Research at the intersection of AI + Systems

Joseph E. Gonzalez

Assistant Professor, UC Berkeley

jegonzal@cs.berkeley.edu

Looking Back on AI Systems

Going back to when I started graduate school …


ML Systems Workshop NIPS
ML Systems Workshop
ICML'16
Search this site
OVERVIEW
ACCEPTED PAPERS
CALL FOR PAPERS
IMPORTANT DATES
INVITED SPEAKERS
ORGANIZERS
SCHEDULE
SITEMAP
Overview
The ML Systems Workshop will be held as part of ICML in NYC on June 24, 2016, 8:30am
Location
Microsoft Technology Center
(11 Times Square, 8th avenue between 41st and 42nd streets)
Room Central Park on 6th Floor
Entrance at the intersection of 8th avenue and 41st street
Phone: 212-245-2100
Map/Directions
2 blocks from the Marriot Marquis Hotel, (between 41st and 42nd streets on 8th Avenue)
Overview
This workshop is a follow up to the ICML audience of the well attended Learning Systems workshop at NIPS 2015 and the Software Engineering for Machine Learning workshop at NIPS 2013.
A new area is emerging at the intersection of machine learning (ML) and systems design. This birth is driven by the explosive growth of diverse applications of ML in production, the continued growth in data volume, and the complexity of large-scale learning systems. The goal of this workshop is to bring together experts working at the crossroads of ML, system design and software engineering to explore the challenges faced when building practical large-scale machine learning systems. In particular, we aim to elicit new connections among these diverse fields, identify tools, best practices and design principles.
The workshop will cover ML and AI platforms and algorithm toolkits (Caffe, Tensor Flow, Torch etc), as well as dive into Machine learning focused developments in distributed learning platforms, programming languages, data structures and general purpose GPU programming.
start
Show pagesource
Trace: » start
Big
Algorithms
Dates
Big
Advances
Speakers
Schedule
Session 1

Machine learning community has had an evolving focus on AI Systems

Fast Algorithms

Distributed Algorithms

Deep Learning Frameworks

2006 → 2017

ML for Systems

Machine Learning Frameworks

→ Integration of Communities

Learning

The focus of AI Systems research has been on model training.

Enabling Machine Learning and Systems Innovations

Stochastic Optimization

Distributed Dataflow Systems

Deep Learning (CNN/RNN)

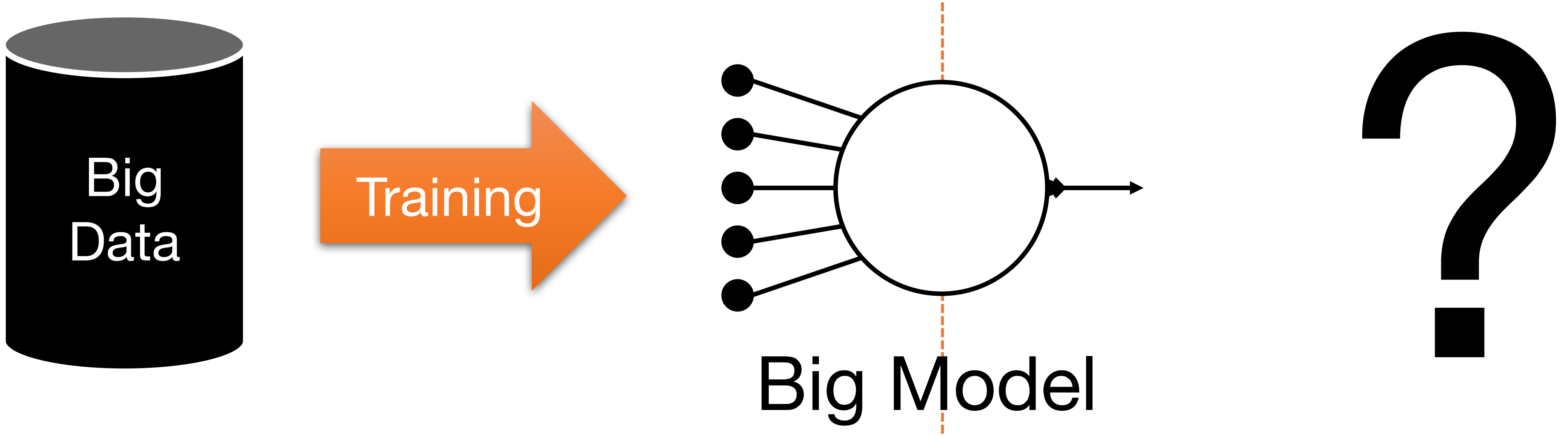
Domain Specific Languages (TensorFlow)

Symbolic Methods

GPU / TPU Acceleration

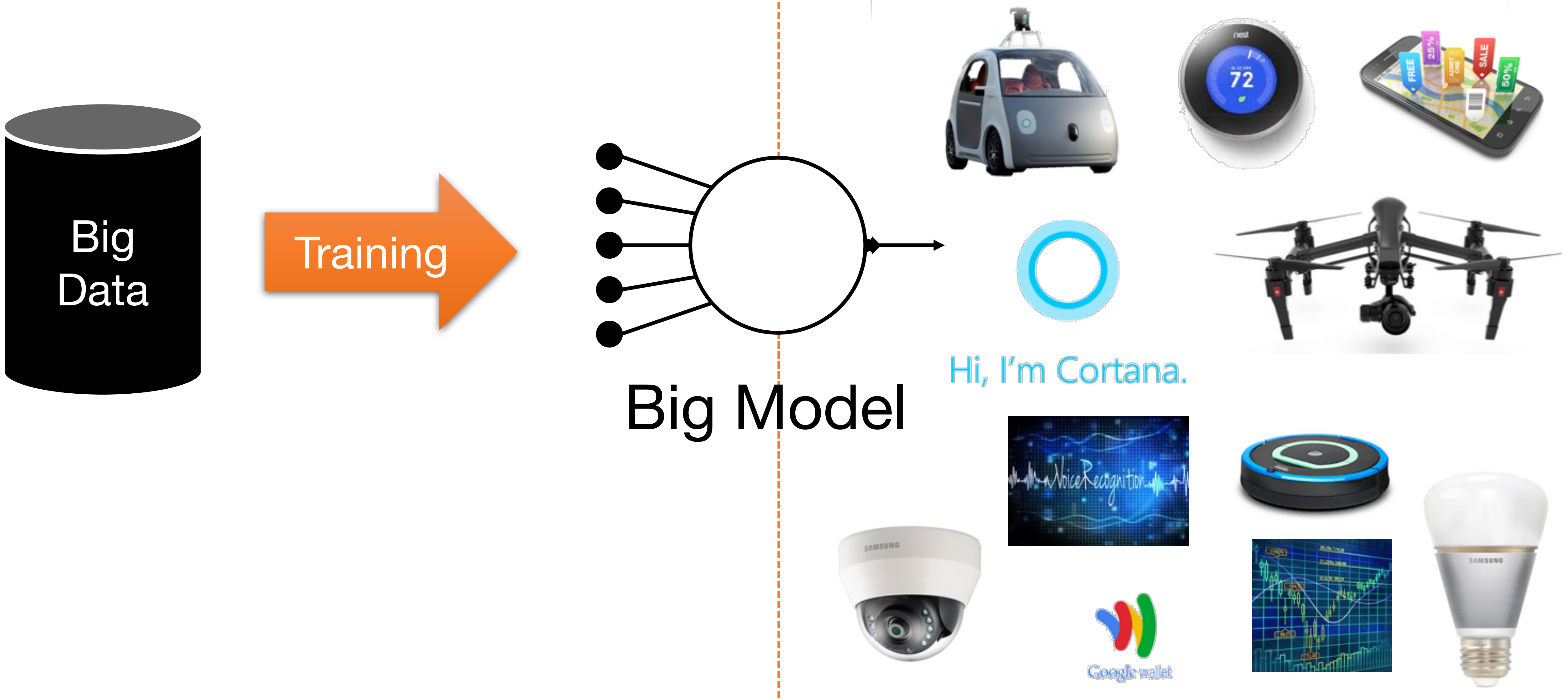
Apache Spark
Caffe
KeystoneML
Splash
BID DATA
MLbase
GraphX
rllab
CoCoA
GraphLab
DMLC
TensorFlow
VW
scikit learn
theano
torch

