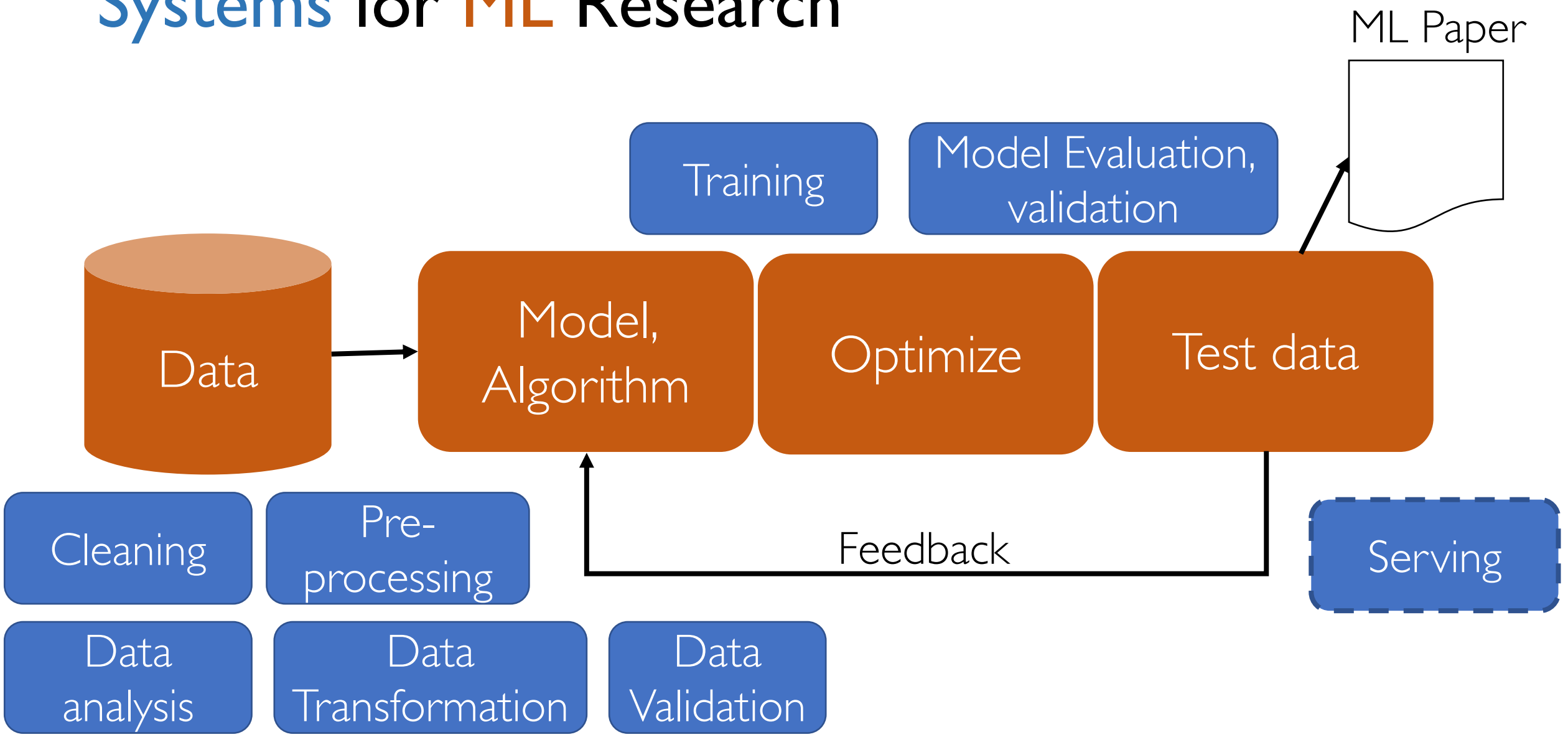
Why is ML research flourishing?

- Availability of data
- Availability of compute
- Advances in algorithms and models
- Abstractions and Interfaces!

