Clipper

A Low-Latency Online Prediction Serving System

Presenter: Alexey Tumanov

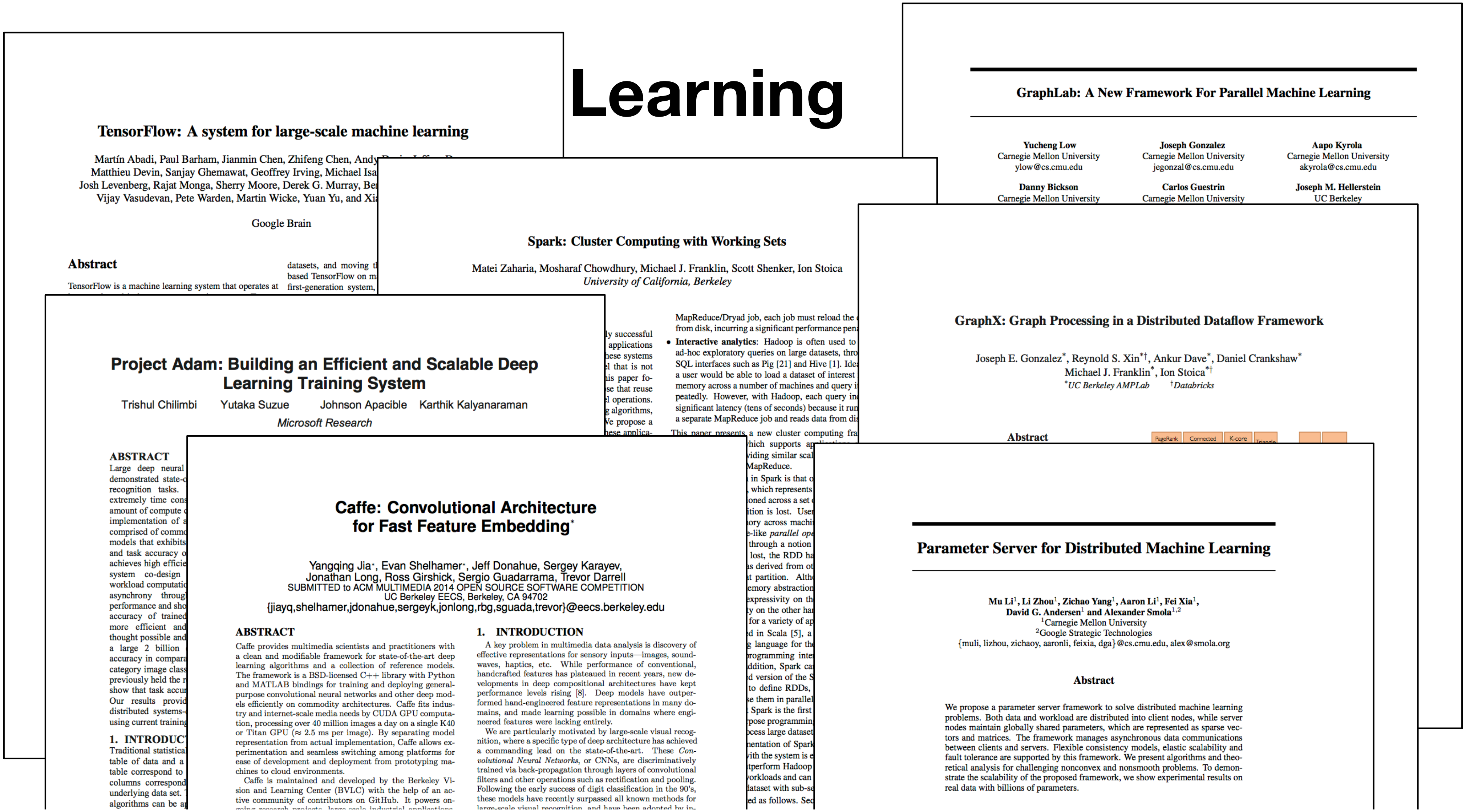
Daniel Crankshaw

Corey Zumar, Guilio Zhou, Xin Wang
Michael Franklin, Joseph Gonzalez, Ion Stoica

BDD/RISE Mini-Retreat 2017

May 2, 2017

Learning



TensorFlow: A system for large-scale machine learning

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy
Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isa
Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Be
Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xi