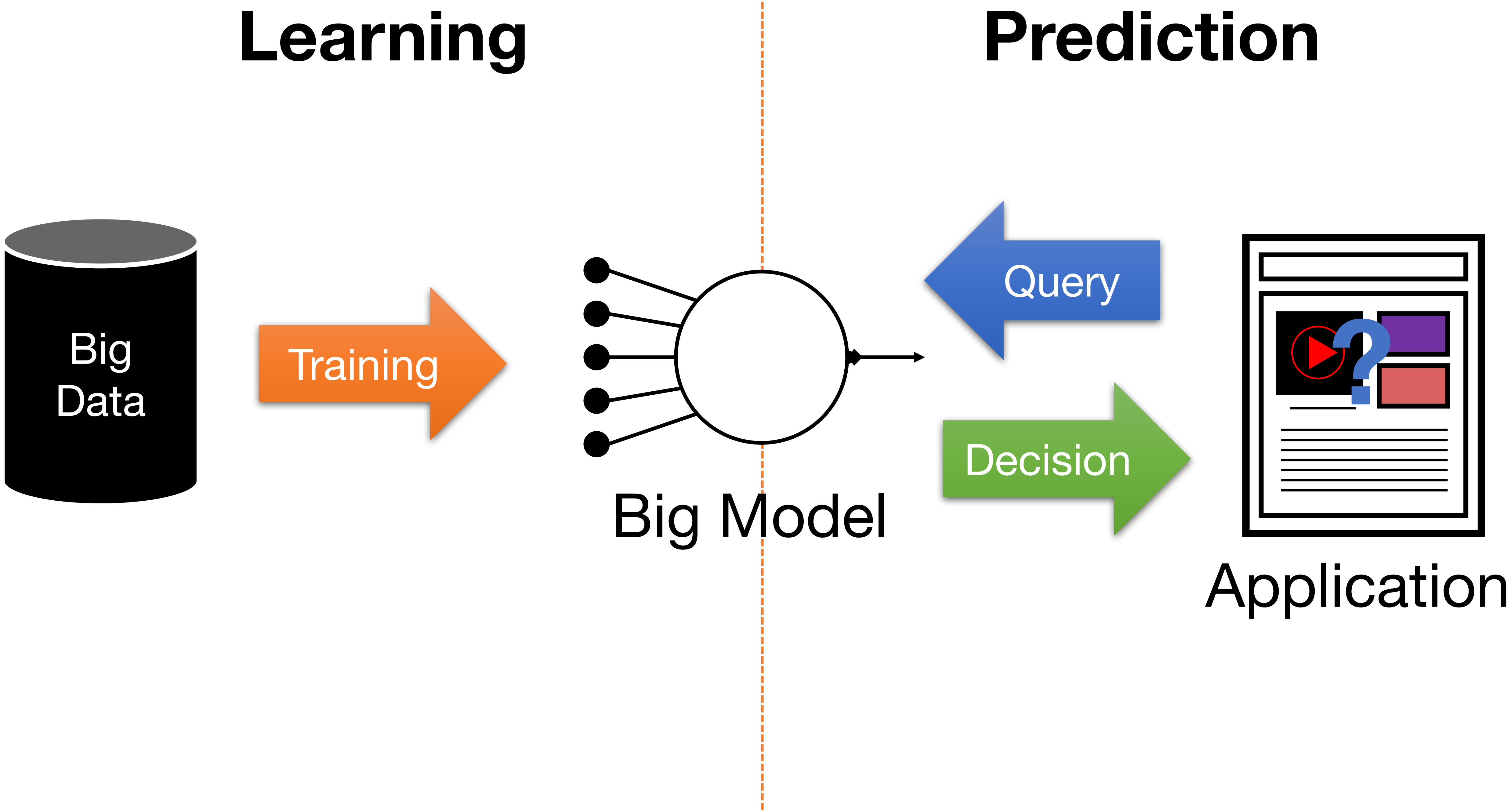
Learning



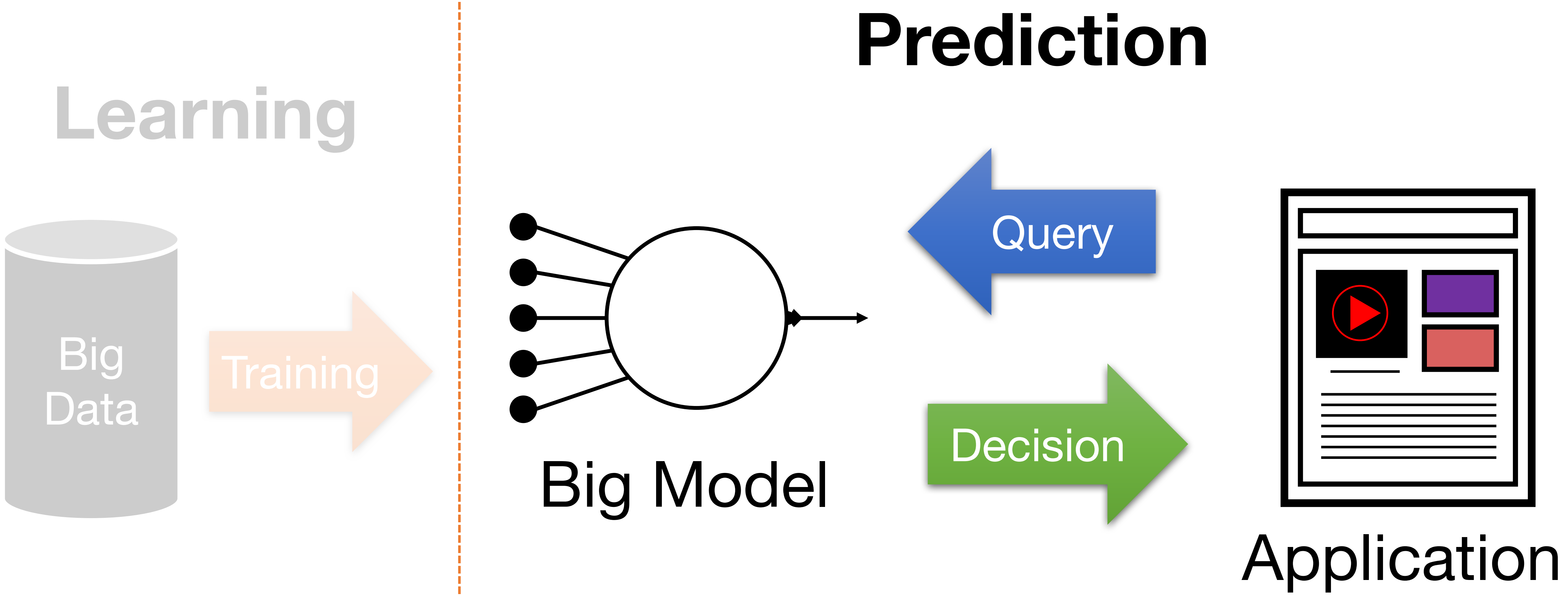
Learning

Drive Actions





Learning
Prediction
Big
Data
Training
Big Model
Query
Decision
Application

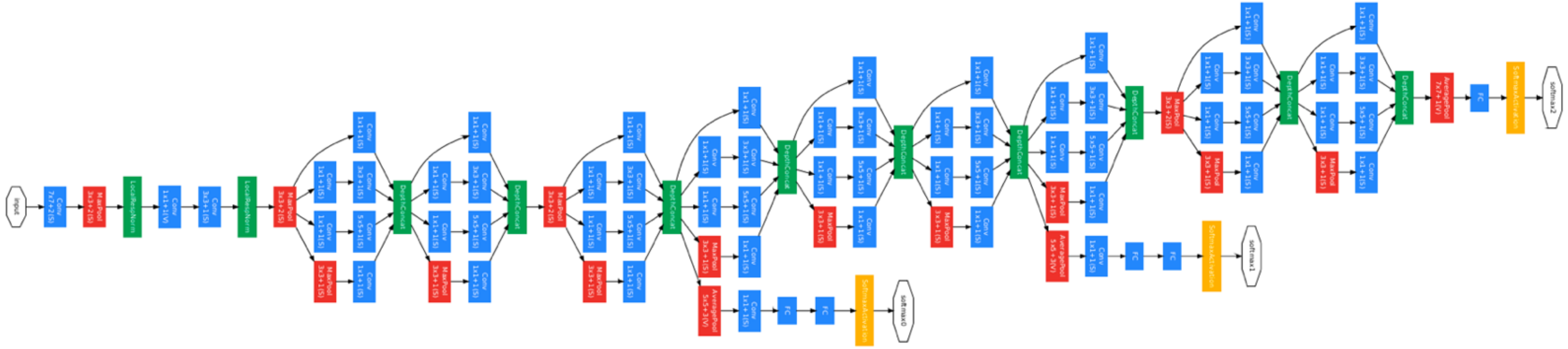


Goal: ~10 ms under heavy load

Complicated by **Deep Learning**

➔ New **ML Algorithms** and **Systems**

Support low-latency, high-throughput serving workloads

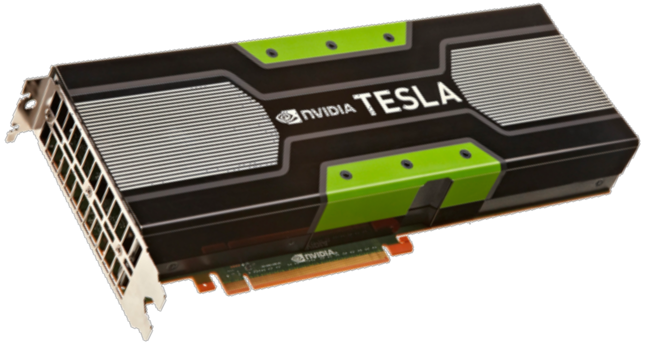


Models getting more complex

- *10s of GFLOPs [1]*
- *Recurrent nets*

Deployed on critical path

- *Maintain latency goals under heavy load*



Using specialized hardware for predictions

[1] Deep Residual Learning for Image Recognition. He et al. CVPR 2015.

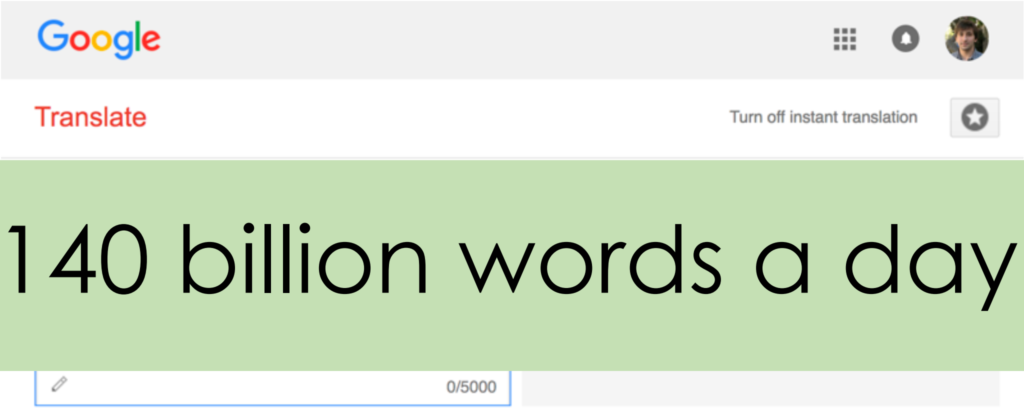
Google Translate

Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi
yonghui,schuster,zhifengc,qvl,mnorouzi@google.com

Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, Jeffrey Dean

Serving



140 billion words a day[1]

82,000 GPUs
running 24/7

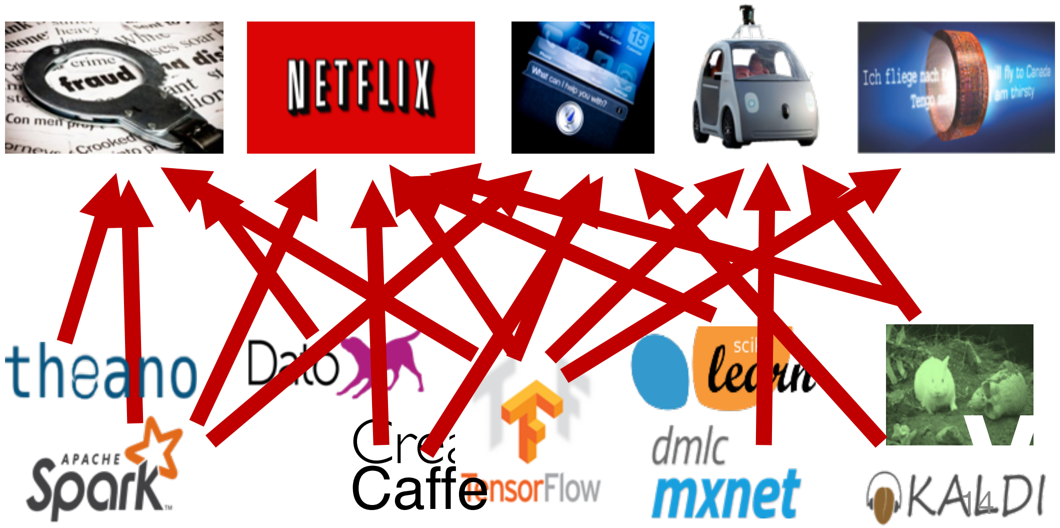
"If each of the ***world's Android phones*** *used the new Google voice search for just* ***three minutes a day****, these engineers realized, the company would* ***need twice as many data centers.****"*
– Wired

Designed New Hardware!
Tensor Processing Unit (TPU)

[1] https://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html

Prediction-Serving Challenges

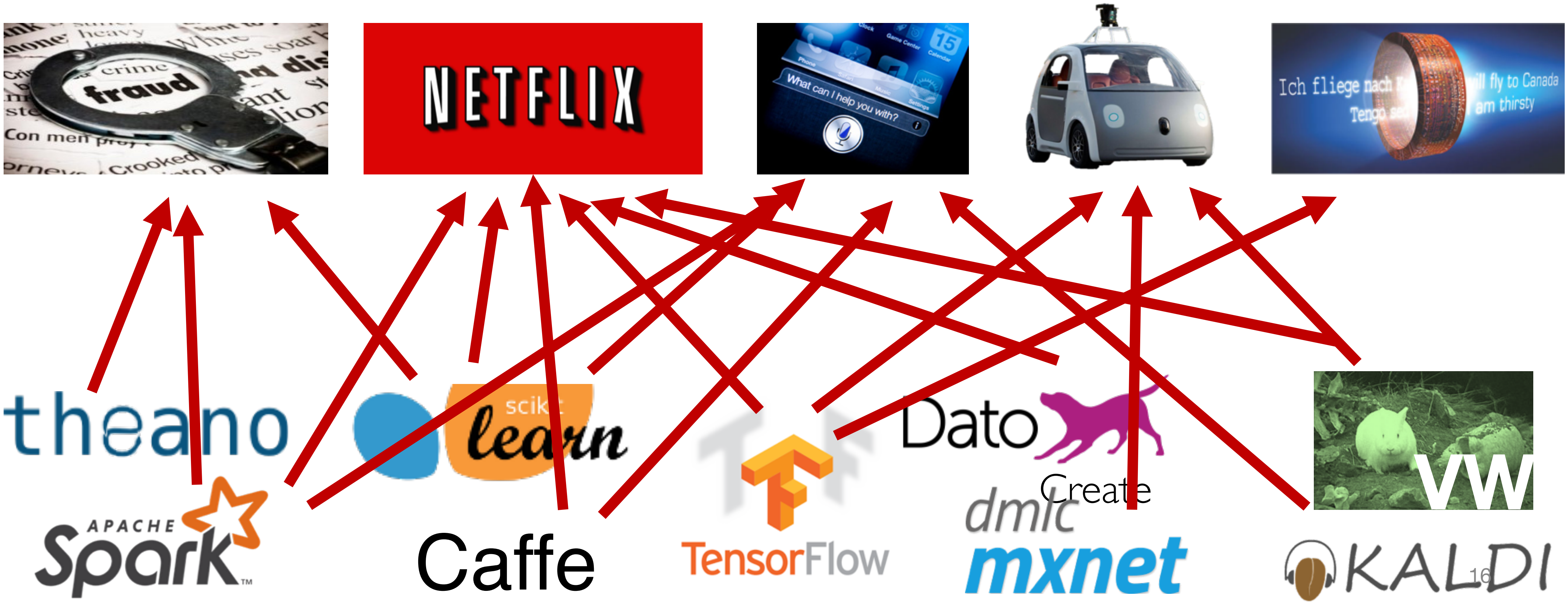
Support low-latency, high-throughput serving workloads