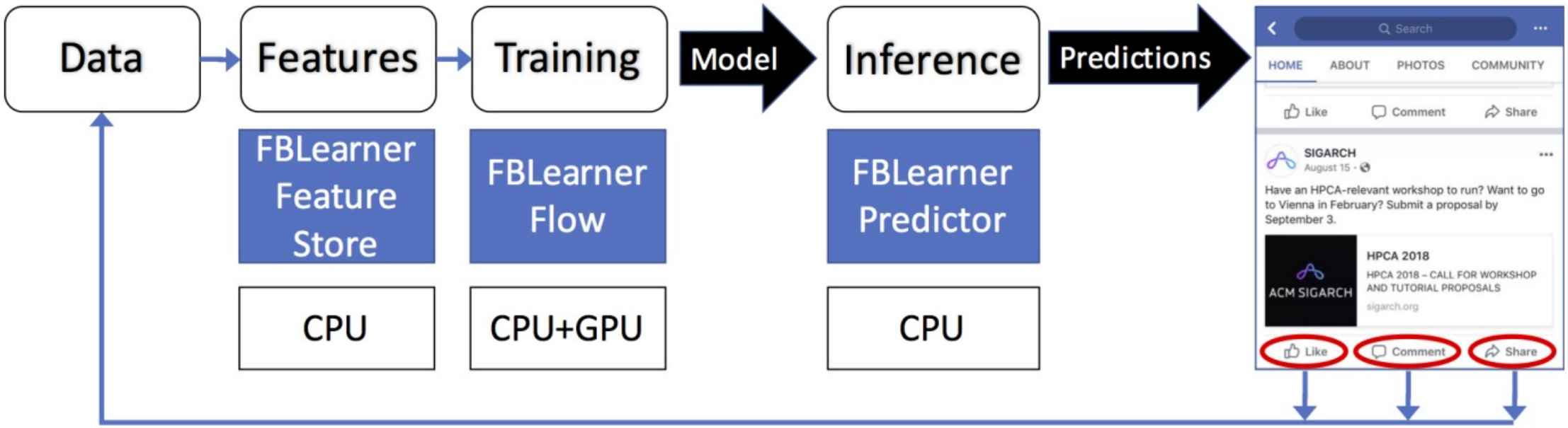
ML Research



Systems for ML Research

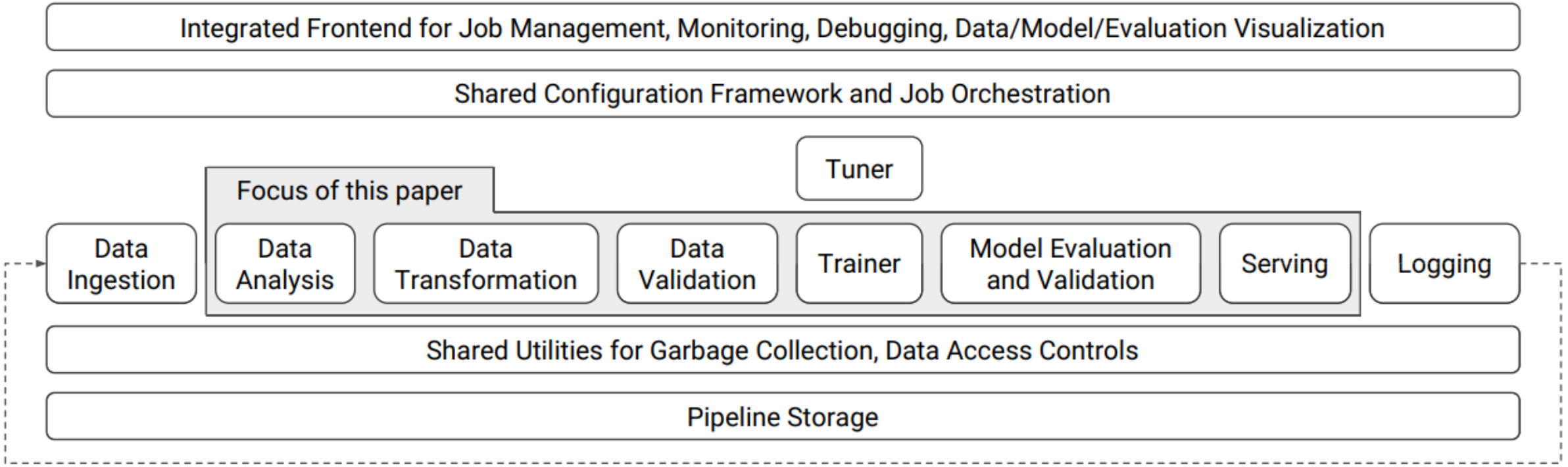


Facebook's pipeline



Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective, HPCA'18

Google's pipeline



TFX: A TensorFlow-Based Production-Scale Machine Learning Platform, KDD'17

Design Complexity

Machine Learning

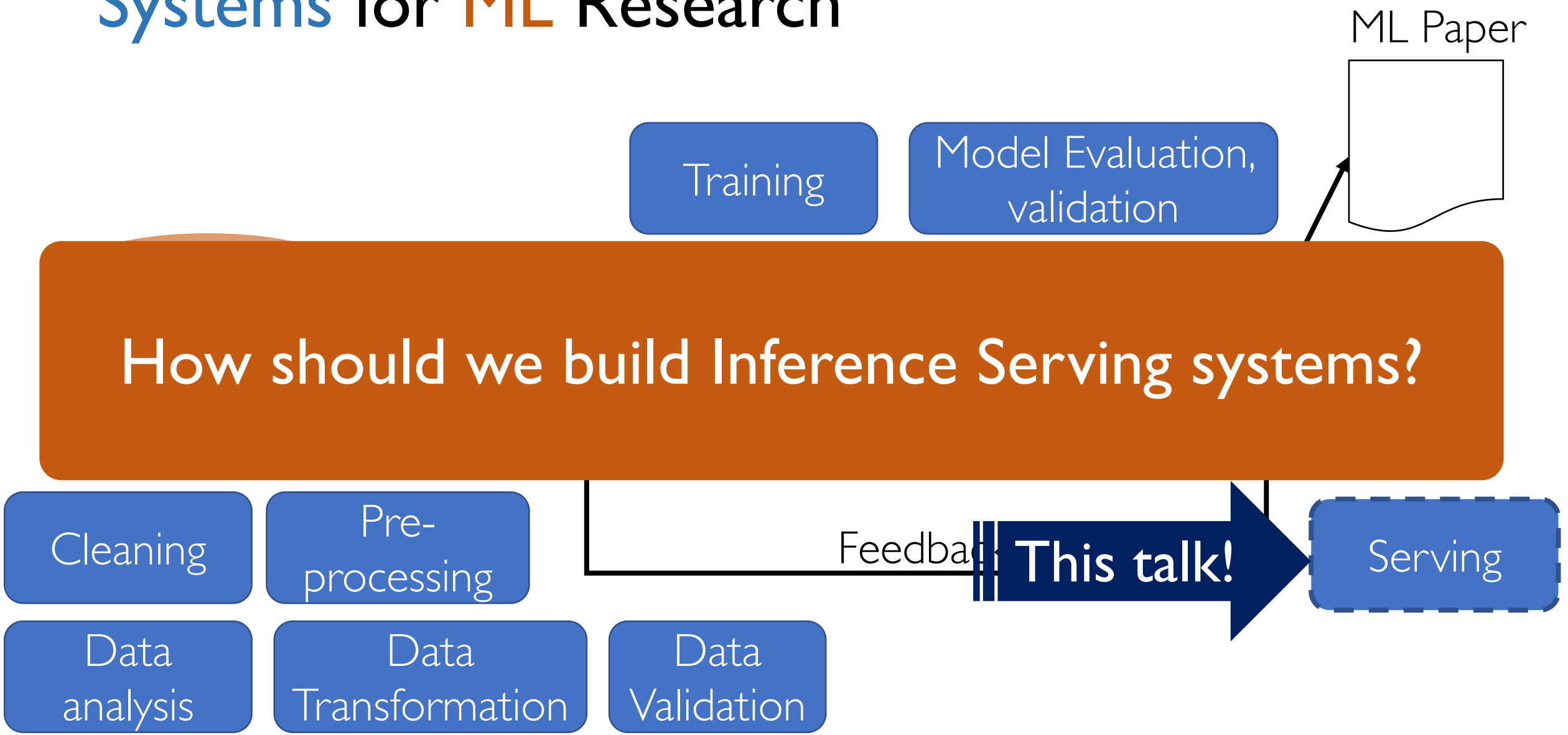
- Model
- Feature engineering
- Training
- Bias
- Overfitting
- Generalizability
- Accuracy

Systems

- Scheduling
- Resource allocation
- Locality
- Fault tolerance
- Power efficiency
- Reliability/Availability
- Security

Systems for ML research brings these two sets of complexities together

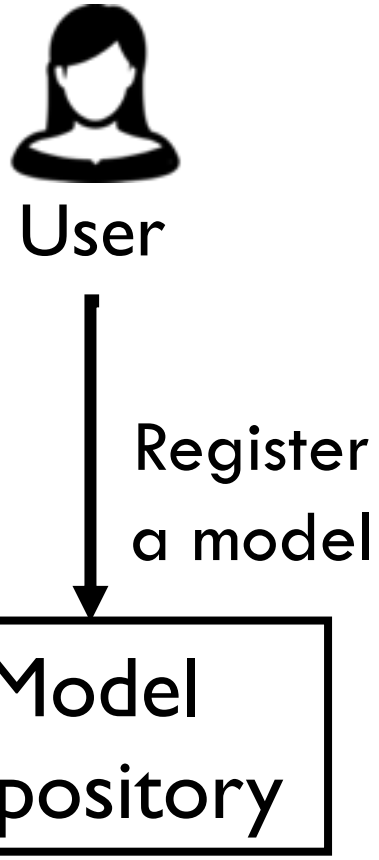
Systems for ML Research



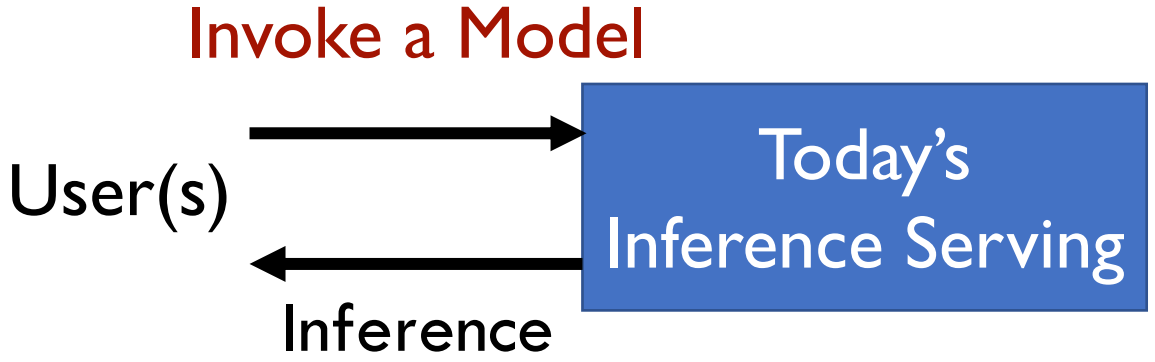
Outline

- What is ML?
- ML Workflow
- Systems for ML: Design Complexity
- Managed Inference Serving by INFaaS (Deep dive)
- Inference Pipelines using Llama
- Sparsely Activated Massive Models

Model Registration



Today's Inference serving



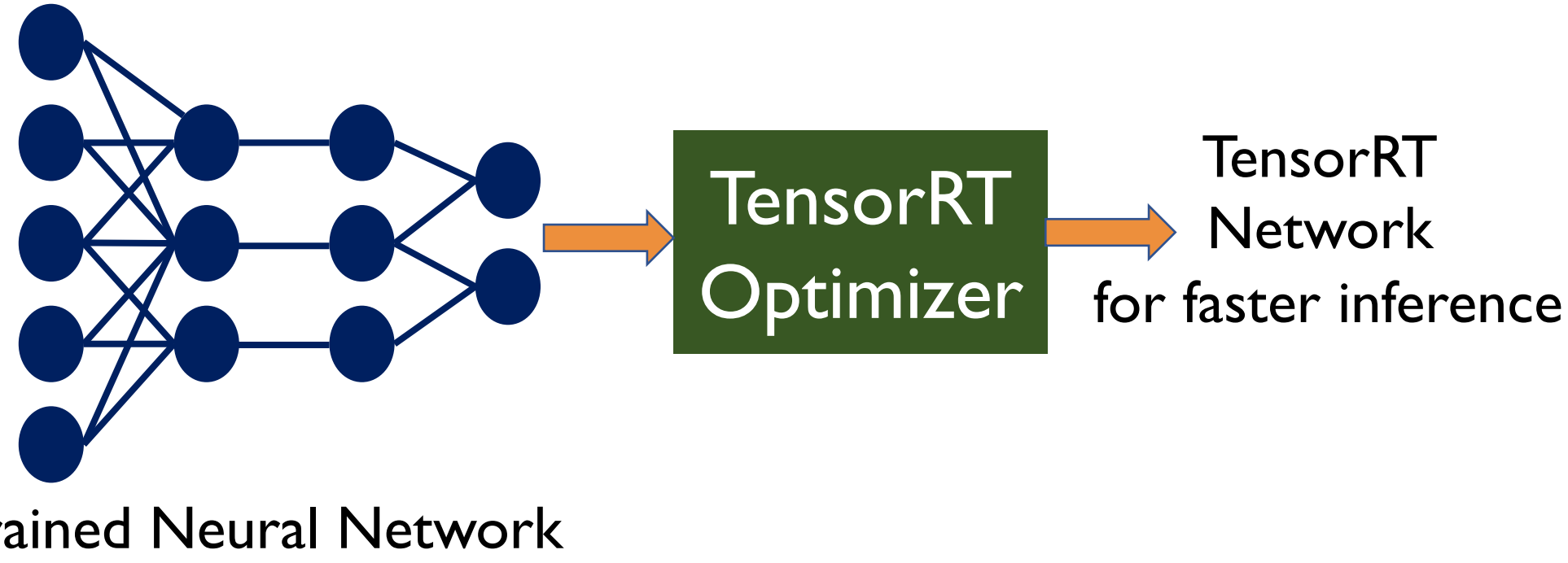
Same model,

multiple *Model-Variants*!