Google Brain

Spark: Cluster Computing with Working Sets

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley

GraphLab: A New Framework For Parallel Machine Learning

Yucheng Low, Carnegie Mellon University, ylow@cs.cmu.edu
Joseph Gonzalez, Carnegie Mellon University, jegonzal@cs.cmu.edu
Aapo Kyrola, Carnegie Mellon University, akyrola@cs.cmu.edu
Danny Bickson, Carnegie Mellon University
Carlos Guestrin, Carnegie Mellon University
Joseph M. Hellerstein, UC Berkeley

Project Adam: Building an Efficient and Scalable Deep Learning Training System

Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, Karthik Kalyanaraman
Microsoft Research

GraphX: Graph Processing in a Distributed Dataflow Framework

Joseph E. Gonzalez*, Reynold S. Xin*†, Ankur Dave*, Daniel Crankshaw*
Michael J. Franklin*, Ion Stoica*†
*UC Berkeley AMPLab †Databricks

Caffe: Convolutional Architecture for Fast Feature Embedding*

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, Trevor Darrell
SUBMITTED to ACM MULTIMEDIA 2014 OPEN SOURCE SOFTWARE COMPETITION
UC Berkeley EECS, Berkeley, CA 94702
{jiayq,shelhamer,jdonahue,sergeyk,jonlong,rbg,sguada,trevor}@eecs.berkeley.edu

ABSTRACT

Caffe provides multimedia scientists and practitioners with a clean and modifiable framework for state-of-the-art deep learning algorithms and a collection of reference models. The framework is a BSD-licensed C++ library with Python and MATLAB bindings for training and deploying general-purpose convolutional neural networks and other deep models efficiently on commodity architectures. Caffe fits industry and internet-scale media needs by CUDA GPU computation, processing over 40 million images a day on a single K40 or Titan GPU (≈ 2.5 ms per image). By separating model representation from actual implementation, Caffe allows experimentation and seamless switching among platforms for ease of development and deployment from prototyping machines to cloud environments.