Large and growing ecosystem of ML models and frameworks

Wide range of **application** and **frameworks**

Wide range of **application** and **frameworks**



One-Off Systems for High-Value Tasks

Problems:

Expensive to build and maintain

- ➢ Requires **AI + Systems expertise**

Tightly-coupled model, framework, and application

- ➢ Difficult to **update models** and **add new frameworks**

Prediction Serving is an Open Problem

- Computationally challenging
  - Need low latency & high throughput
- No standard technology or abstractions for serving models

Clipper

Low Latency
Prediction Serving System
[NSDI'17]

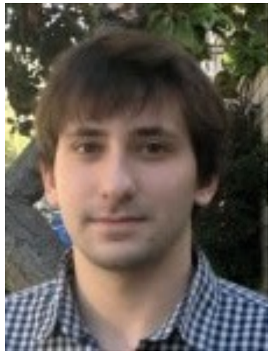
IDK Prediction Cascades

Learning to make fast predictions

[Work in Progress]

Clipper

Low Latency Prediction Serving System


Daniel Crankshaw


Xin Wang


Yika Luo

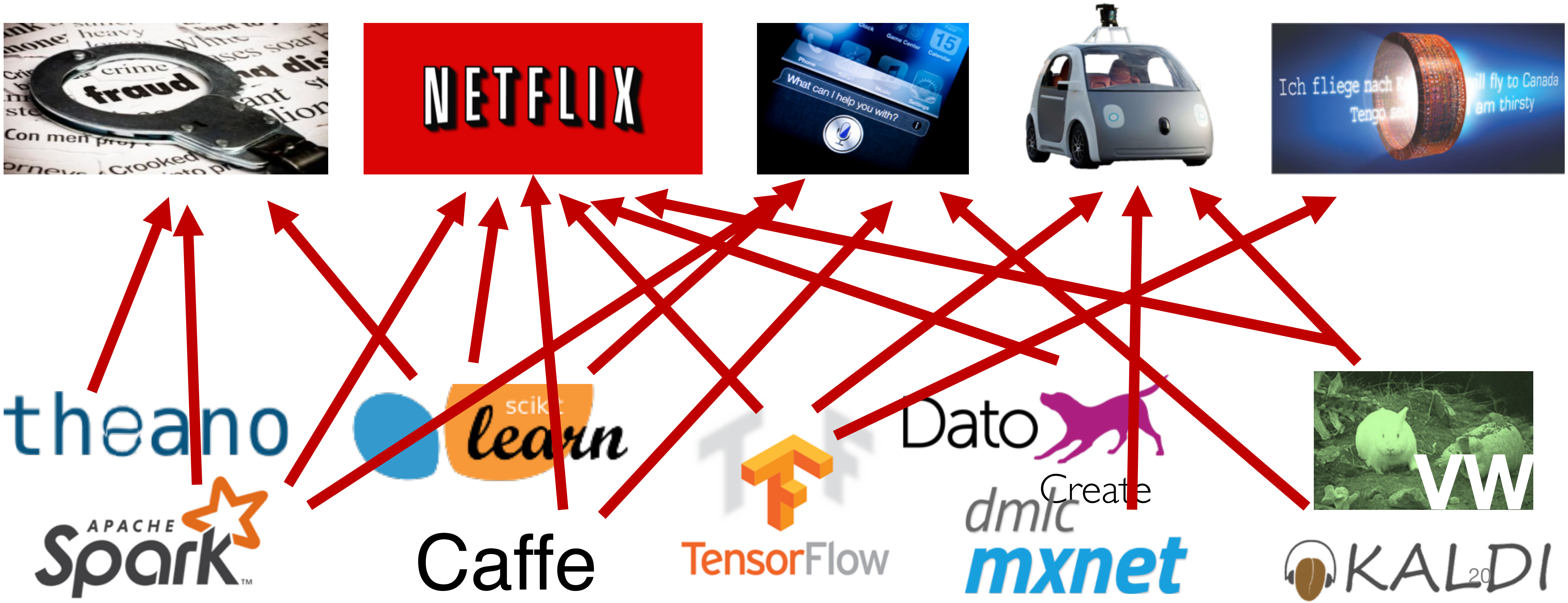

Giulio Zhou


Corey Zumar


Alexey Tumanov


Ion Stoica

Wide range of **application** and **frameworks**



NETFLIX

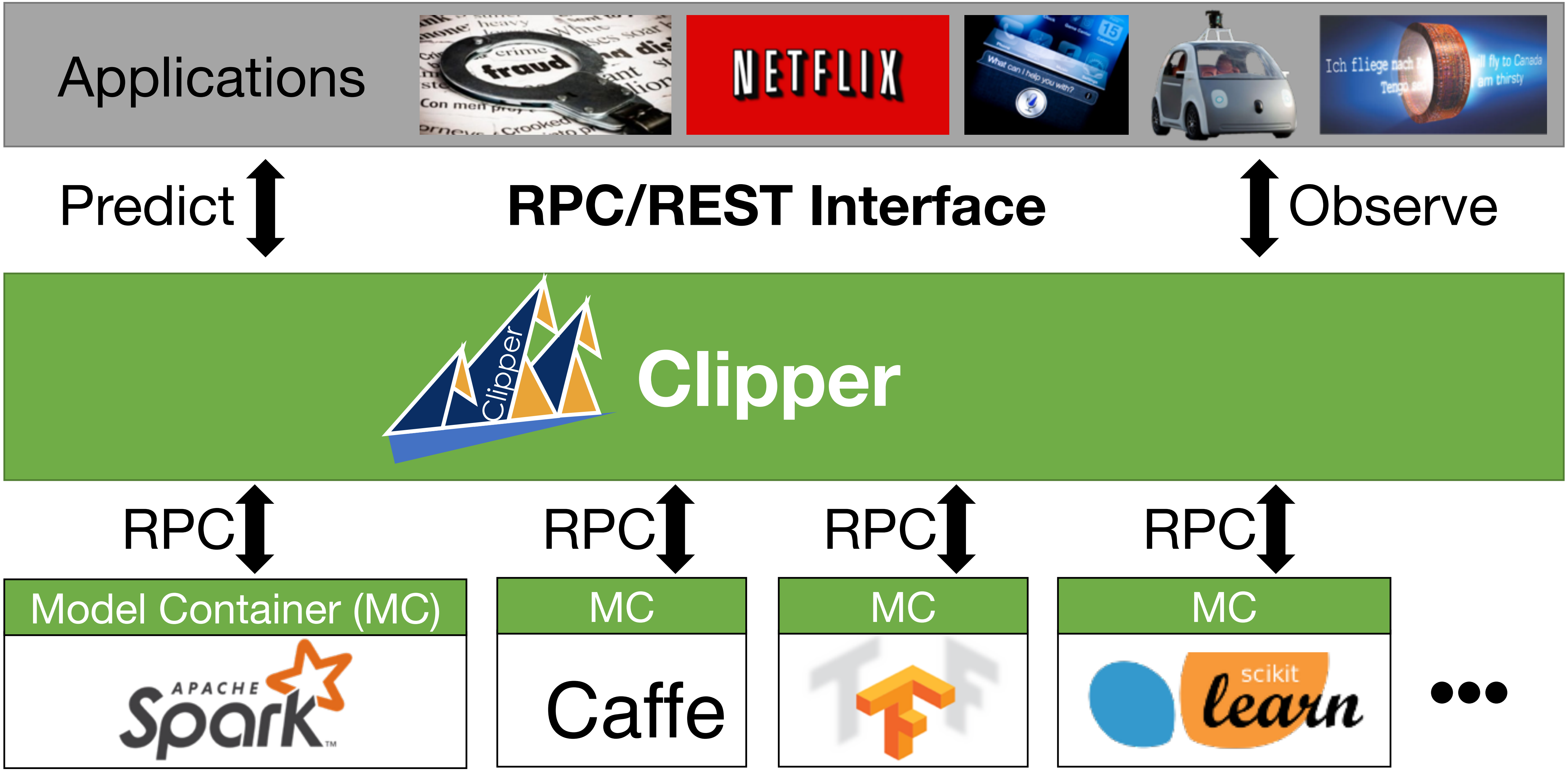
Middle layer for prediction serving.

Common Abstraction

System Optimizations

theano
scikit learn
Dato Create
VW
Apache Spark
Caffe
TensorFlow
dmlc mxnet
KALDI

Clipper Decouples Applications and Models



Clipper Architecture

Applications

Predict ↕ **RPC/REST Interface** ↕ Observe

Clipper

Combine predictions across frameworks — Model Selection Layer

Provide a common interface and system optimizations — Model Abstraction Layer

RPC ↕ RPC ↕ RPC ↕ RPC ↕

Model Container (MC)	MC	MC	MC	...

Clipper Architecture

Applications

Predict | RPC/REST Interface | Observe

Clipper

Optimized Batching

Caching

Common API

Model Isolation

Combine predictions *across frameworks*

Model Selection Layer

Provide a ***common interface*** *and* ***system optimizations***

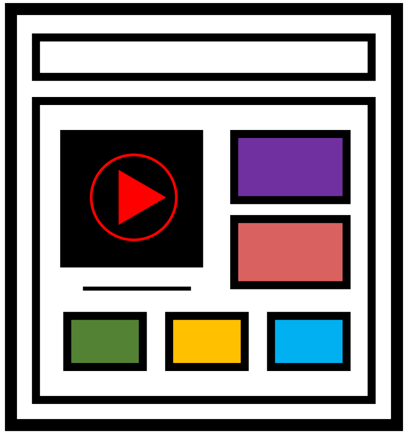
Model Abstraction Layer

RPC | RPC | RPC | RPC

Model Container (MC) | MC | MC | MC

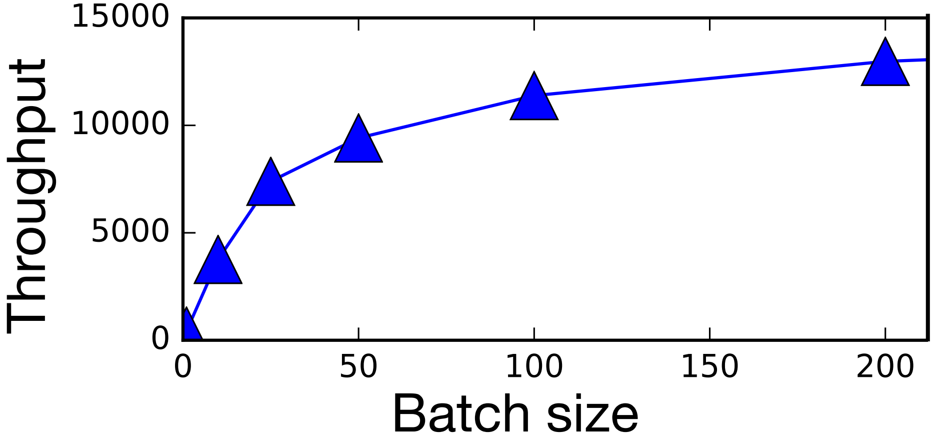
Batching to Improve Throughput

- Why batching helps:



A single page load may generate many queries

Throughput-optimized frameworks

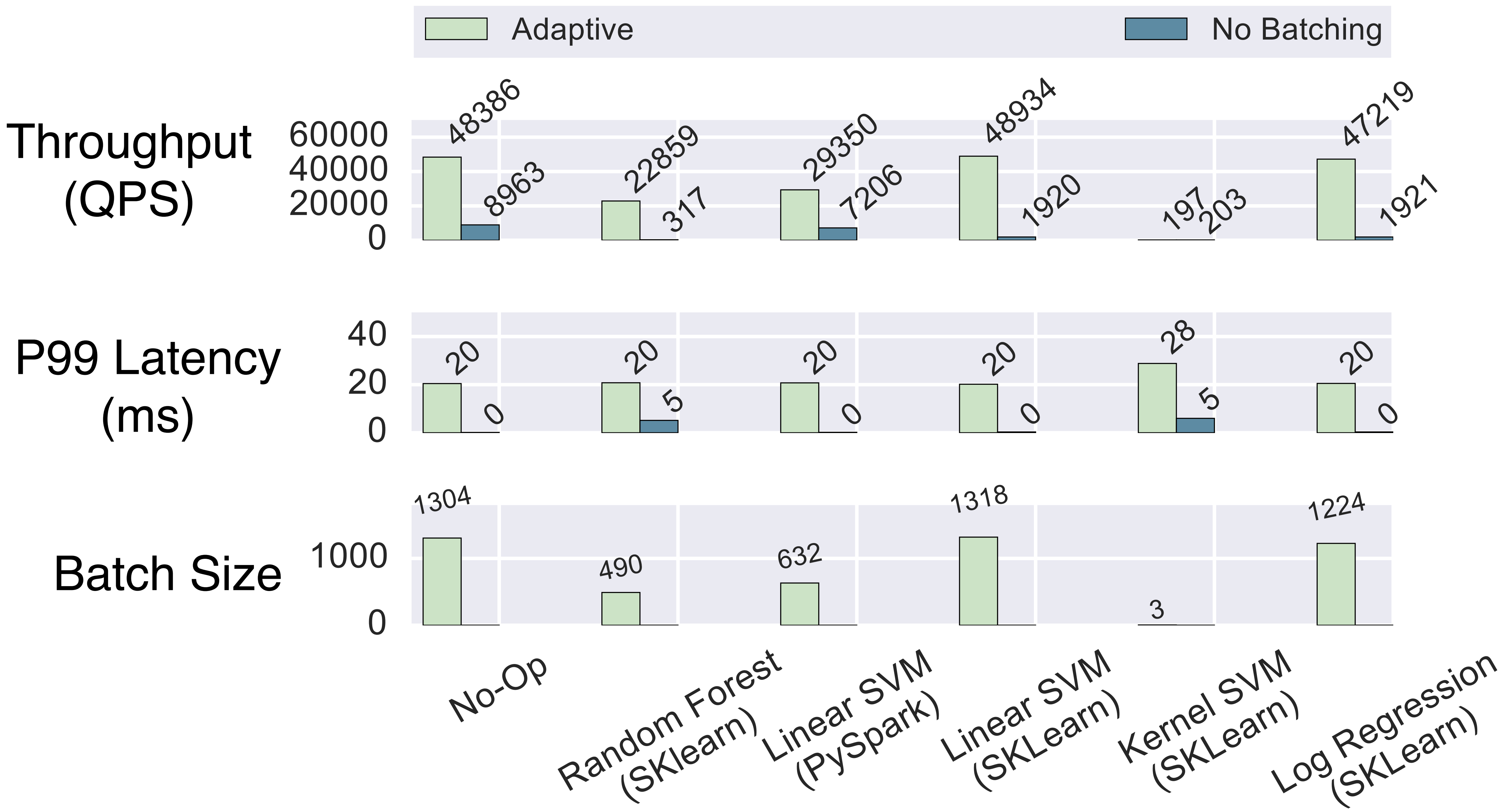


- Optimal batch depends on:
  - hardware configuration
  - model and framework
  - system load

Clipper Solution:

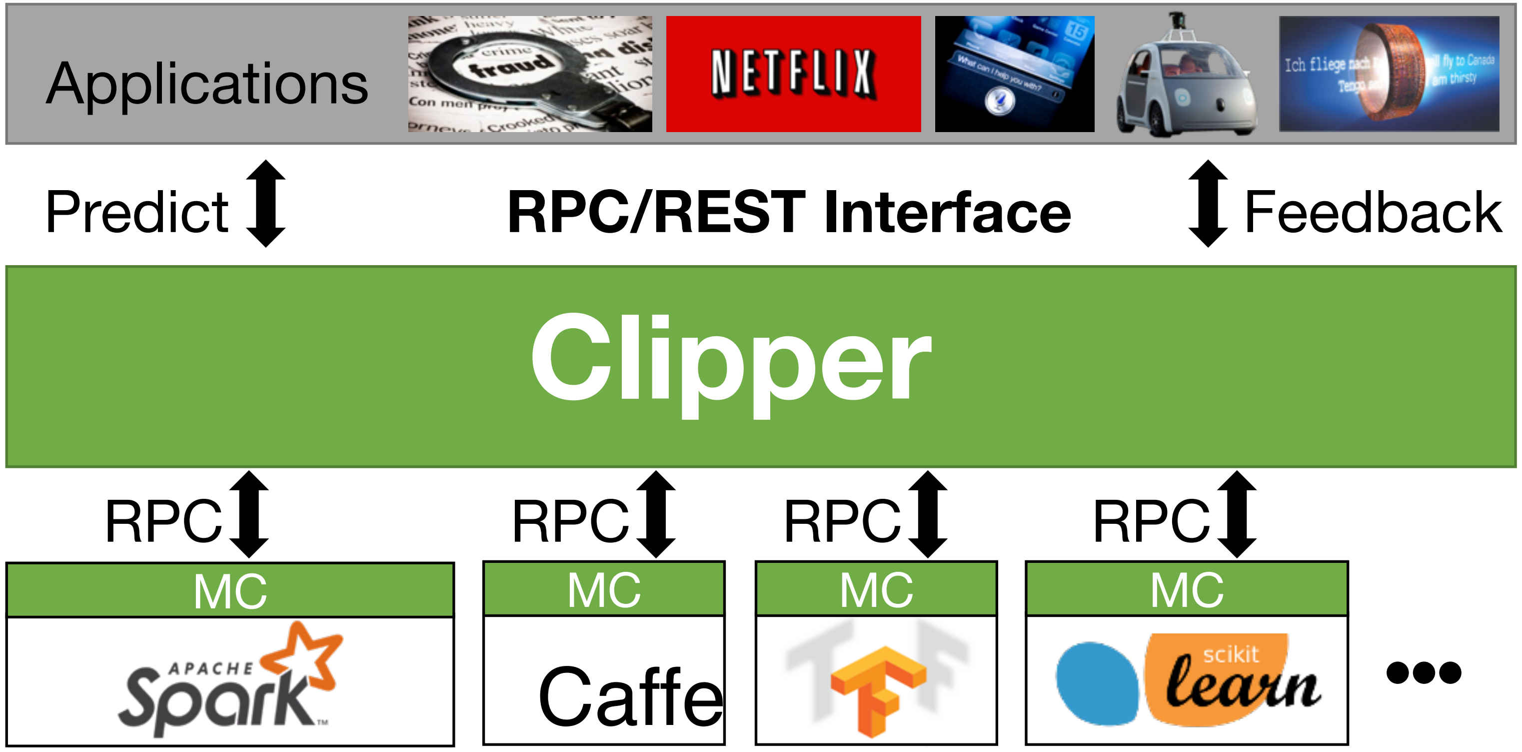
Adaptively tradeoff latency and throughput…

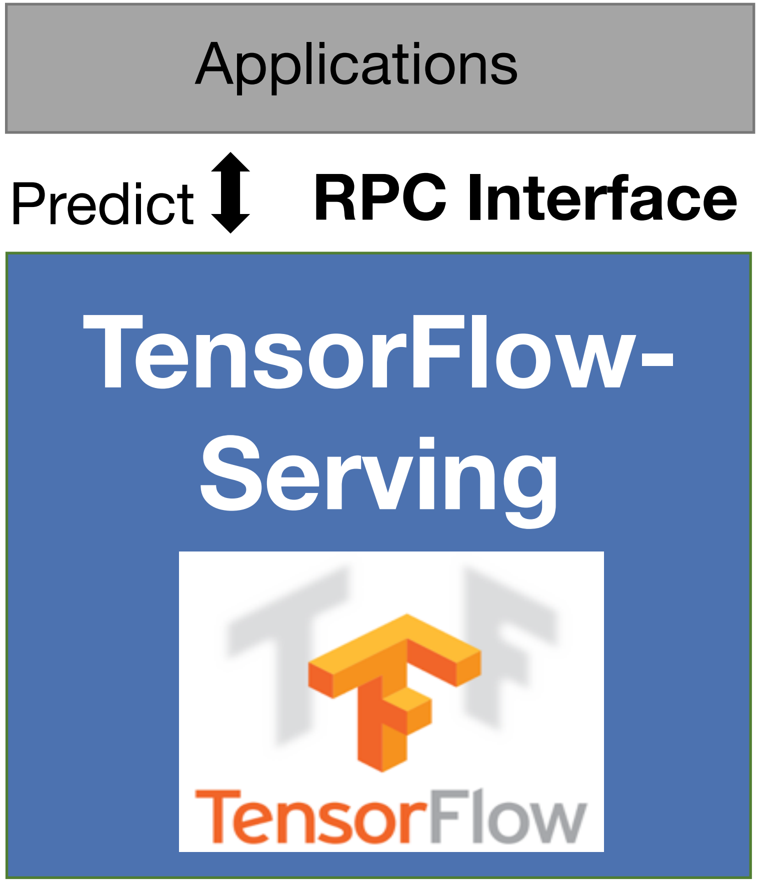
- Inc. batch size *until the latency objective is exceeded* (**Additive Increase**)
- If latency exceeds SLO cut batch size by a fraction (**Multiplicative Decrease**)



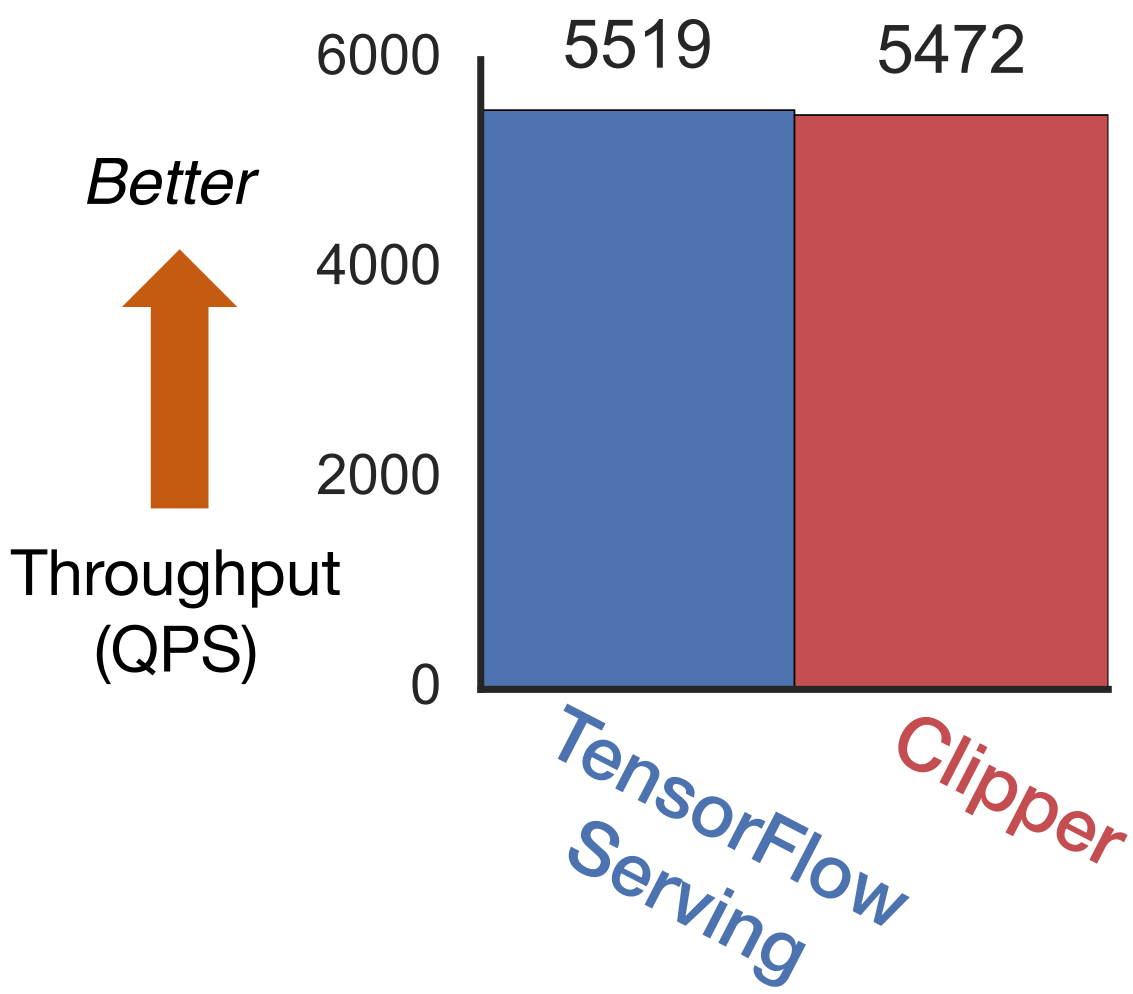
Adaptive
No Batching
Throughput (QPS)
60000
40000
20000
0
48386
8963
22859
317
29350
7206
48934
1920
197
203
47219
1921
P99 Latency (ms)
40
20
0
20
0
20
5
20
0
20
0
28
5
20
0
Batch Size
1000
0
1304
490
632
1318
3
1224
No-Op
Random Forest (SKlearn)
Linear SVM (PySpark)
Linear SVM (SKLearn)
Kernel SVM (SKLearn)
Log Regression (SKLearn)