Same model, multiple *Model-Variants!*

Compiler optimizations

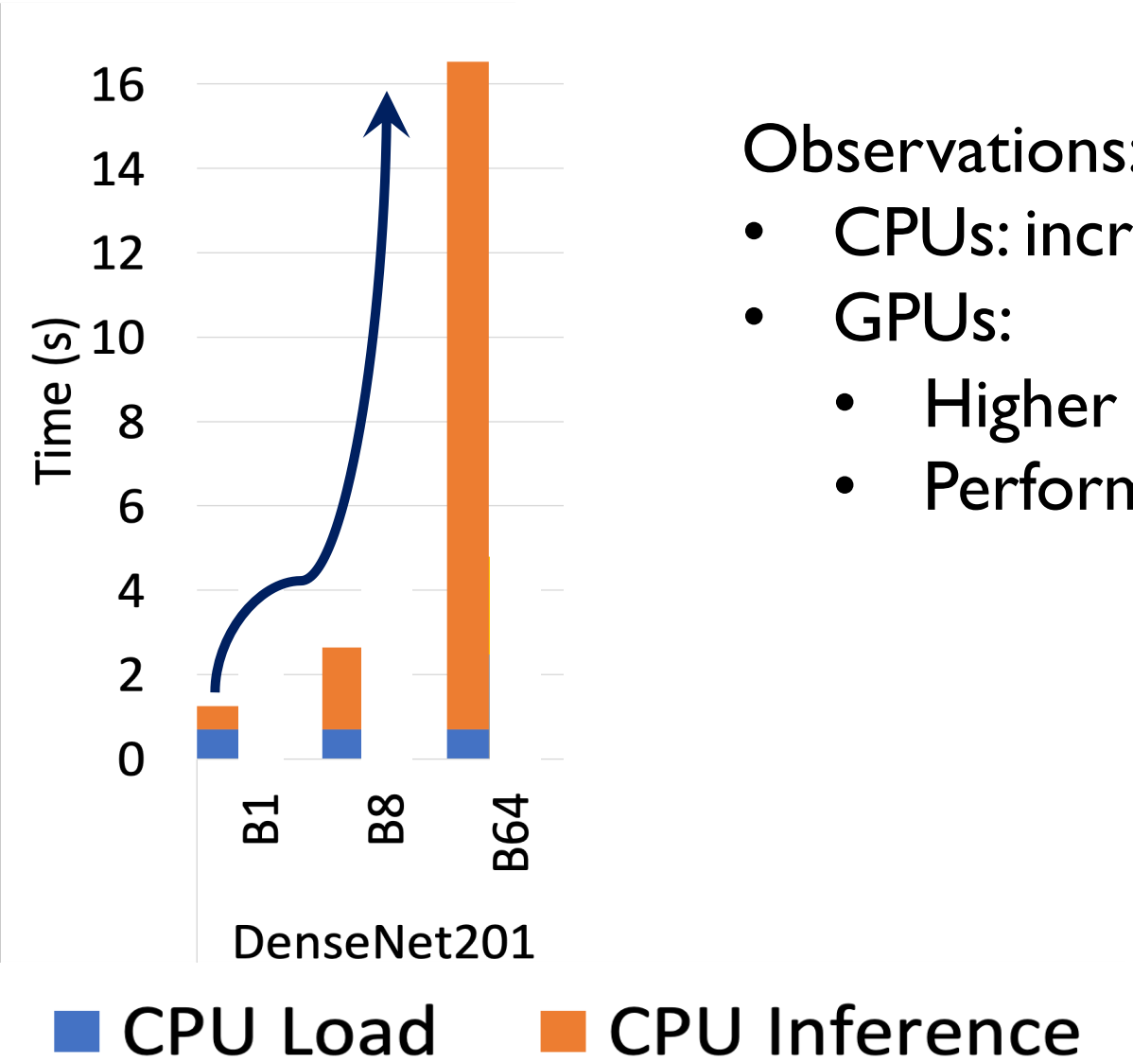
TVM, TensorRT



Same model, multiple *Model-Variants!*

Compiler optimizations	Different precisions	Hyperparameter optimizations	Heterogeneous Hardware
TVM, TensorRT	INT8, FP16, FP32	Batch size	CPUs, GPUs, …, *PUs

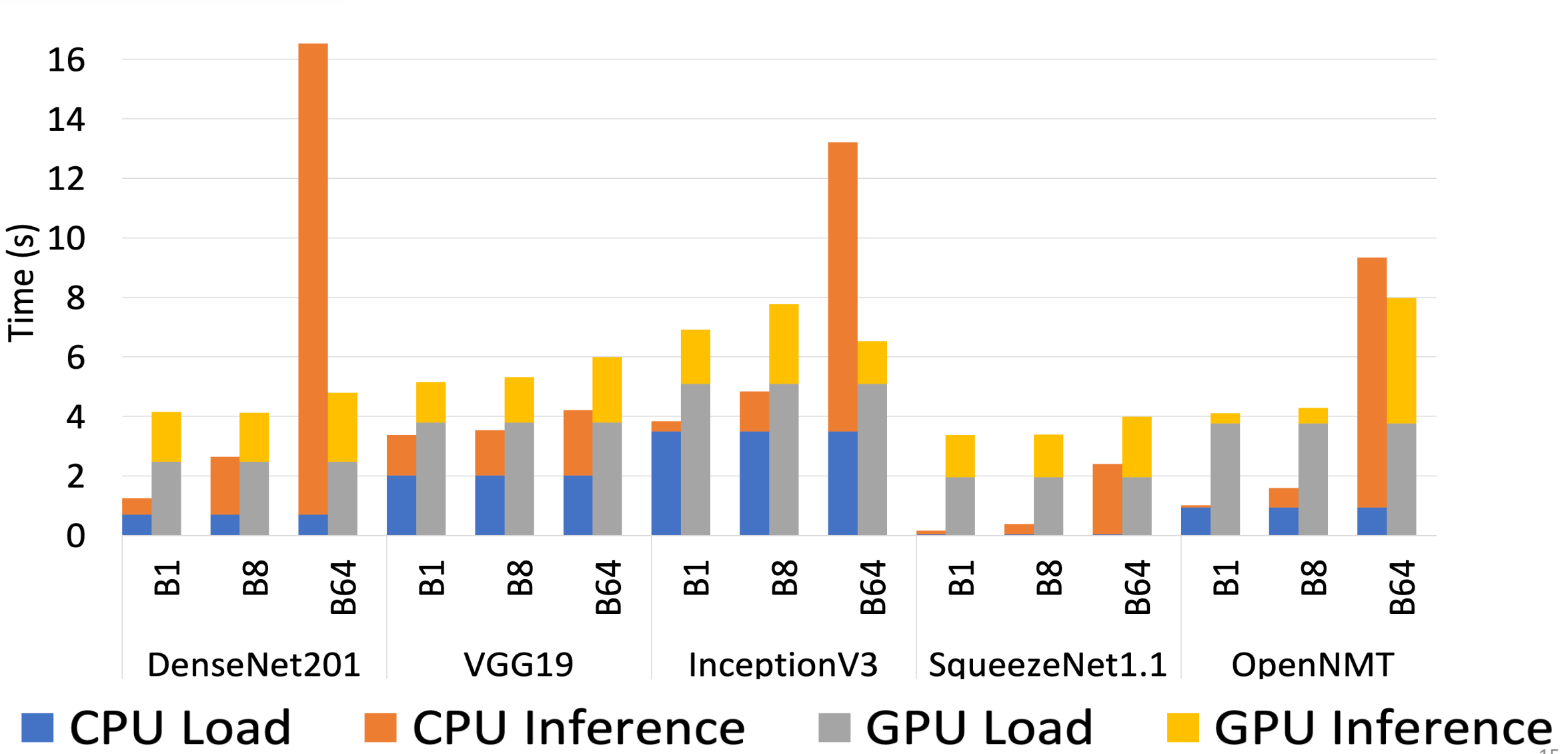
Trade-offs for inference due to heterogeneous hardware



Observations:

- CPUs: increased inference latency with batch-size
- GPUs:
  - Higher loading latencies
  - Perform significantly better on high batch sizes

Trade-offs for inference due to heterogeneous hardware



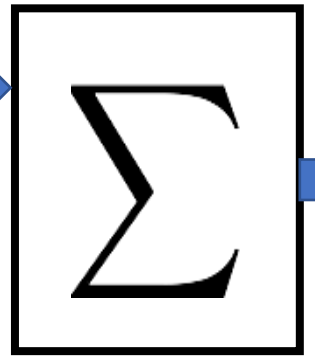
Same model, multiple *Model-Variants!*

Compiler optimizations	Different precisions	Hyperparameter optimizations	Heterogeneous Hardware
TVM, TensorRT	INT8, FP16, FP32	Batch size	CPUs, GPUs, …, *PUs

We can compile the same model to 10s (100s) of versions

Diverse application requirements

Object Detection



Face Recognition

Example: Face Recognition

Accuracy

Latency

Social Media

Latency

Accuracy

Cost

Navigation for visually impaired person

Default user choices

- Overprovision:
  - Use dedicated resources
  - Keep the models “always on”
  - Replicate a query across multiple models

Model-Variants: Challenge, but an untapped Opportunity!

Challenge

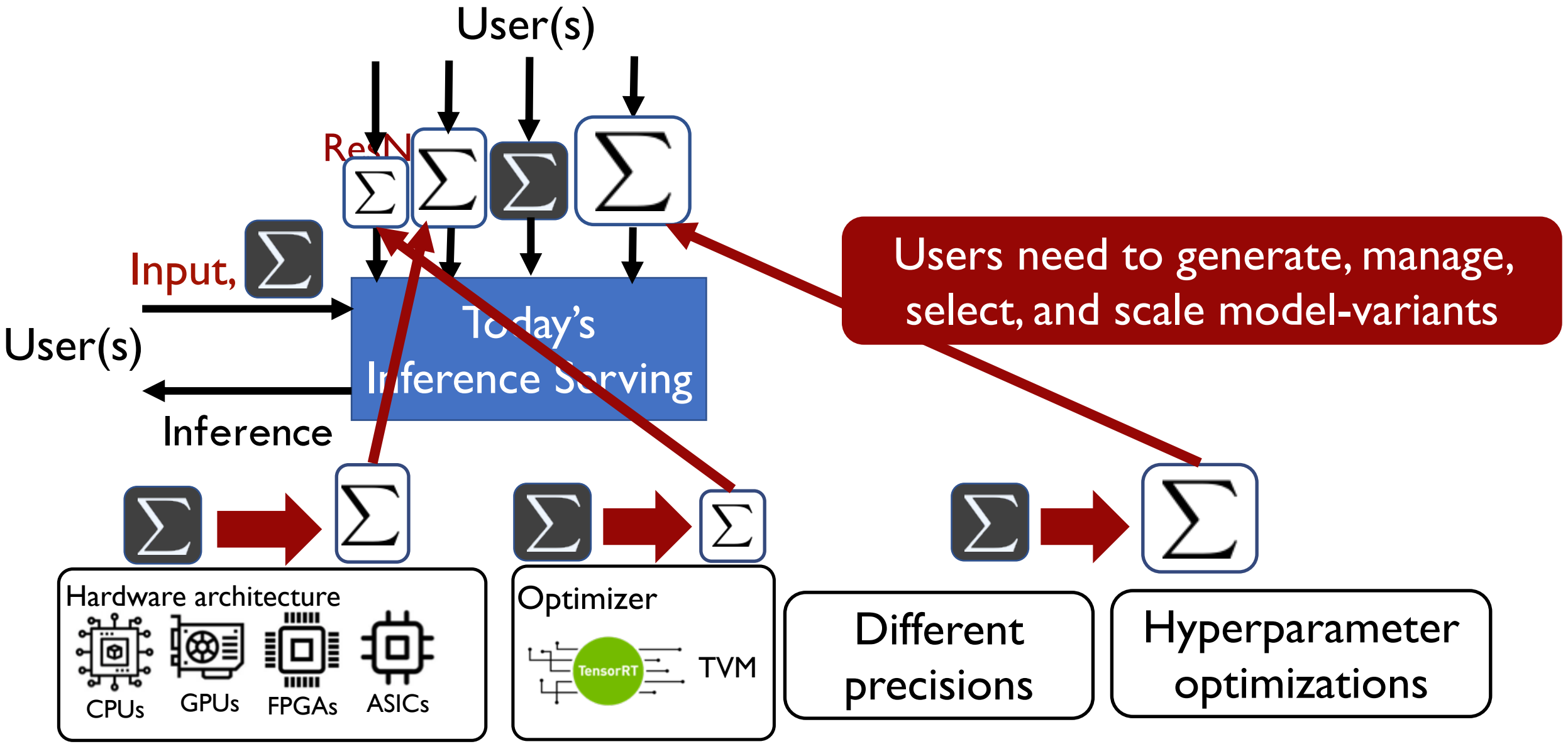
Hardware x
Optimizers x
Precisions x
Hyperparameters
→ a large search space