Parameter Server for Distributed Machine Learning

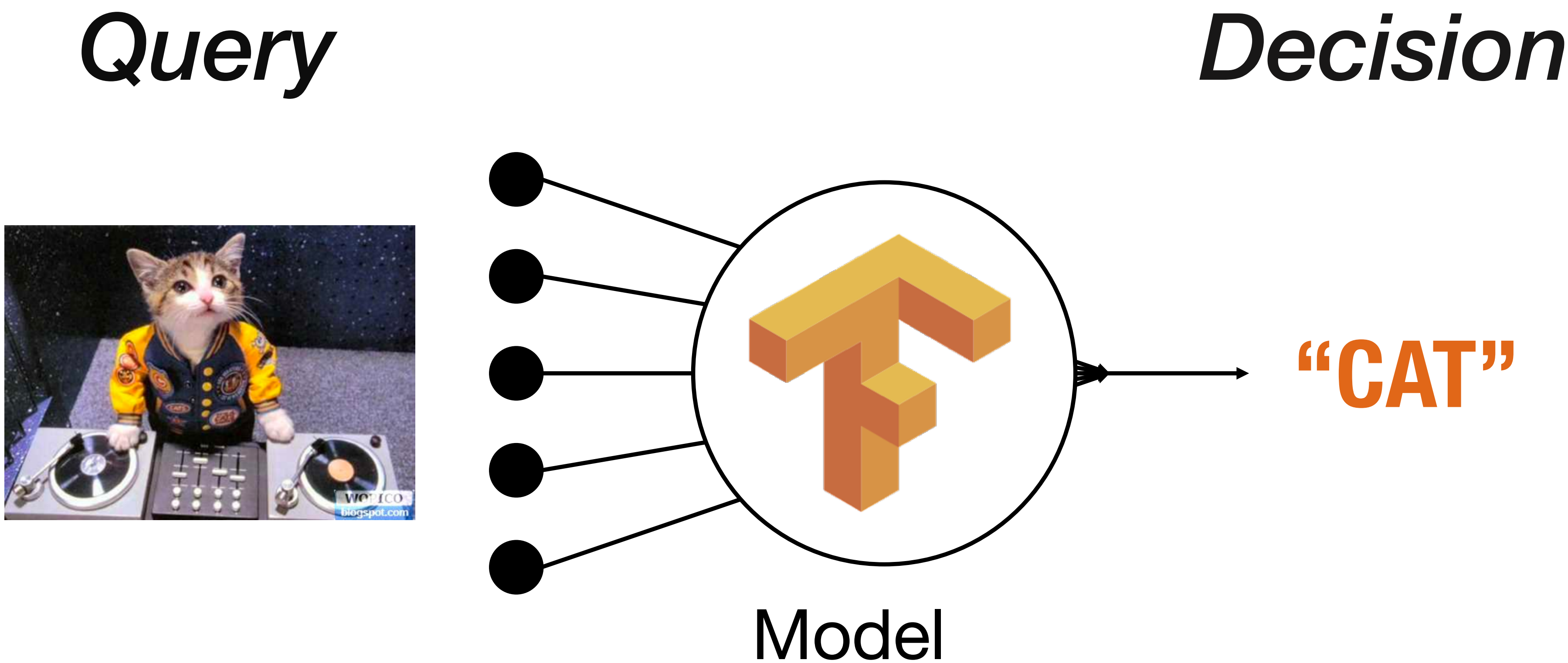
Mu Li[1], Li Zhou[1], Zichao Yang[1], Aaron Li[1], Fei Xia[1], David G. Andersen[1] and Alexander Smola[1,2]
[1]Carnegie Mellon University
[2]Google Strategic Technologies
{muli, lizhou, zichaoy, aaronli, feixia, dga}@cs.cmu.edu, alex@smola.org