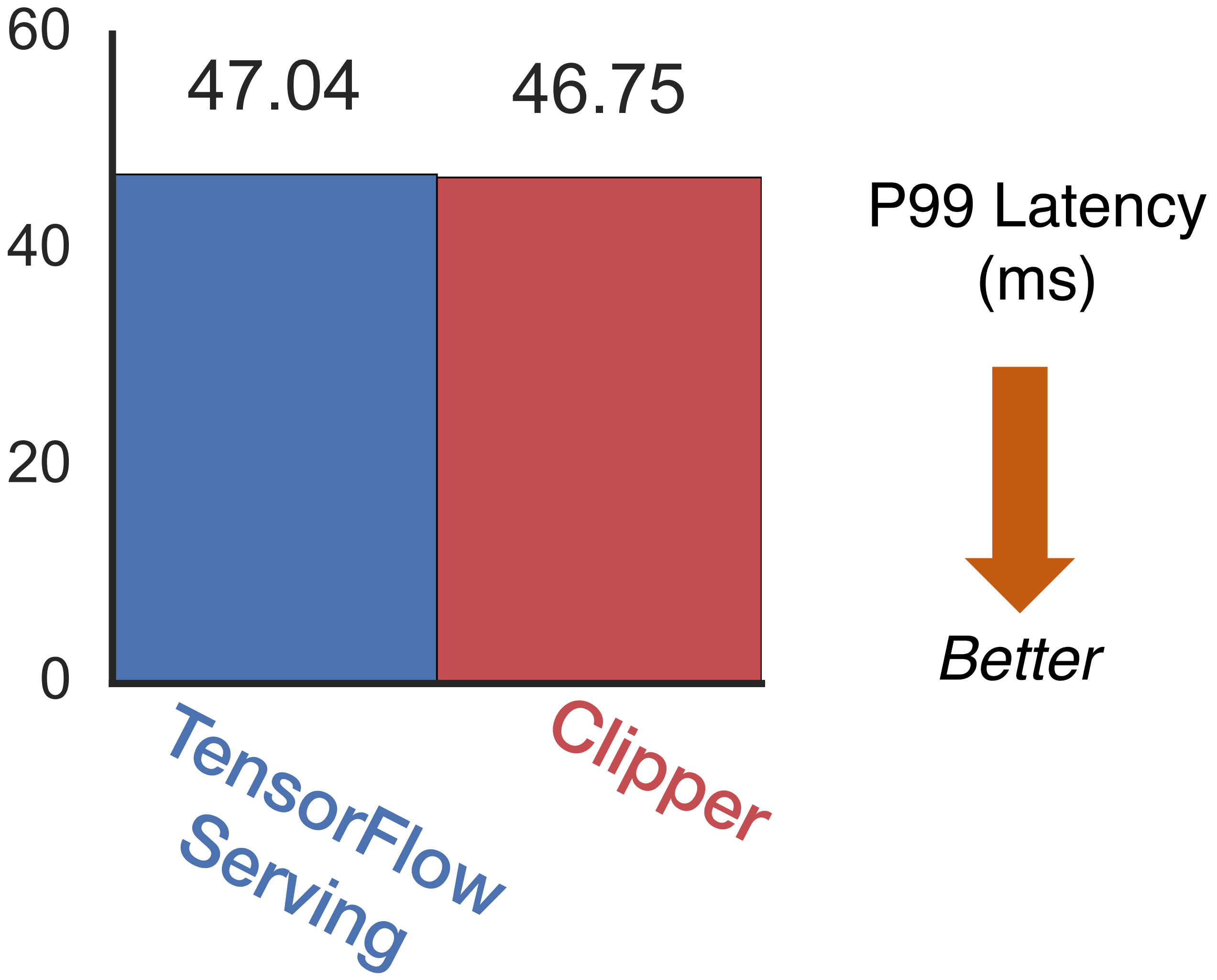
Overhead of decoupled architecture





Overhead of decoupled architecture





Decentralized system matches performance of centralized design.

Clipper Architecture

Applications

Predict

RPC/REST Interface

Observe

Clipper

Combine predictions *across frameworks*

Model Selection Layer

Provide a ***common interface*** *and* ***system optimizations***

Model Abstraction Layer

RPC

RPC

RPC

RPC

Model Container (MC)

MC

MC

MC

Clipper Architecture

Applications

NETFLIX

Predict

RPC/REST Interface

Observe

Clipper

Combine predictions *across frameworks*

Model Selection Layer

Provide a ***common interface*** *and* ***system optimizations***

Model Abstraction Layer

RPC

RPC

RPC

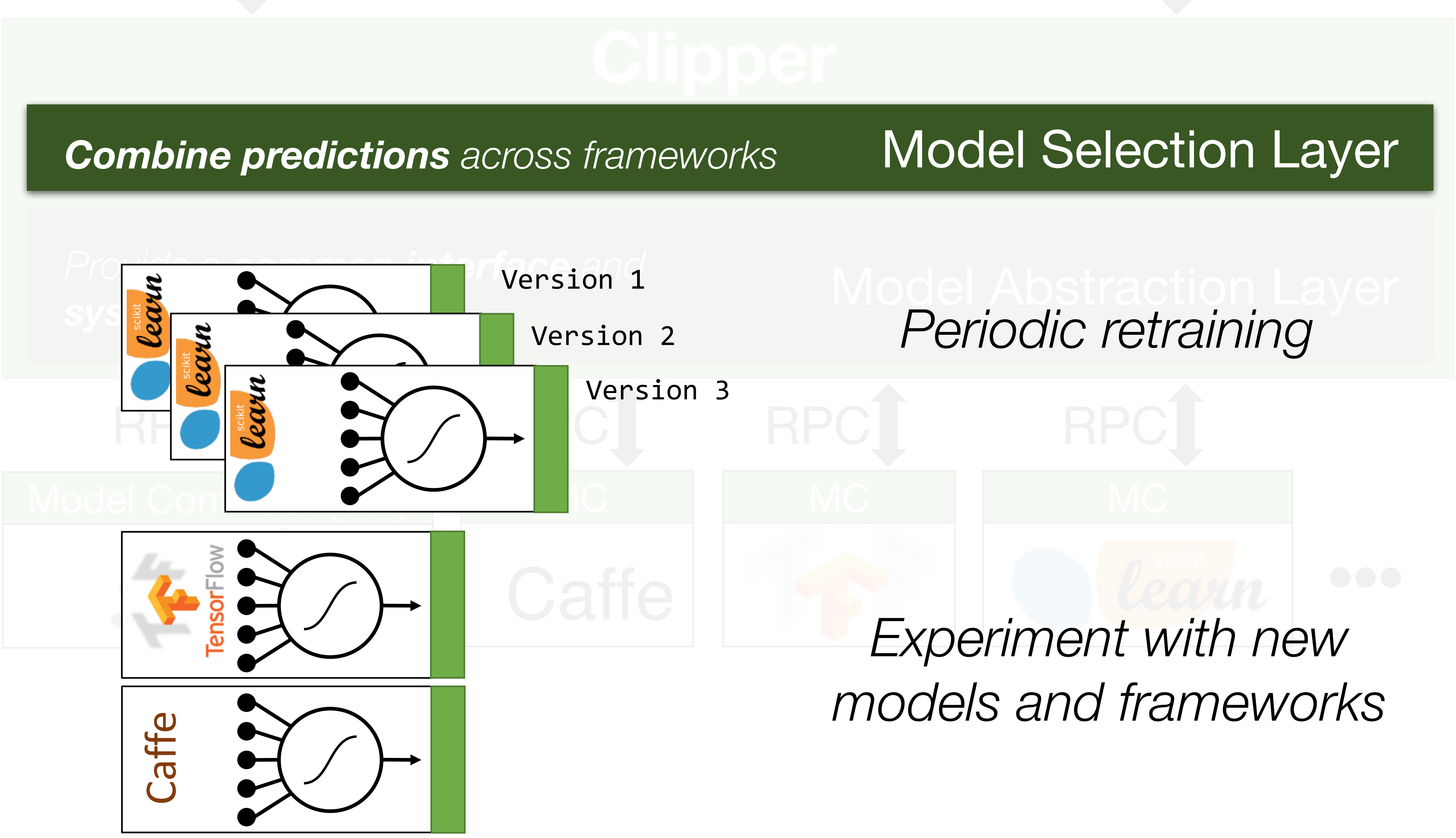
RPC

Model Container (MC)