Opportunity

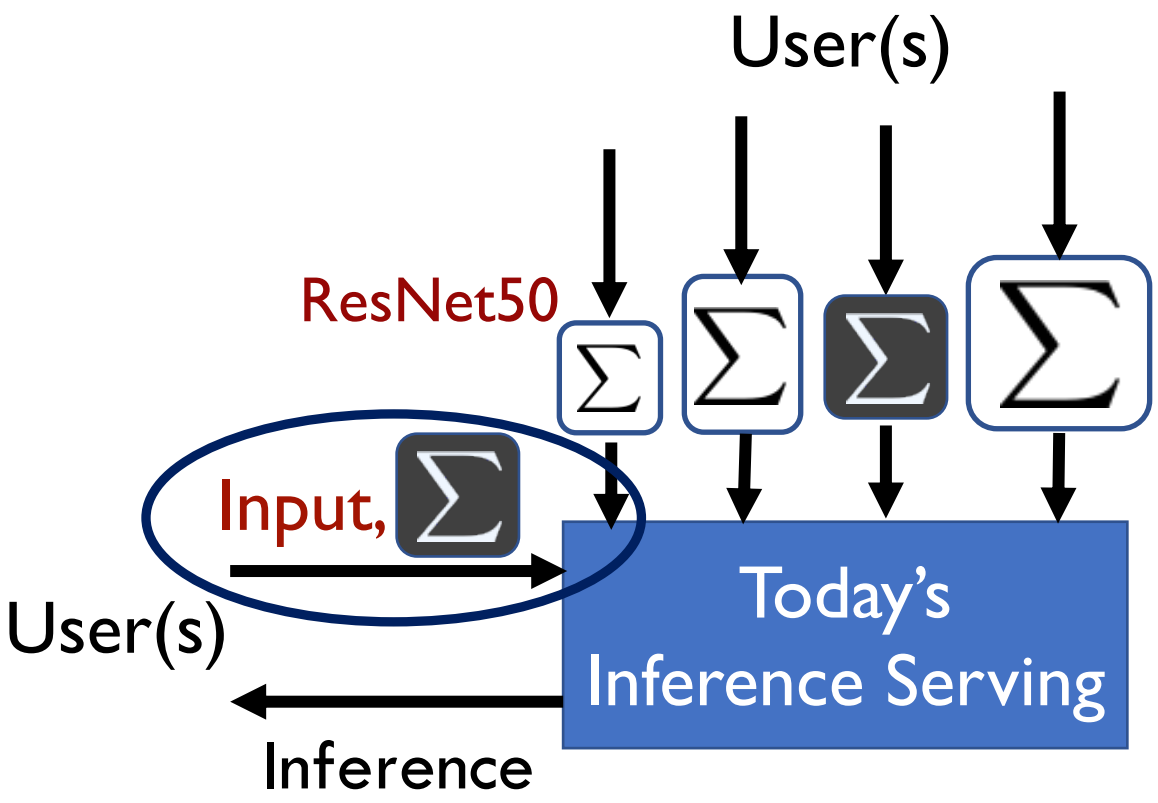
A large trade-off space of Latency, Throughput, Accuracy, Resources required, and Cost