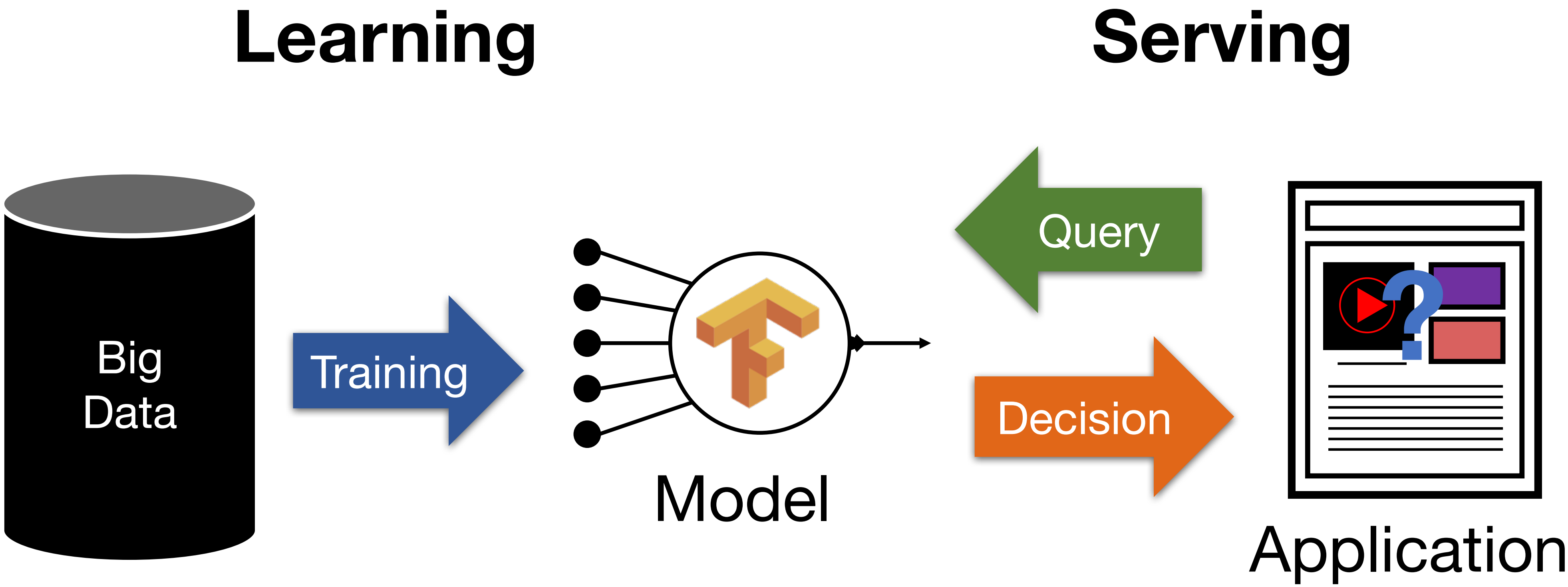
Abstract

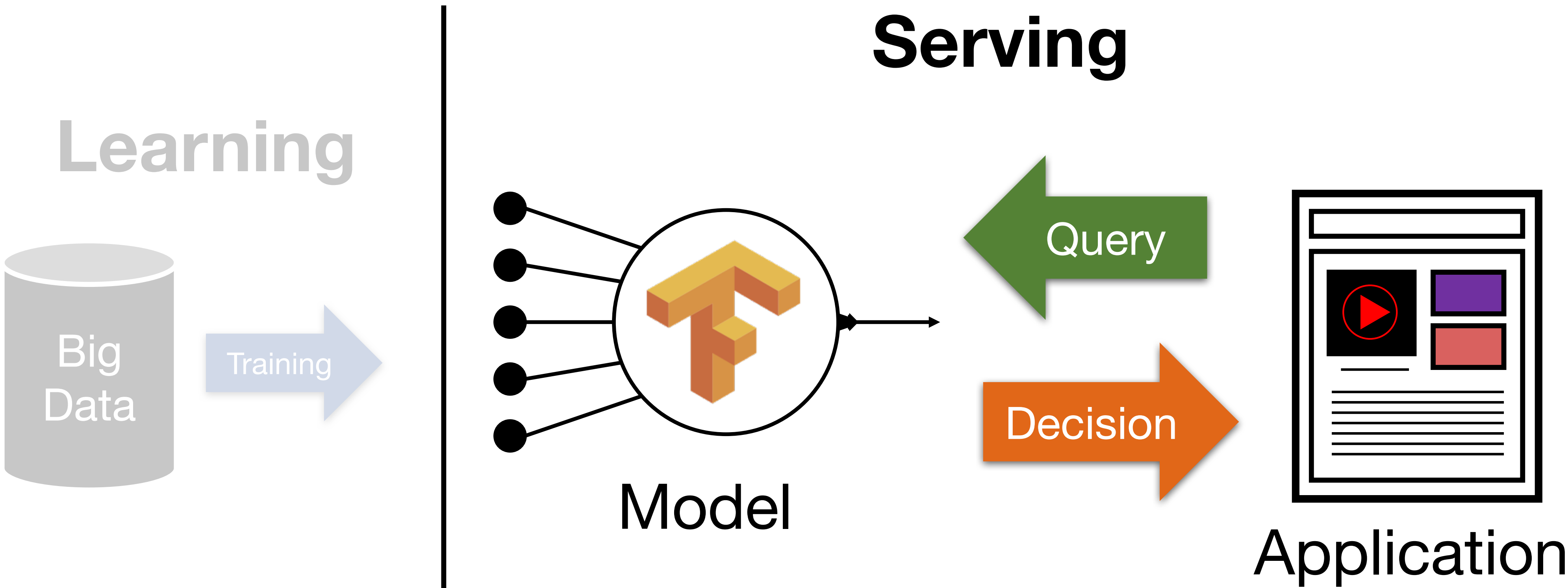
We propose a parameter server framework to solve distributed machine learning problems. Both data and workload are distributed into client nodes, while server nodes maintain globally shared parameters, which are represented as sparse vectors and matrices. The framework manages asynchronous data communications between clients and servers. Flexible consistency models, elastic scalability and fault tolerance are supported by this framework. We present algorithms and theoretical analysis for challenging nonconvex and nonsmooth problems. To demonstrate the scalability of the proposed framework, we show experimental results on real data with billions of parameters.