MC

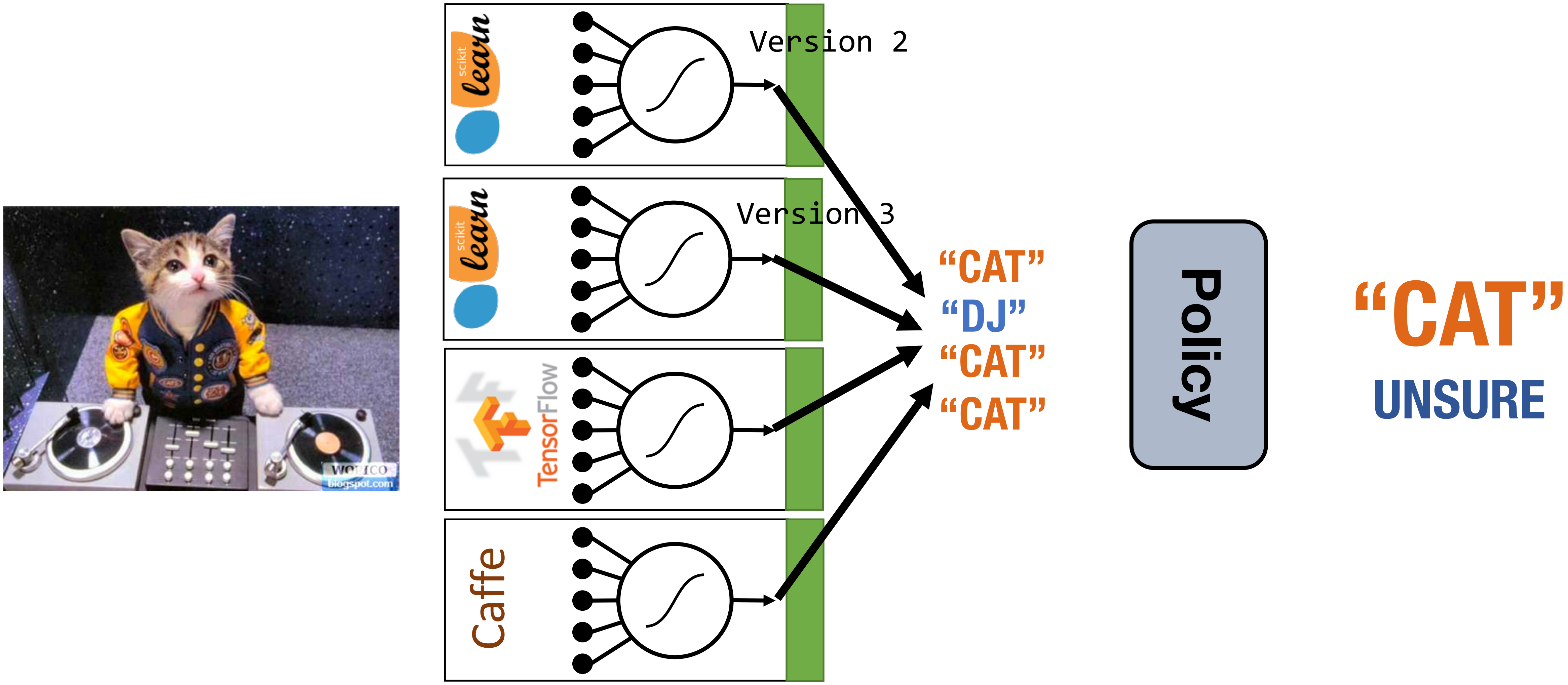
MC

MC



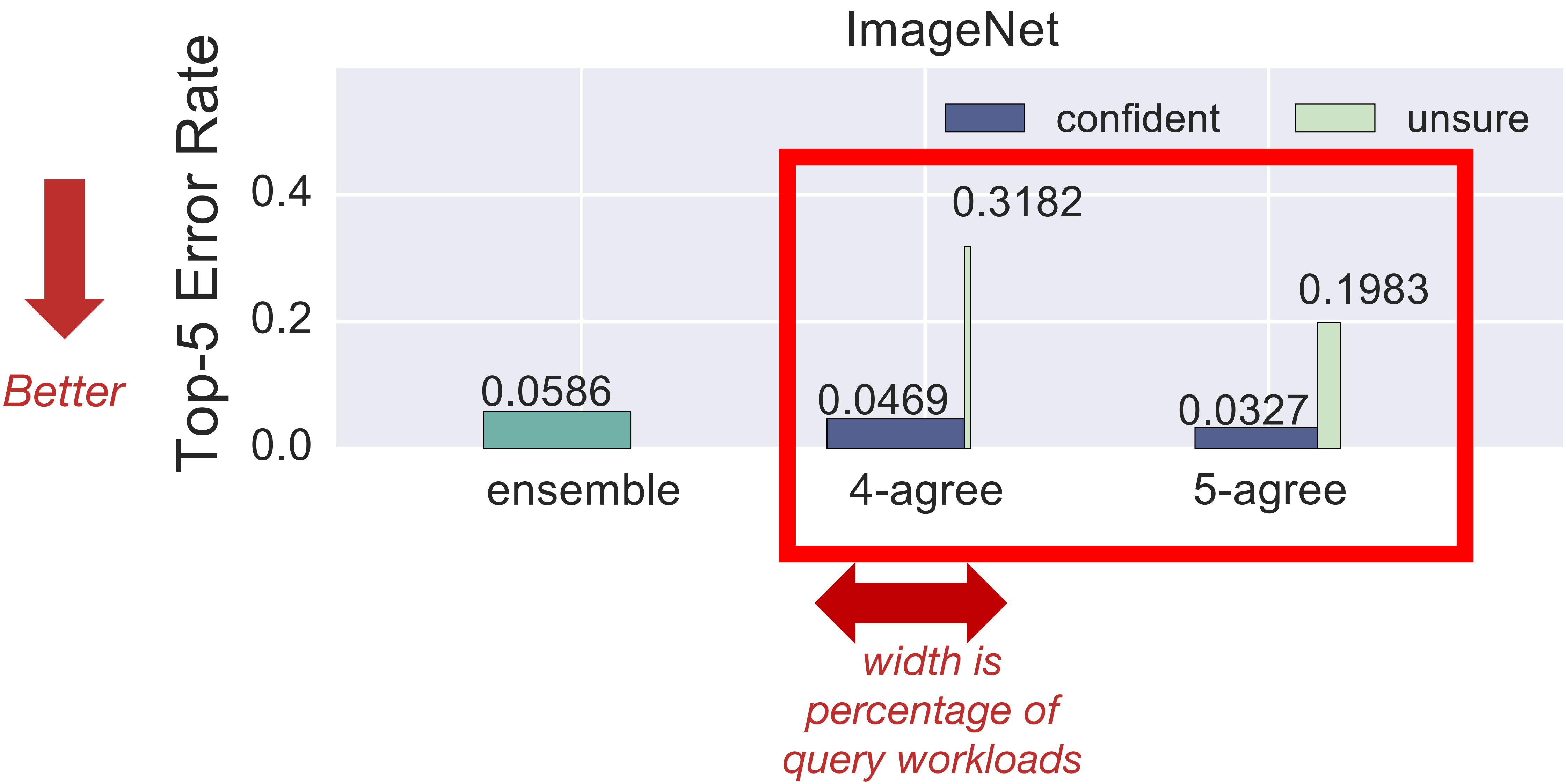
Clipper
Combine predictions across frameworks
Model Selection Layer
Model Abstraction Layer
Version 1
Version 2
Version 3
scikit learn
TensorFlow
Caffe
Periodic retraining
RPC
RPC
MC
MC
Caffe
Experiment with new models and frameworks

Selection Policy can Calibrate Confidence



Selection Policy: Estimate confidence

Selection Policy: Estimate confidence



Open Research Questions

- Efficient execution of **complex model compositions**
  - Optimal batching to achieve end-to-end latency goals
- Automatic **model failure identification and correction**
  - Use anomaly detection techniques to identify model failures
- Prediction **serving on the edge**
  - Allowing models to span cloud and edge infrastructure

http://clipper.ai

Clipper

Low Latency
Prediction Serving System
[NSDI'17]

IDK Prediction Cascades

Learning to make fast predictions.

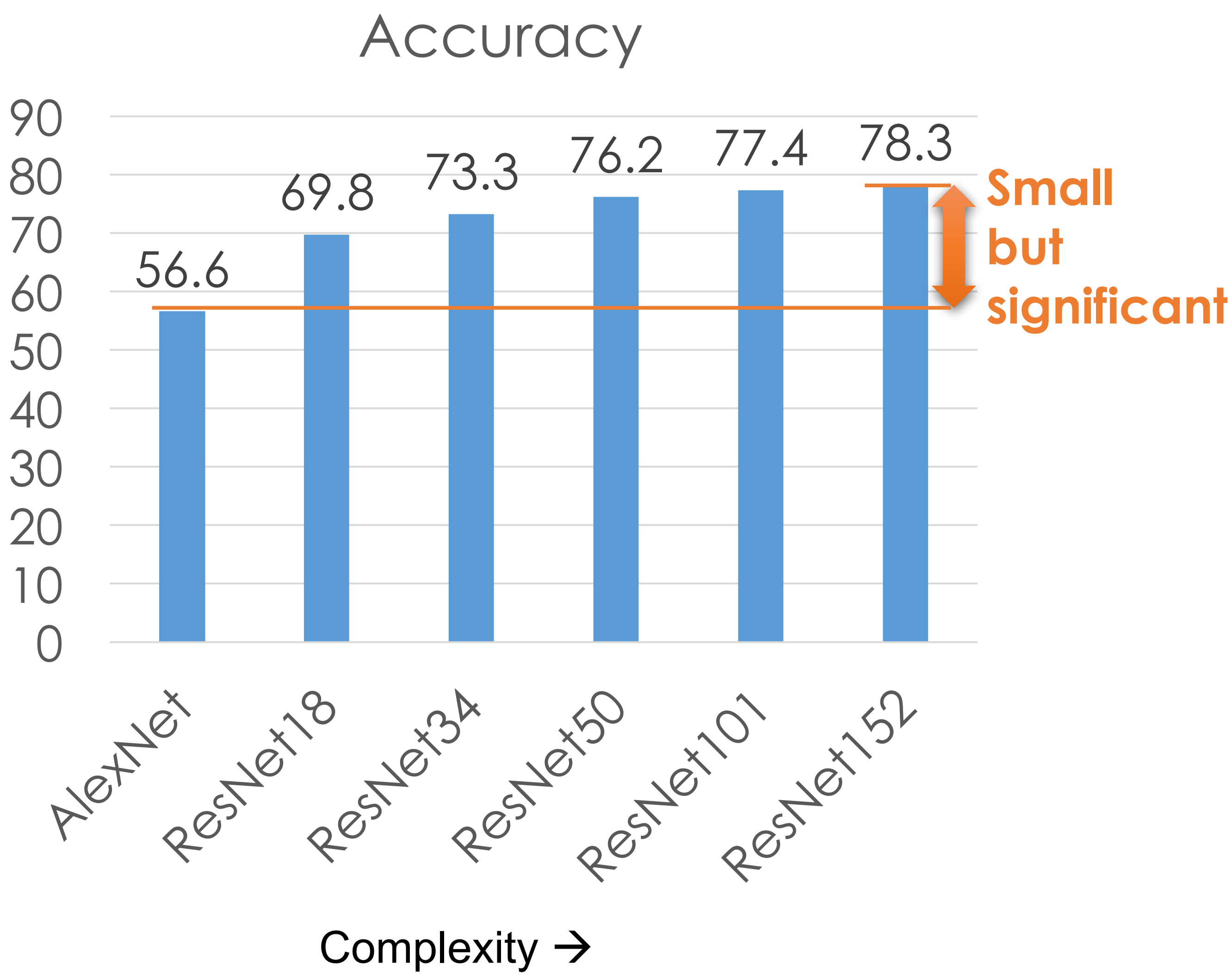
[Work in Progress]

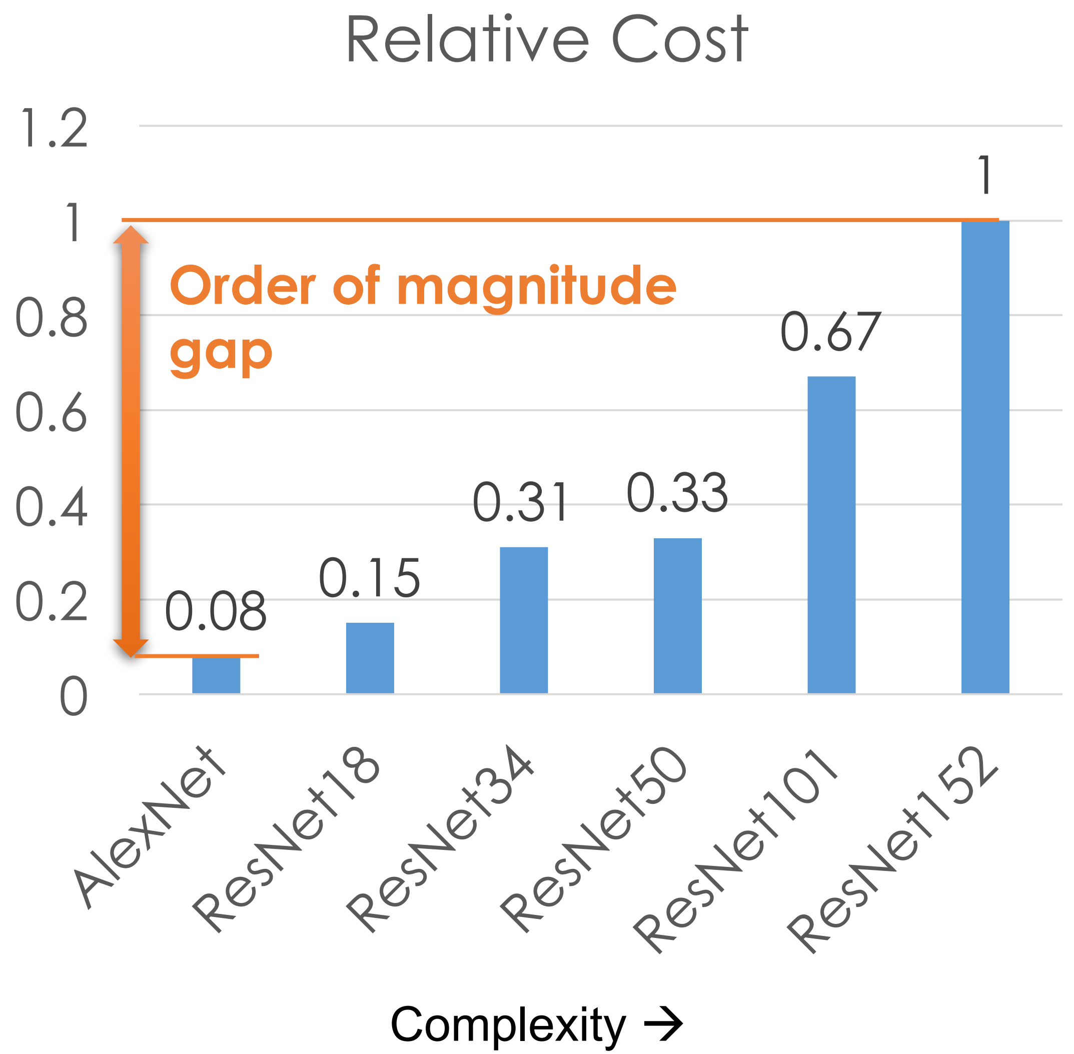
Low Latency
Prediction Serving System
[NSDI'17]

IDK Prediction Cascades

Learning to make fast predictions.