Our Proposal:
Model-less
Inference Serving

Today's Inference serving

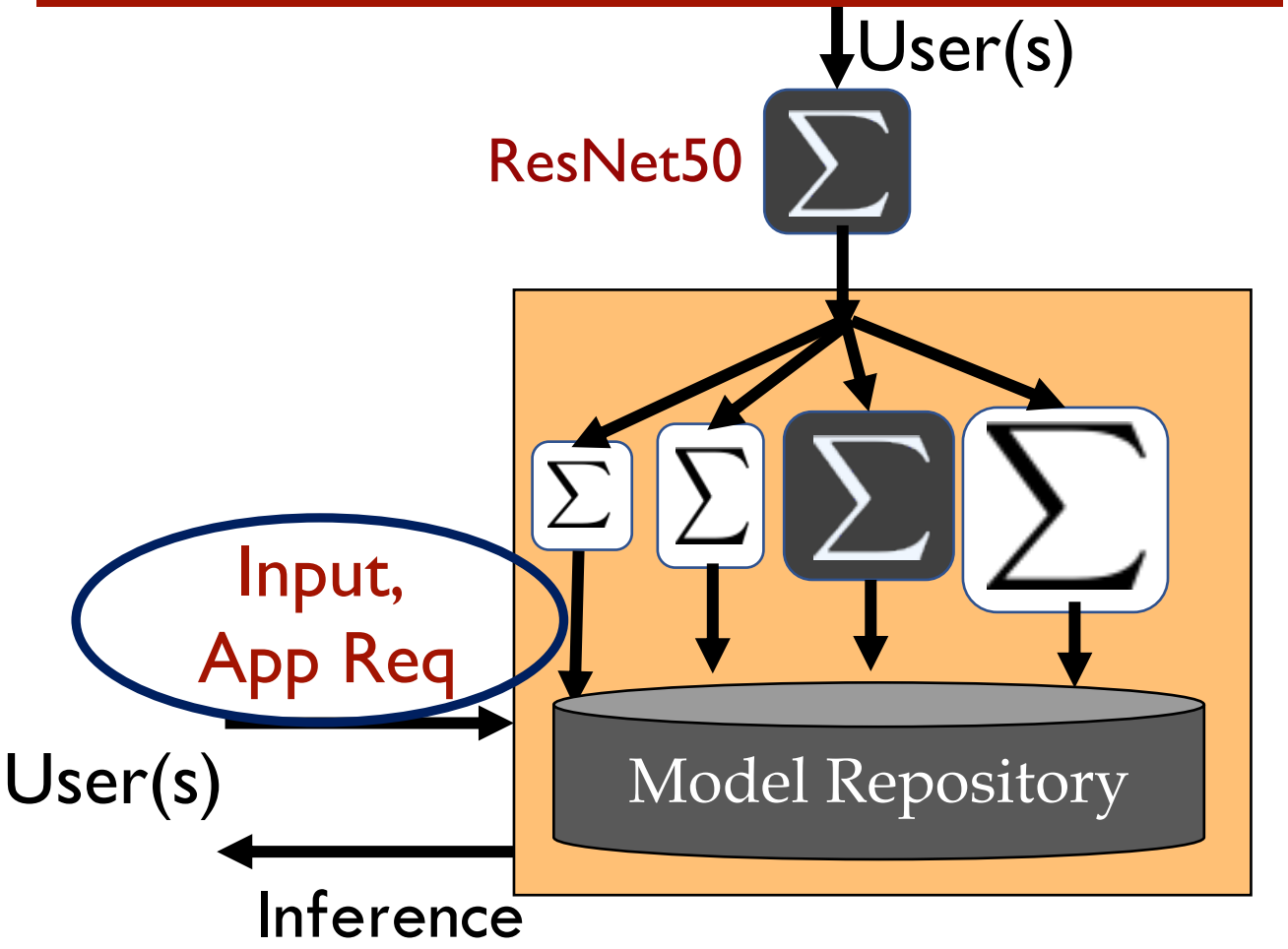


Today's Inference serving

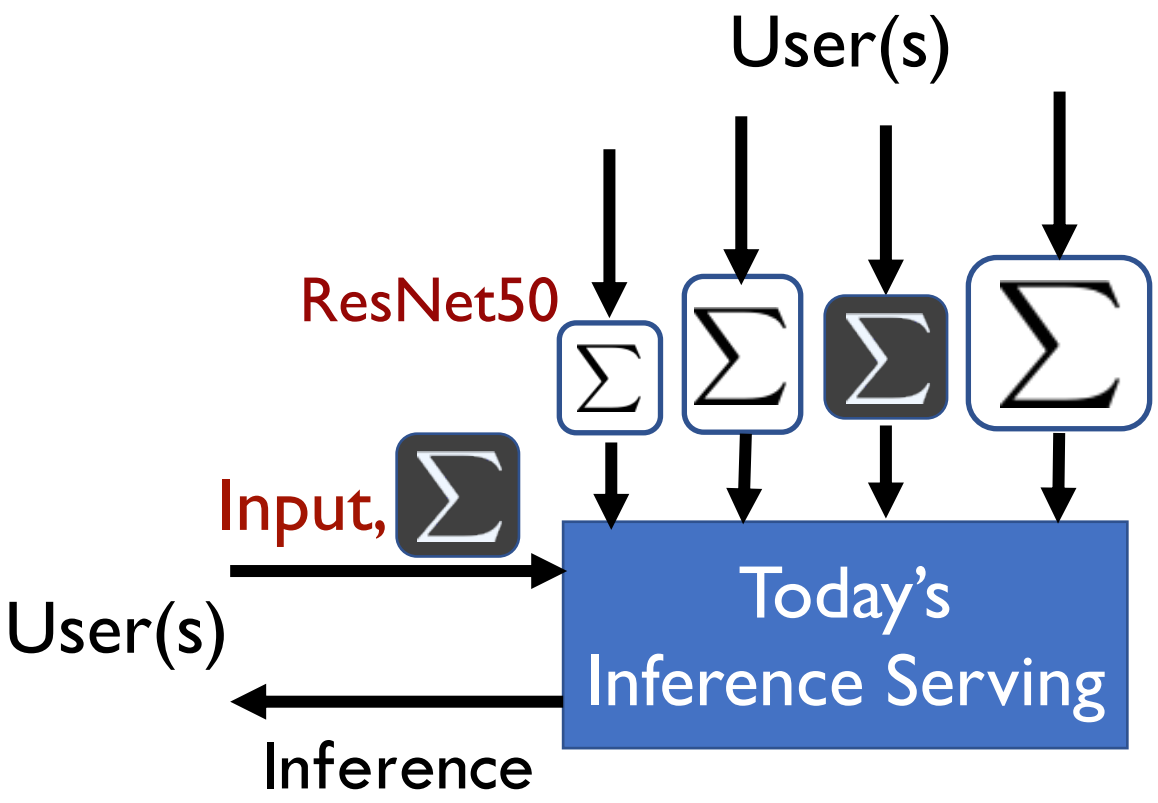


Users need to generate, manage, select, and scale model-variants

Model-less Inference serving

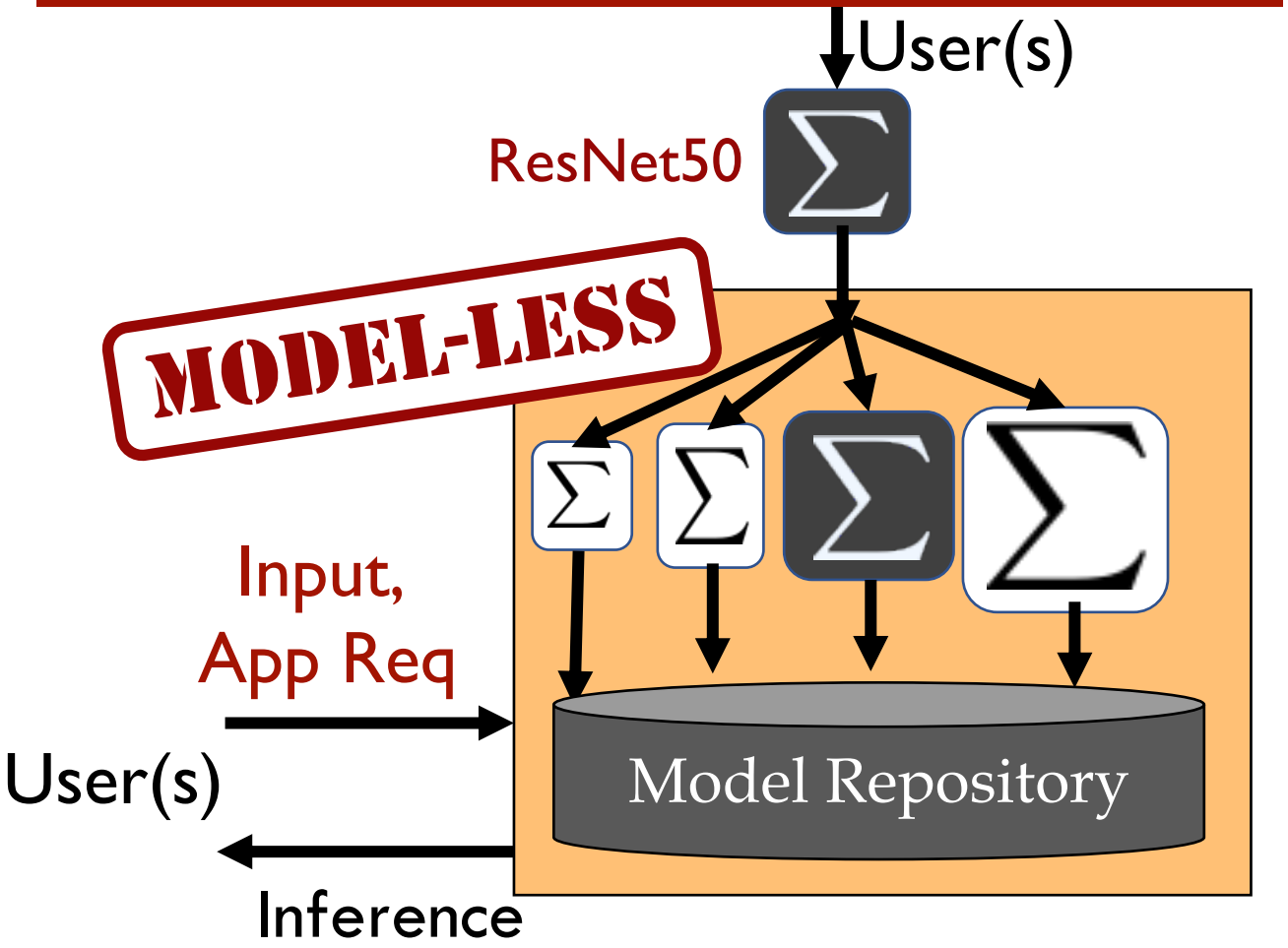


Today's Inference serving



Users need to generate, manage, select, and scale model-variants

Model-less Inference serving



Key: Automatically and efficiently selecting and scaling model-variants

INFaaS: A Model-less Inference Serving System

- No models to generate and manage for users
- Automatic selection of the right model-variant for each query
- Autoscaling to respond to the changes in query load

INFaaS' Model-less API

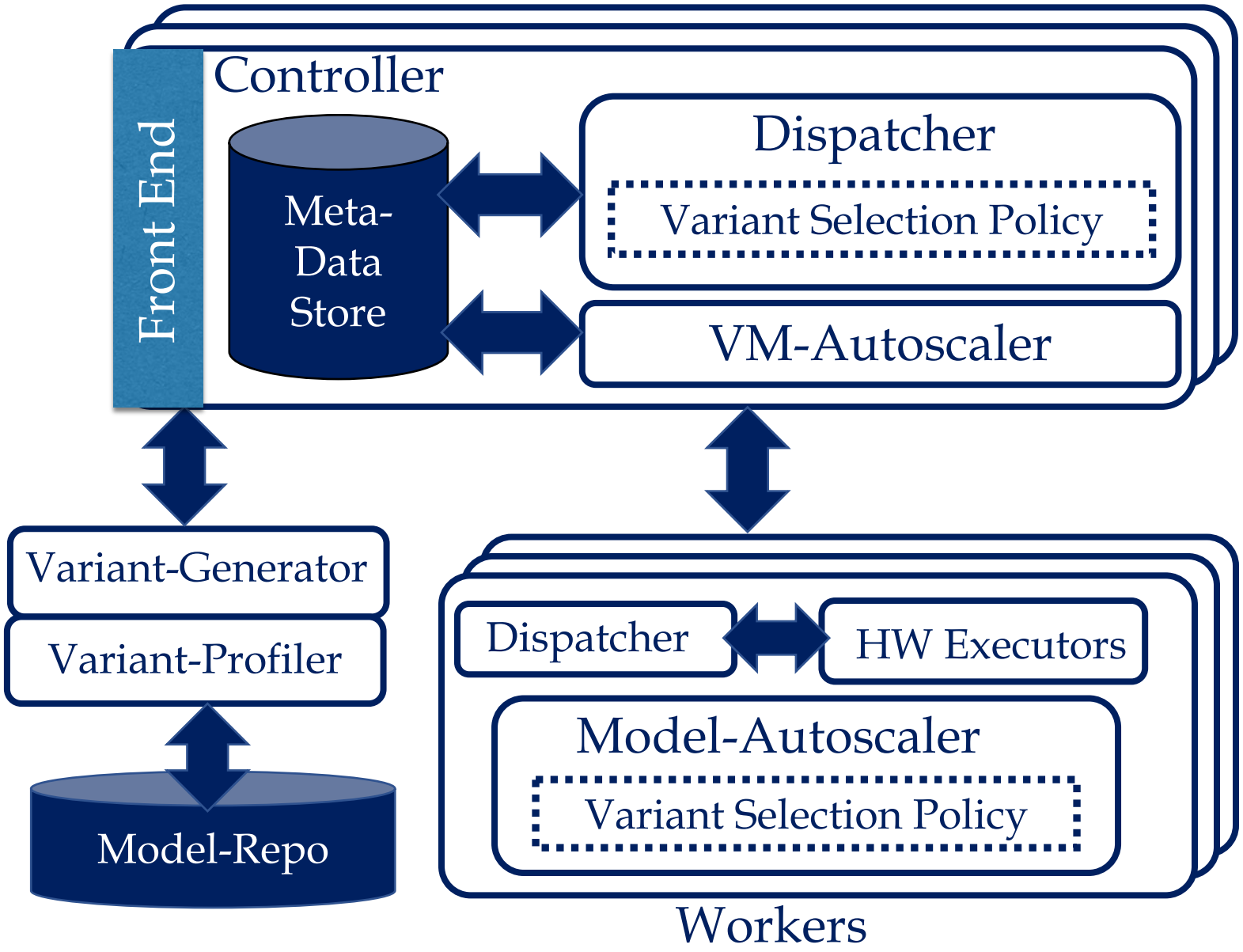
INFaaS' Model Registration API

```
Register_model("ResNet50",
               ResNet50.pt,
               validationSet,
               detectFaceApp)
```

INFaaS' Query Submission API

```
query(input.jpg,
      detectFaceApp,
      latency=200ms,
      accuracy=70%)
```

INFaaS: Architecture Overview



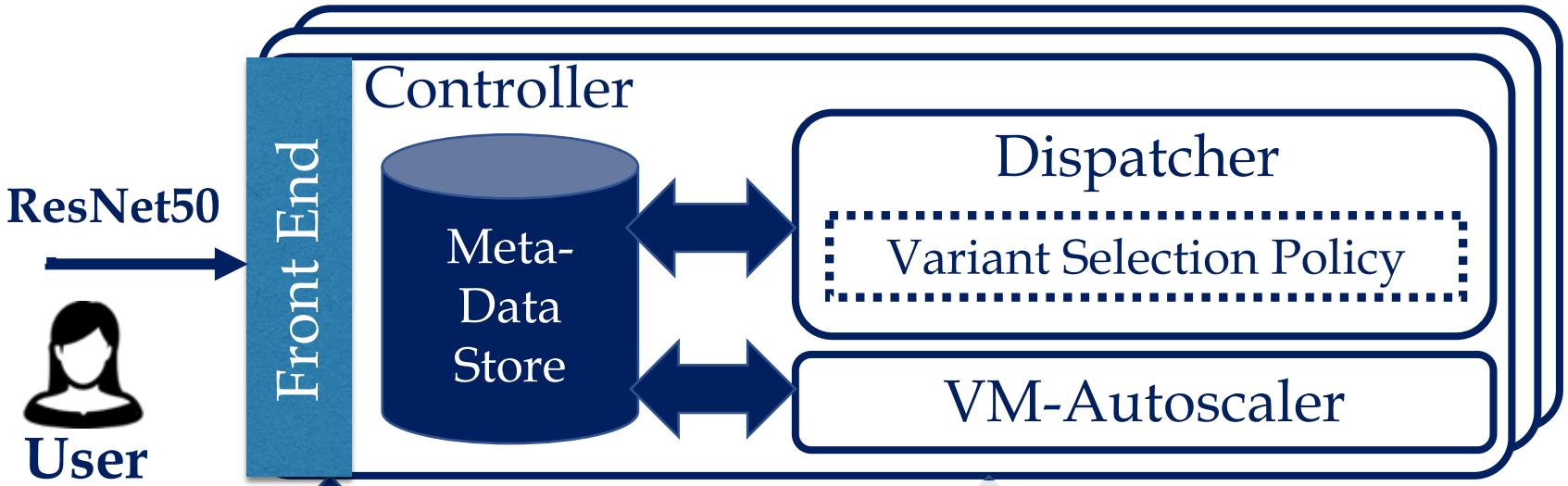
Three Workflows:

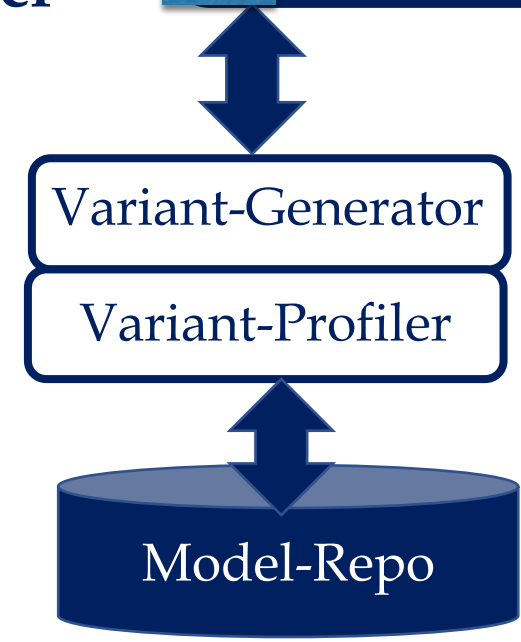
(a) Model Registration

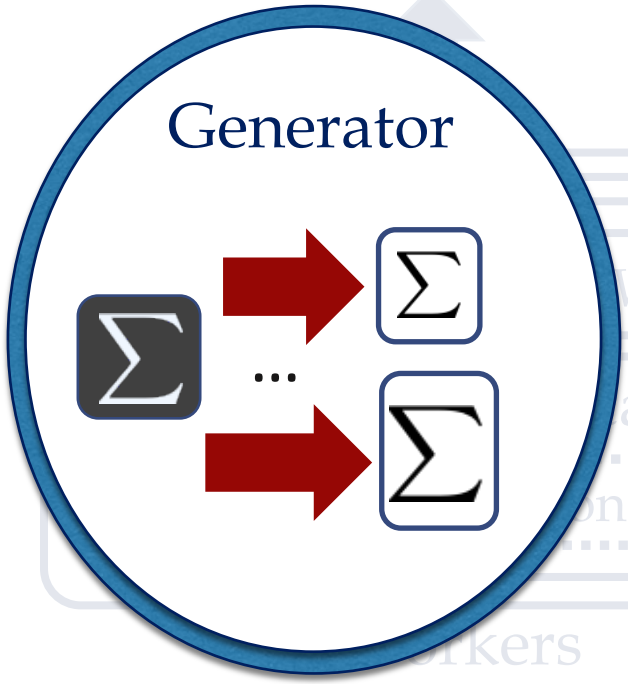
(b) Query Execution

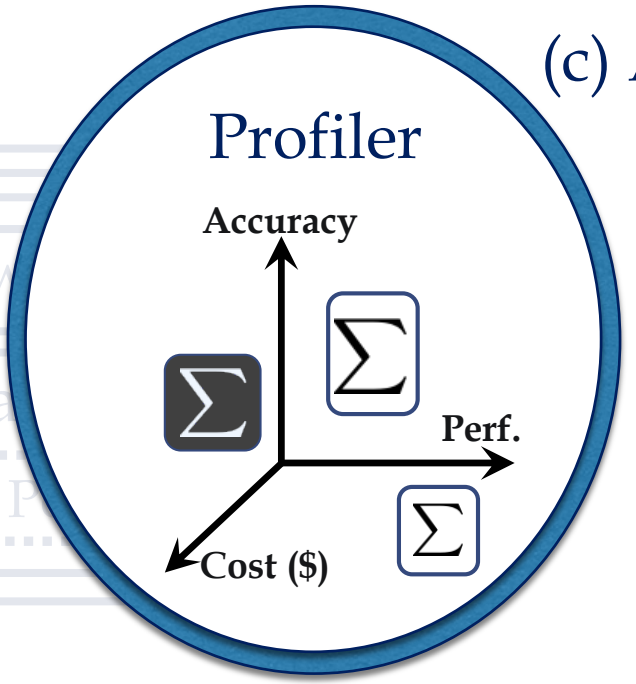
(c) Autoscaling

INFaaS: Model Registration Workflow









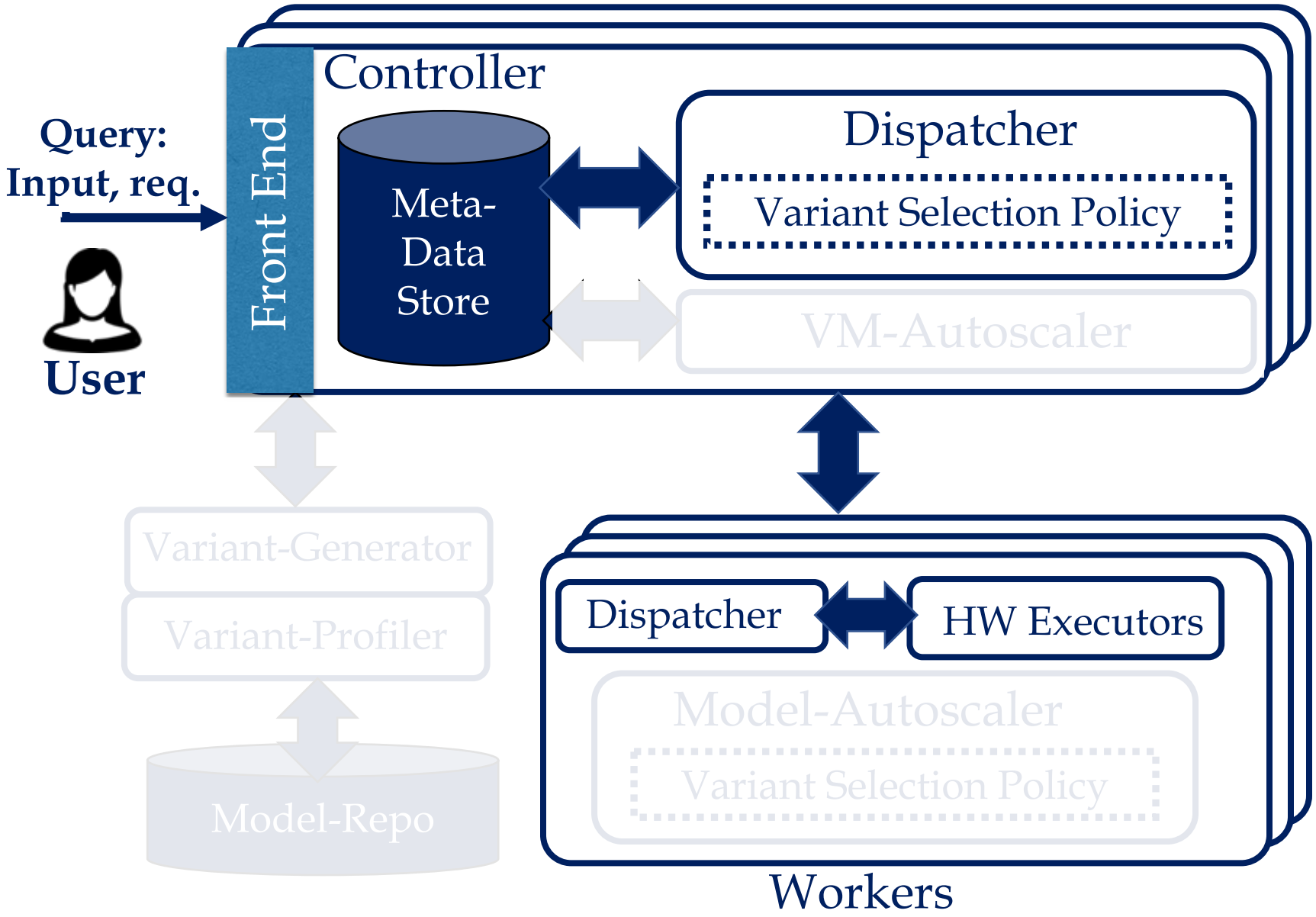
Three Workflows:

(a) Model Registration

(b) Query Execution

(c) Autoscaling

INFaaS: Query Execution Workflow



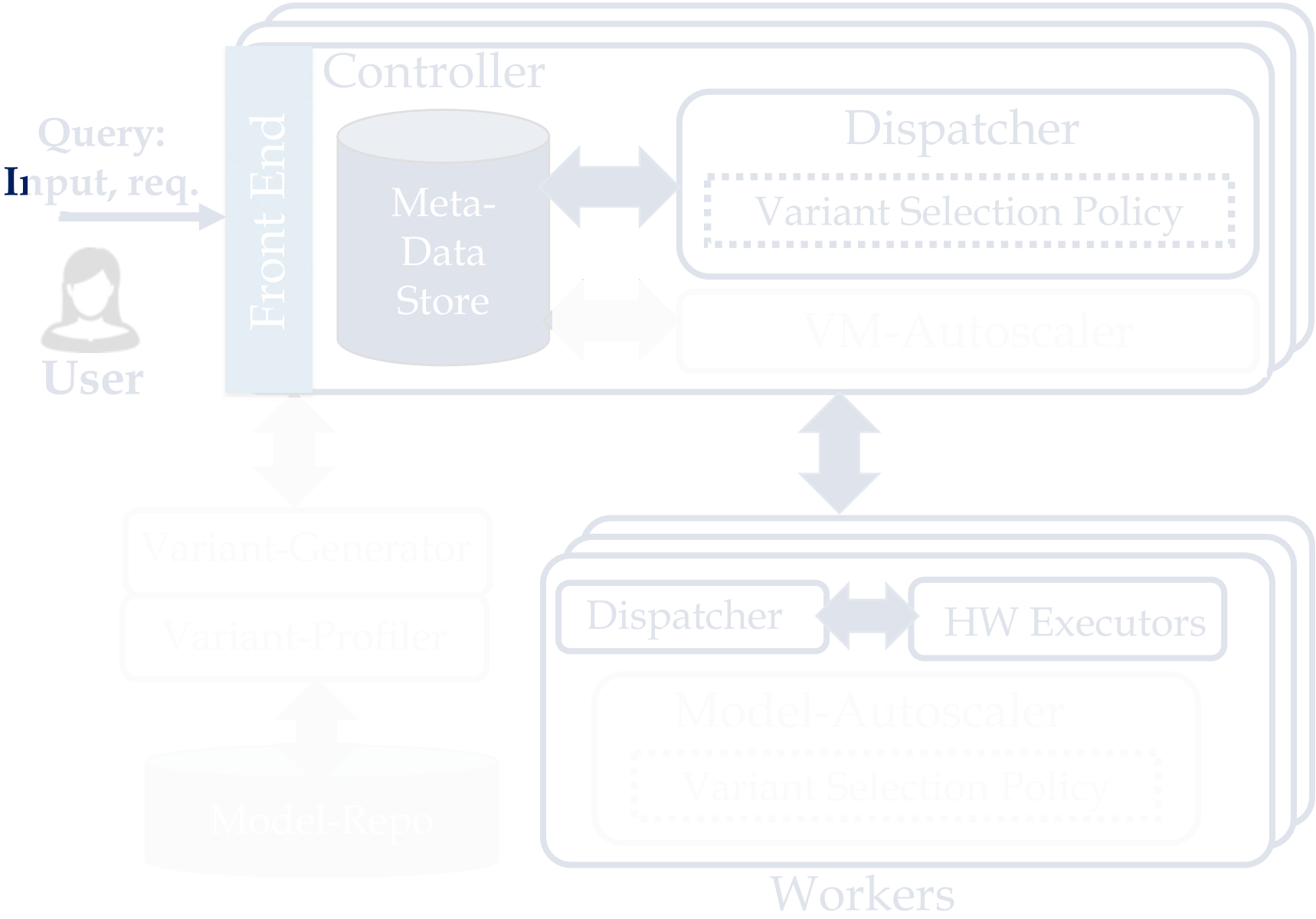
Three Workflows:

(a) Model Registration

(b) Query Execution

(c) Autoscaling

INFaaS: Query Execution Workflow



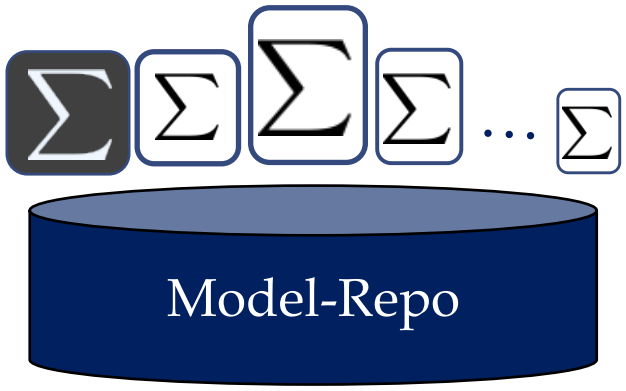
- Least-loaded
- Interference-aware

Selecting a Variant

Goal

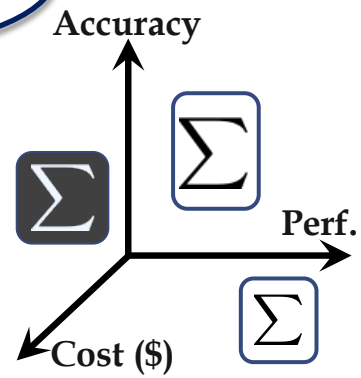
Query Requirements

map



Challenge

- Just Static profile of variants isn't enough



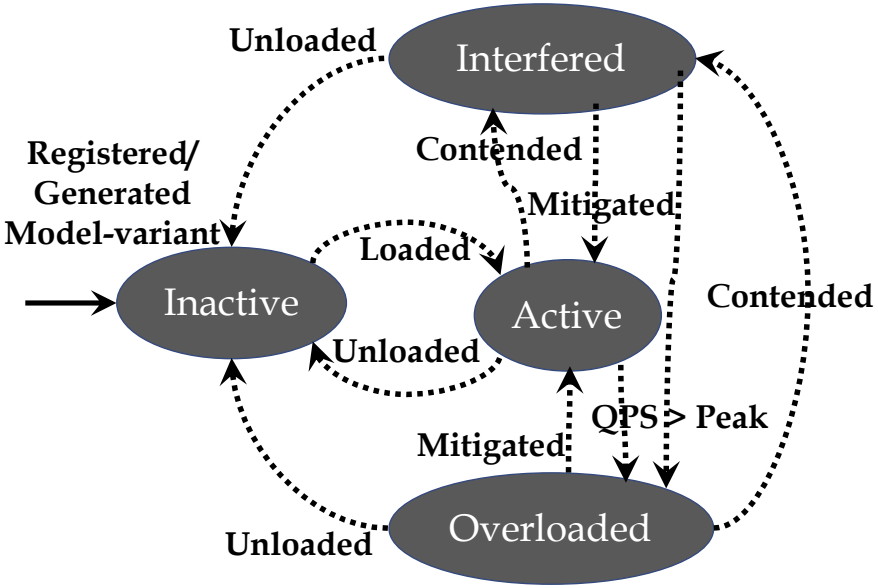
  - Selected variant may not be loaded
  - Selected variant may be overloaded or interfered
  - System may not have enough resources

- Choice also depends on the *Dynamic* state of the variants

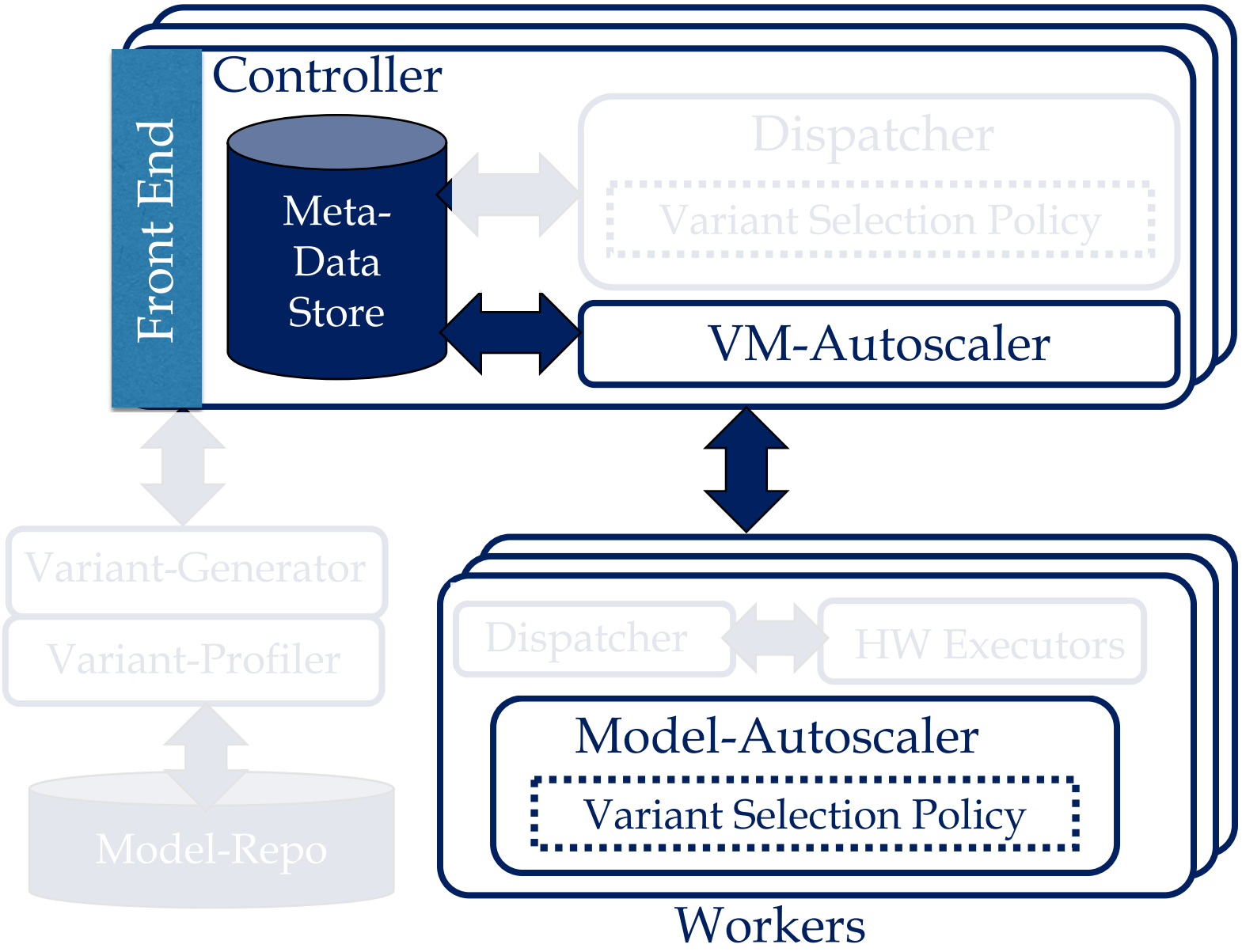
Proposed Solution

Variant-Profiler

State is maintained per variant by the workers at the Metadata-Store



INFaaS: Autoscaling Workflow



Three Workflows:

(a) Model Registration

(b) Query Execution

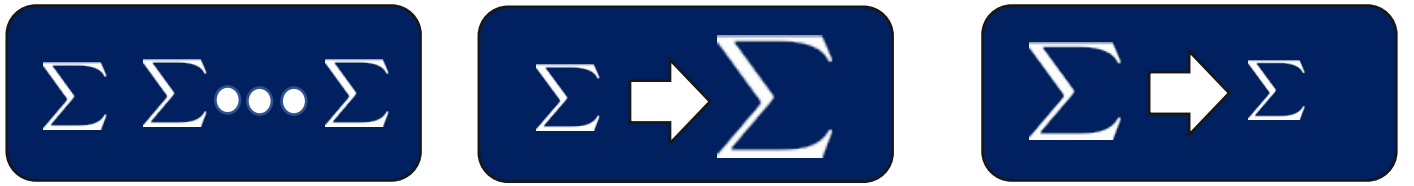
(c) Autoscaling

INFaaS' Model-Autoscaler

- Right variant for each query may be different
- May not have enough resources to replicate

Model horizontal scaling

Model vertical scaling



Replication

Switch

Existing inference Serving systems based on user-specification

Introduced by INFaaS

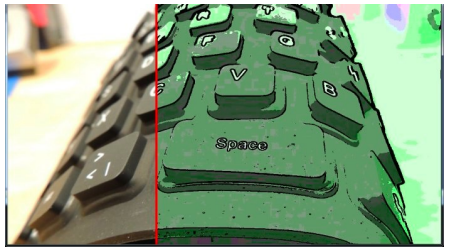
Question: What combination of variants (types and number) is required to support the changed load?