Learning

Learning Produces a Trained Model




Learning
Serving
Big Data
Training
Model
Query
Decision
Application



Prediction-Serving for interactive applications

Timescale: ~10s of milliseconds

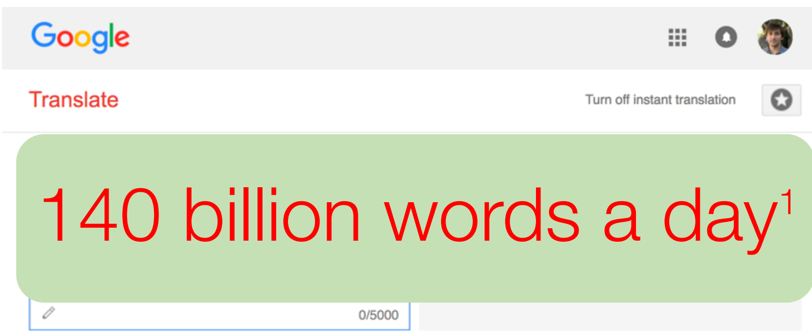
Google Translate

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]

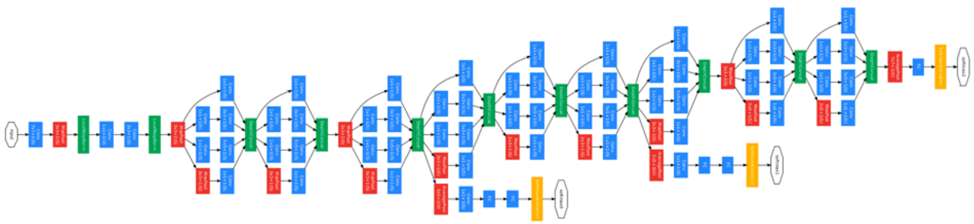
82,000 GPUs
running 24/7

Invented New Hardware!
Tensor Processing Unit (TPU)