[Work in Progress]





Model **costs are increasing** much **faster than** gains in **accuracy**.

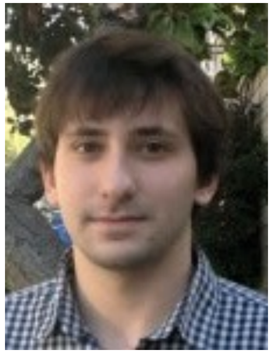
IDK Prediction Cascades

Simple models for simple tasks

https://arxiv.org/abs/1706.00885

Xin Wang

Yika Luo



Daniel Crankshaw

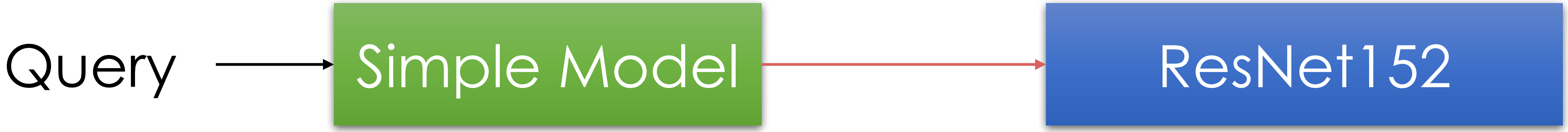
Alexey Tumanov

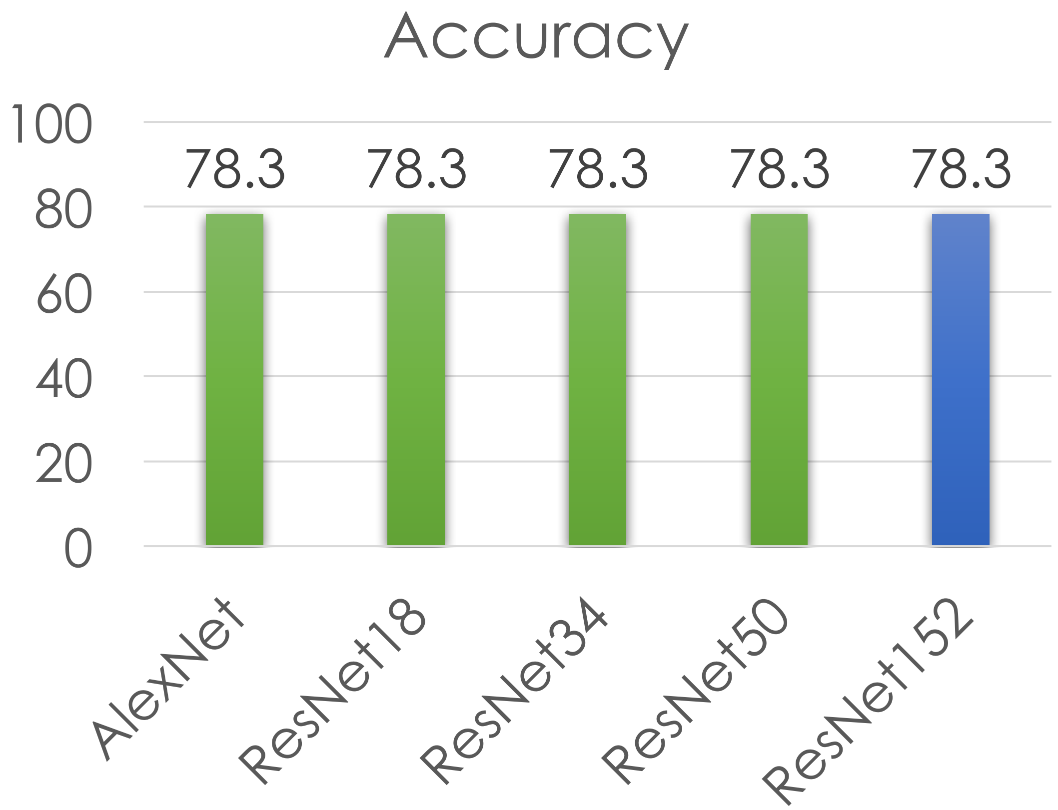
Query → Simple Model → *I don't Know* → Accurate Model

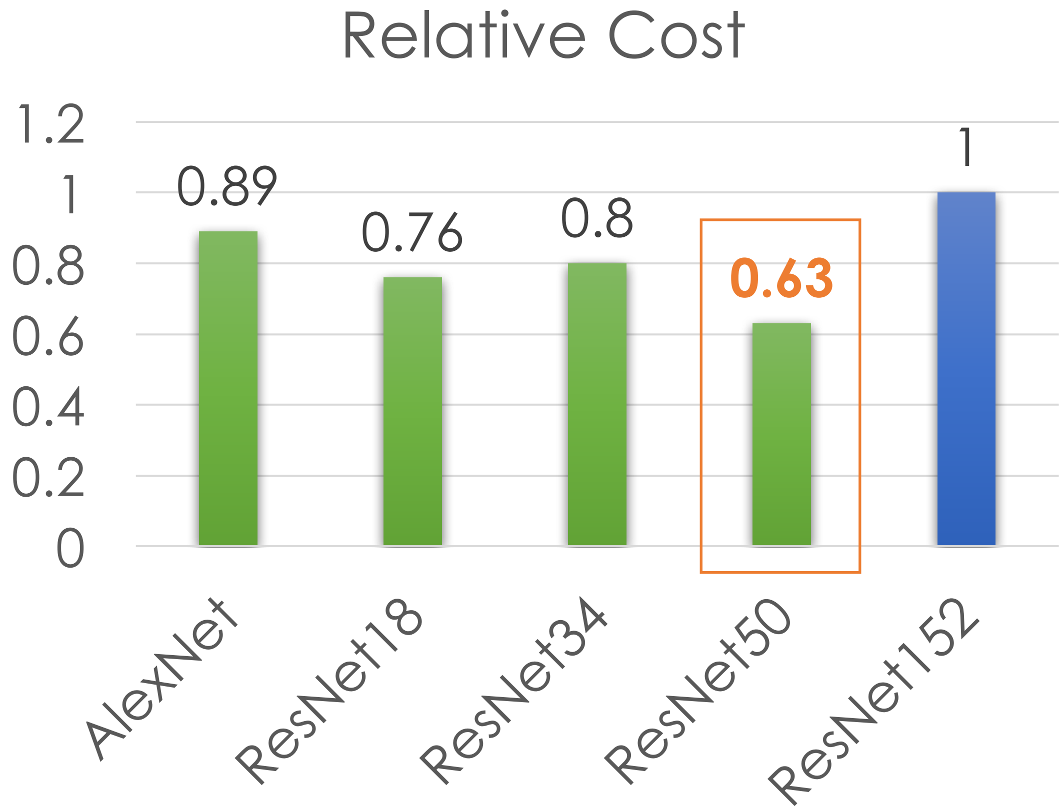
Simple Model → Fast → Prediction

Accurate Model → Slow → Prediction

Combine **fast (inaccurate) models** with **slow (accurate) models** to maximize accuracy while reducing computational costs.