Outline

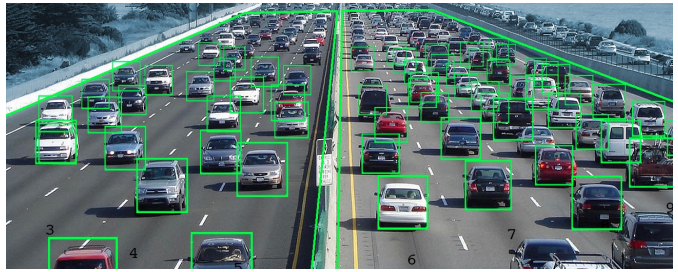
- What is ML?
- ML Workflow
- Systems for ML: Design Complexity
- Managed Inference Serving by INFaaS (Deep dive)
- Inference Pipelines using Llama
- Sparsely Activated Massive Models

Inference pipelines

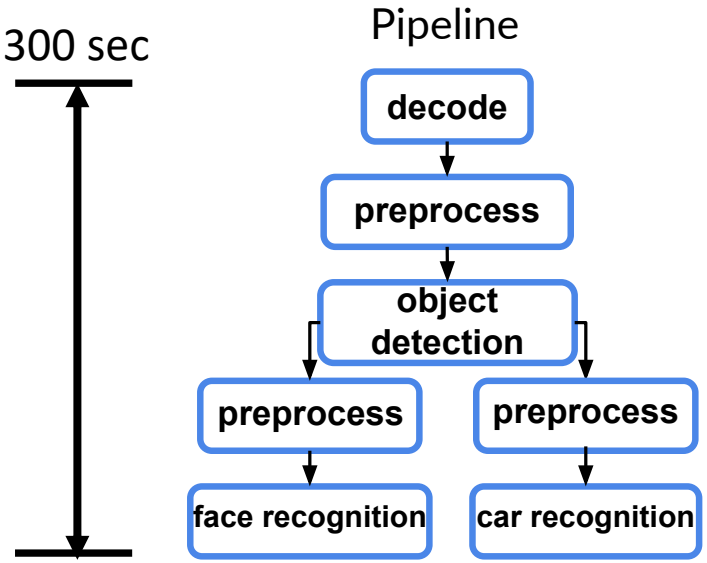
- Real-world applications issue pipelines that include inference tasks
  - Example: Video pipelines are ubiquitous with various cost-perf targets



"Add a vintage filter to the video"



"Identify cars and faces from the traffic feed"

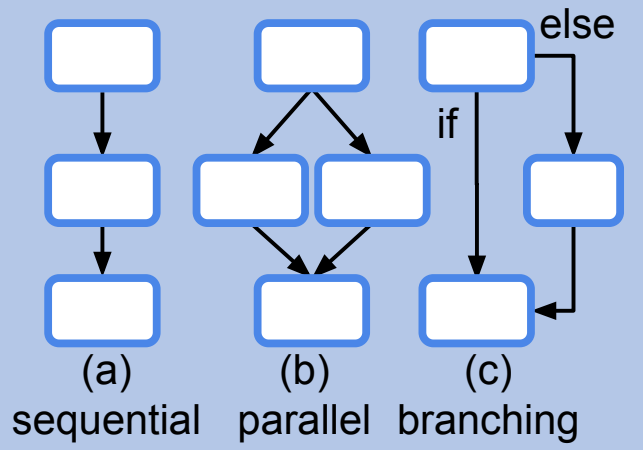


Users configure operation knobs to best meet targets

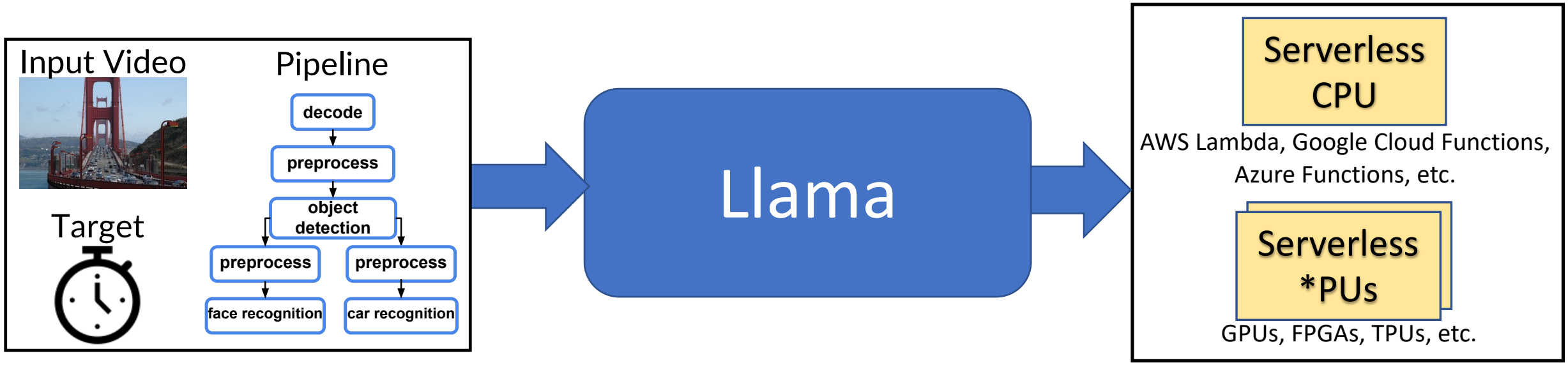
<Hardware resources, batch size, resolution, …>

Challenges:

- Large configuration space
- Input-dependent execution flow
- Exhaustive profiling is expensive



Llama



Fine-grained SLOs (E2E SLO to per-operation SLOs)

Fine-grained resource allocations (per-video, per-frame)

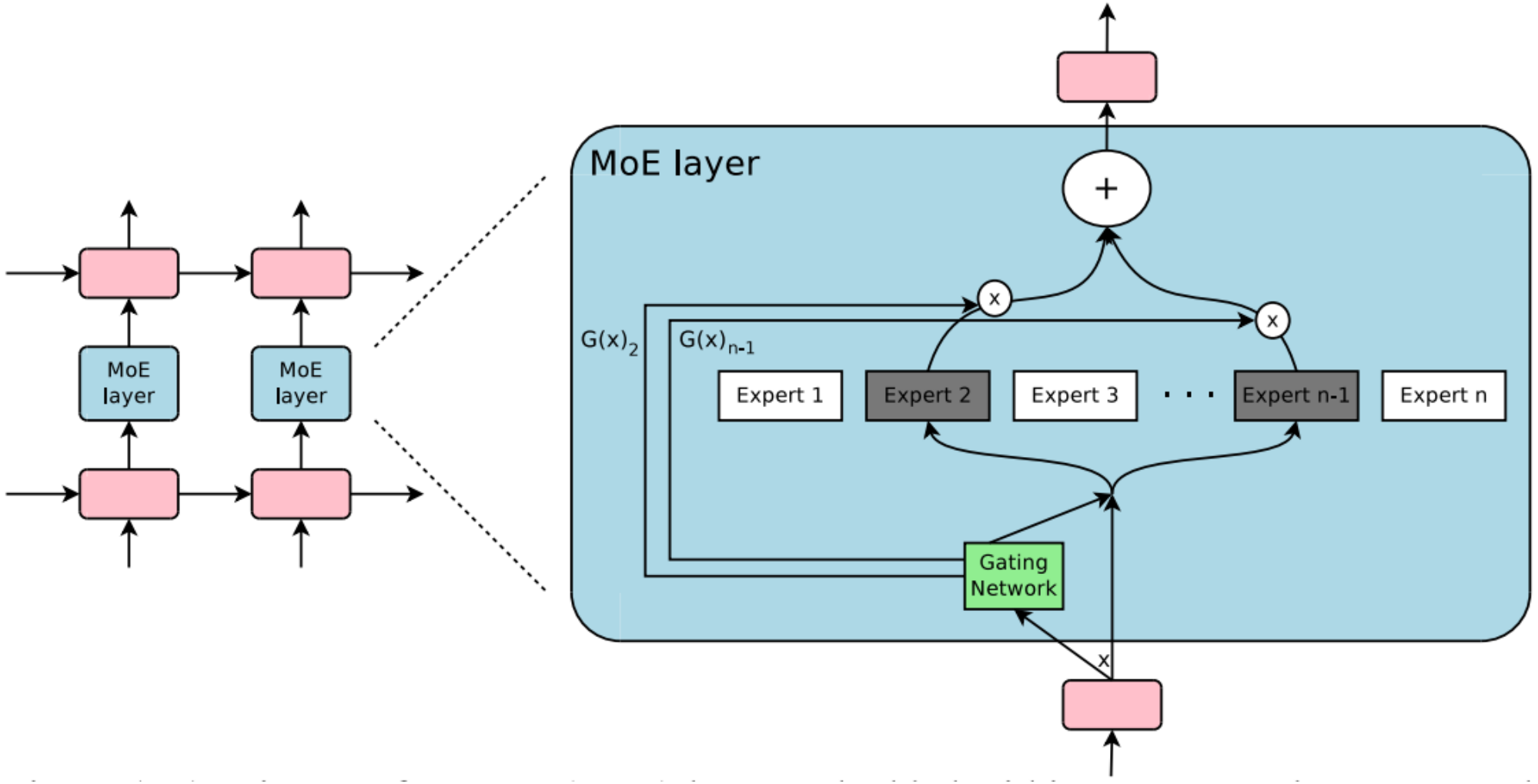
Outline

- What is ML?
- ML Workflow
- Systems for ML: Design Complexity
- Managed Inference Serving by INFaaS (Deep dive)
- Inference Pipelines using Llama
- Sparsely Activated Massive Models

Massive Neural Networks

- **The capacity of a neural network to absorb information is limited by its number of parameters.**
  - More capacity → more accuracy
  - But more the parameters → more computation
- **So sparse models have been proposed**
  - Decreased accuracy
- **Conditionally sparse models: parts of the network are activated per example**
  - Improved model capacity without increasing number of parameters

One such way of building conditionally sparse models: Mixture of Experts (MoEs)



*Outrageously large neural networks: The sparsely-gated Mixture-of-Experts layer, ICLR'17

Research questions

- Resource allocation for inference requests?

- Inference in resource-constrained edge settings

Systems support for ML: It's all about the choices

- ML is a new workload that imposes various new trade-offs and choices for systems
  - Choices that matter
  - Choices that are hard to make
- Who, better than systems folks, should navigate these choices?

Neeraja J. Yadwadkar
neeraja@austin.utexas.edu