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

Prediction-Serving Raises New Challenges

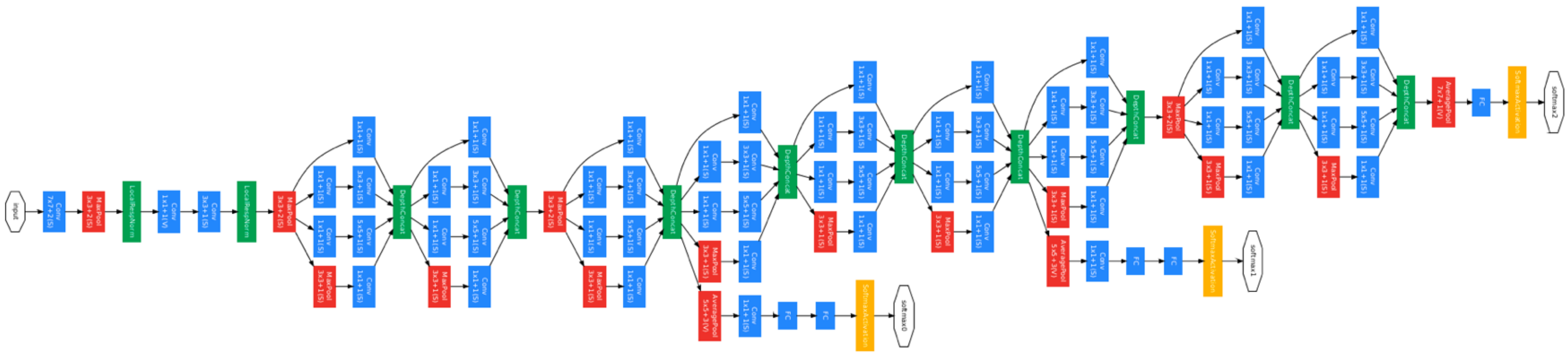
Prediction-Serving Challenges



Support low-latency, high-throughput serving workloads

Large and growing ecosystem of ML models and frameworks

Support low-latency, high-throughput serving workloads



Models getting more complex

- *10s of GFLOPs [1]*

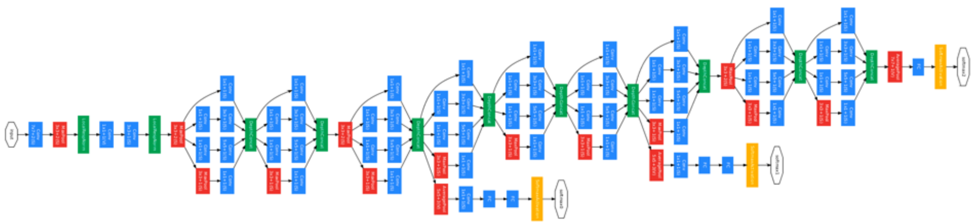
Deployed on critical path

- *Maintain SLOs under heavy load*

Using specialized hardware for predictions

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

Prediction-Serving Challenges