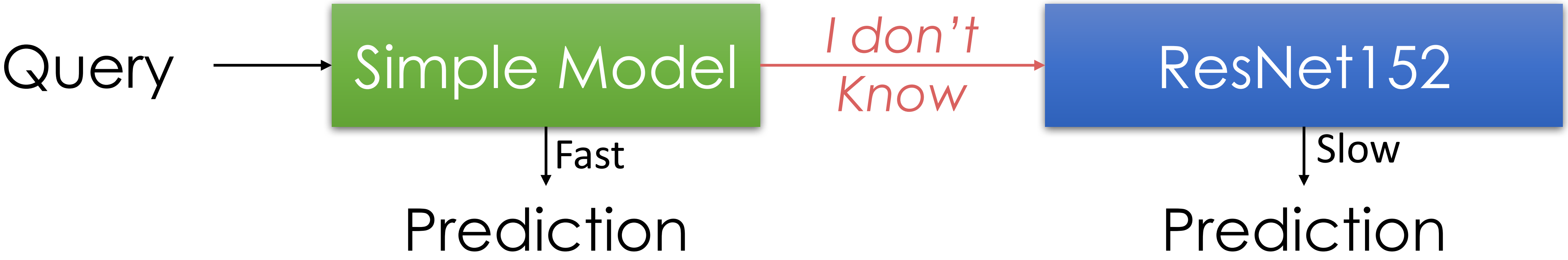




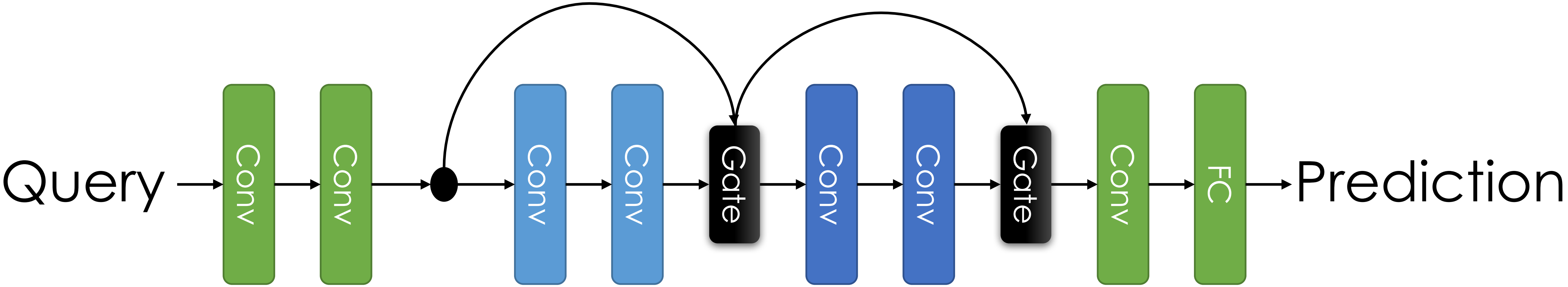


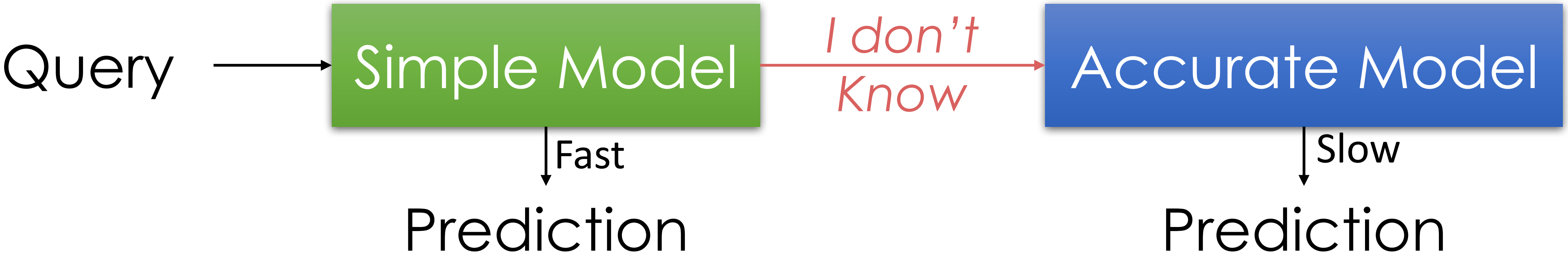
37% reduction in runtime

@ **no loss** in accuracy

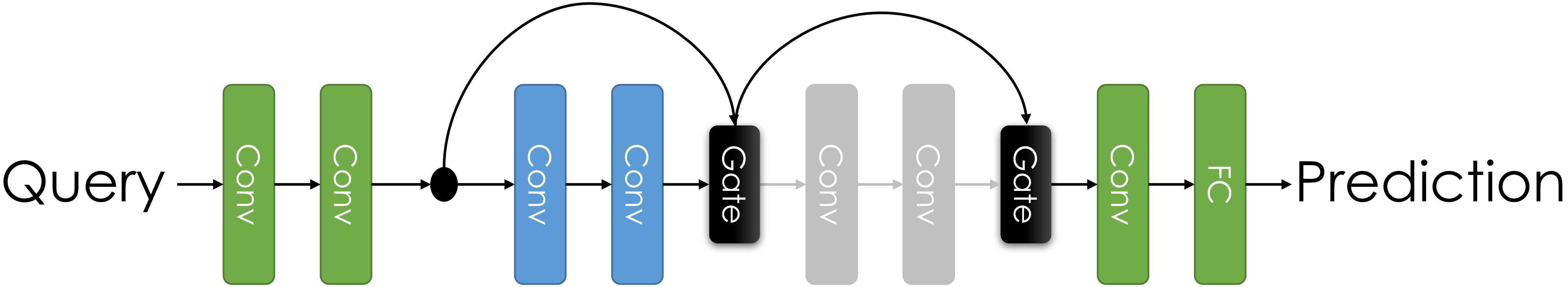


- Cascades within a Model



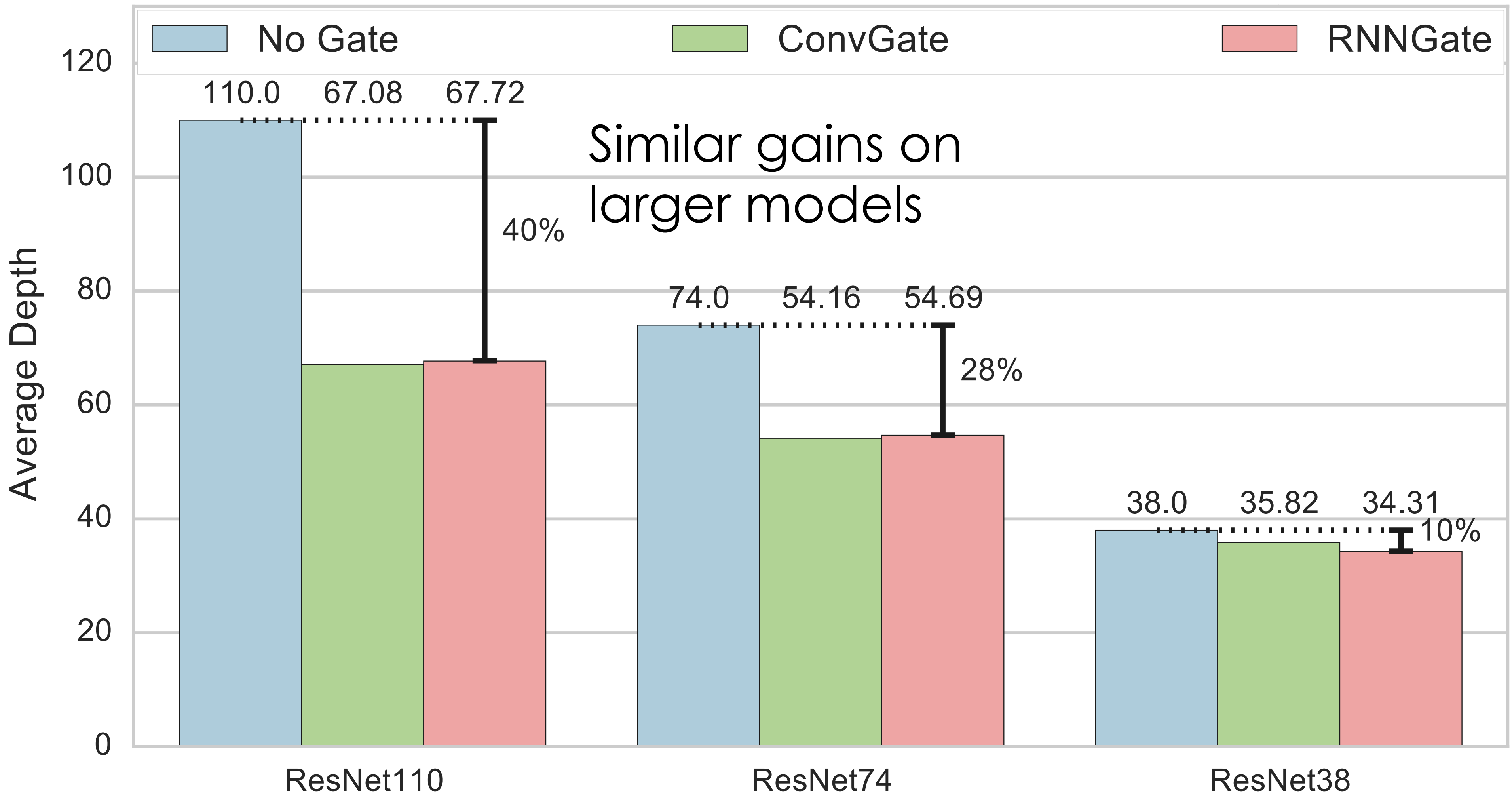


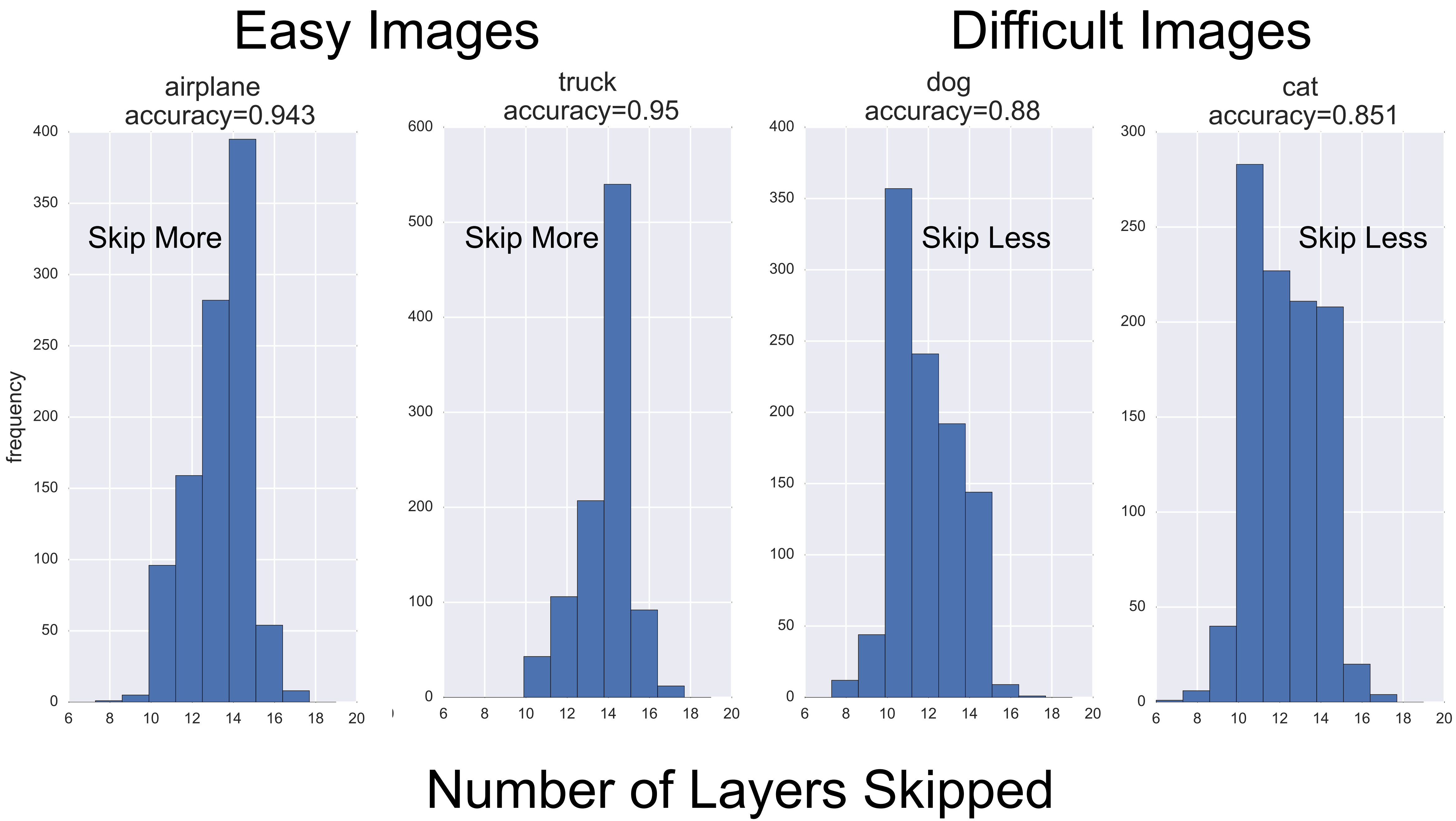
➢ Cascades within a Model



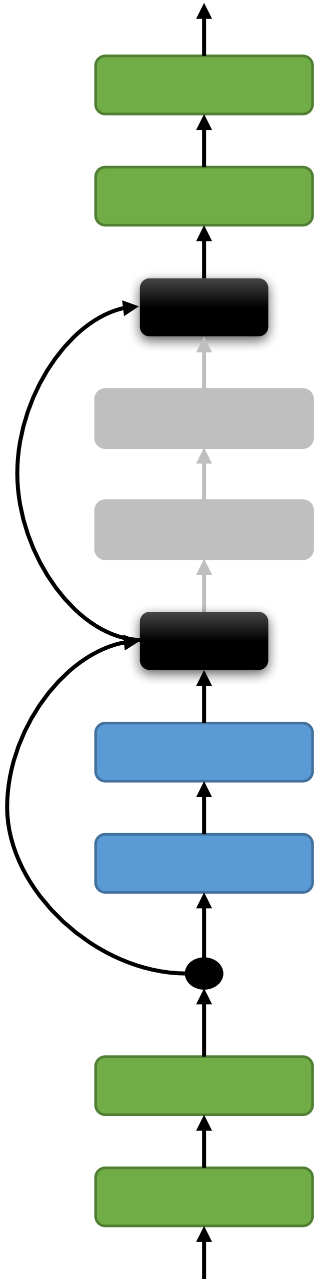
Skip Blocks

Cascading reduces computational cost





Easy Images
Difficult Images
airplane
accuracy=0.943
truck
accuracy=0.95
dog
accuracy=0.88
cat
accuracy=0.851
Skip More
Skip More
Skip Less
Skip Less
frequency
Number of Layers Skipped



Future Directions for Cascades

- Using **reinforcement learning** techniques to reduce gating costs
- **Query triage** during **load spikes** → forcing fractions of the network to go dark
- **Irregular execution** →
  - complicates batching
  - Issues for parallel execution

Clipper

Low Latency
Prediction Serving System
[NSDI'17]

IDK Prediction Cascades

Simple models for simple tasks

[Work in Progress]

Other AI Systems Projects in RISE

Jarvis

Managing the Machine Learning Lifecycle

Ray

Distributed Python for Reinforcement Learning

riselab

UC Berkeley

We are developing new technologies that will enables applications to make low-latency intelligent decision on live data with strong security guarantees.

Joseph E. Gonzalez
jegonzal@cs.berkeley.edu