Support low-latency, high-throughput serving workloads

Large and growing ecosystem of ML models and frameworks

Large and growing ecosystem of ML models and frameworks

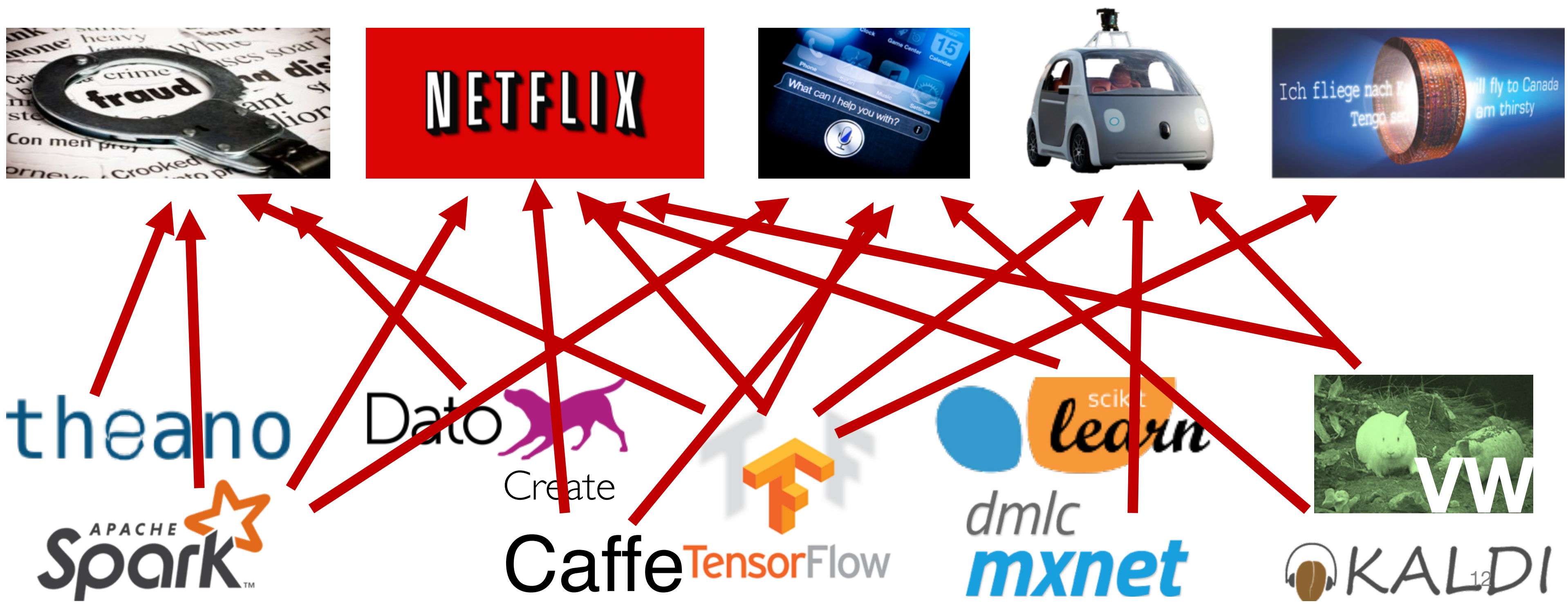
Fraud Detection

Content Rec.

Personal Asst.

Robotic Control

Machine Translation



Big Companies Build One-Off Systems

Problems:

- Expensive to build and maintain
  - Highly specialized and require ML and systems expertise
- Tightly-coupled model and application
  - Difficult to change or update model
- Only supports single ML framework

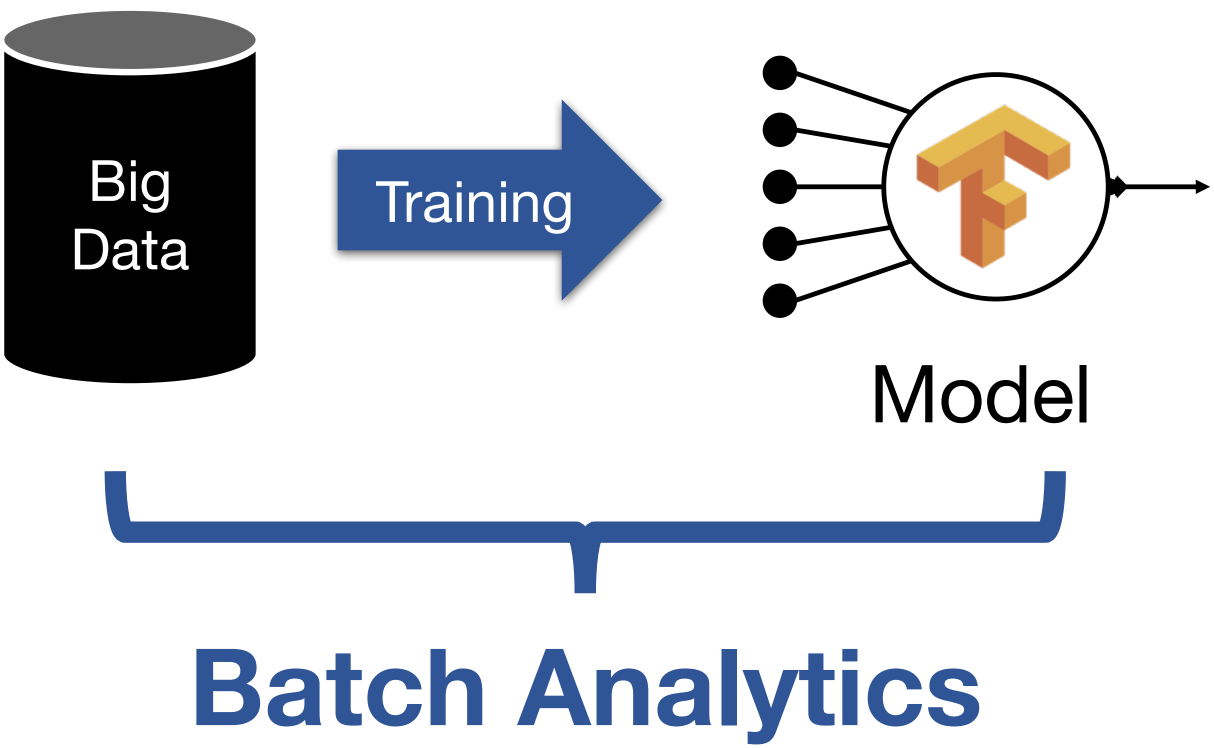
Large and growing ecosystem of ML models and frameworks

Difficult to deploy and brittle to manage

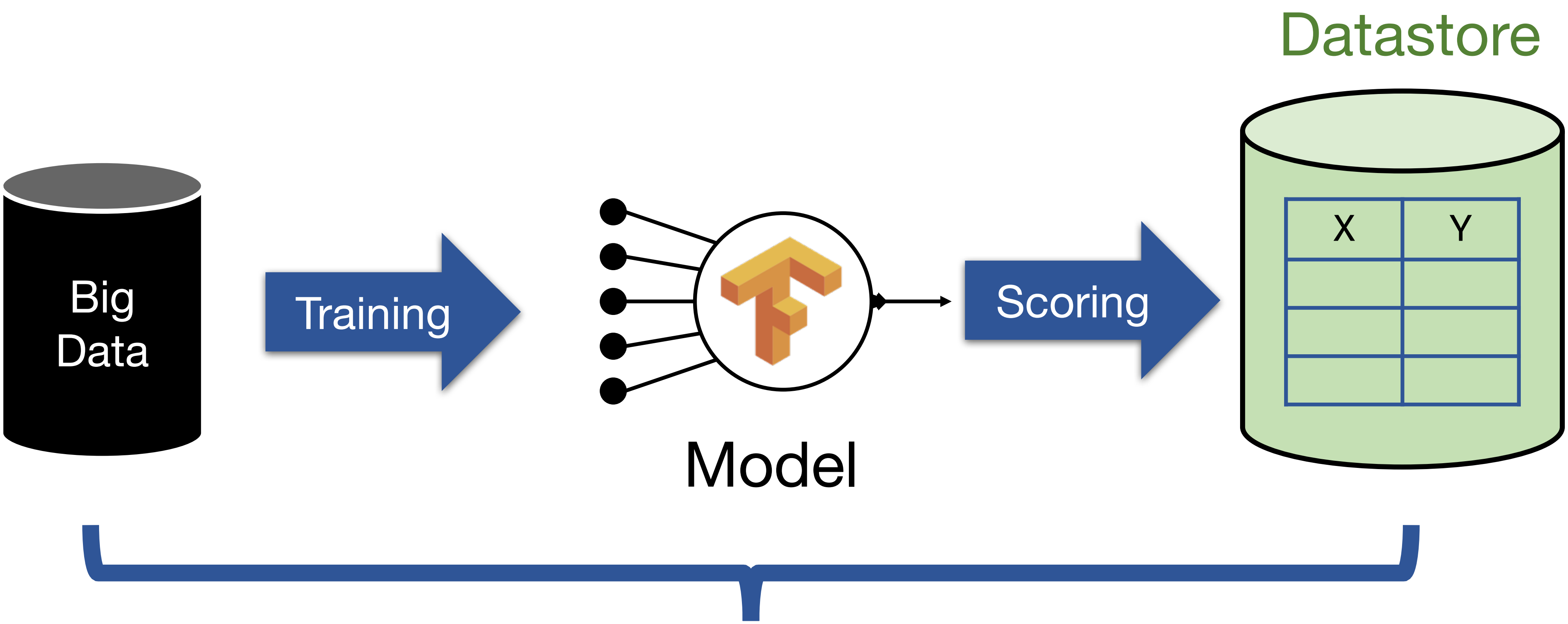
Varying physical resource requirements

But most companies can't build new serving systems…

Use existing systems: Offline Scoring



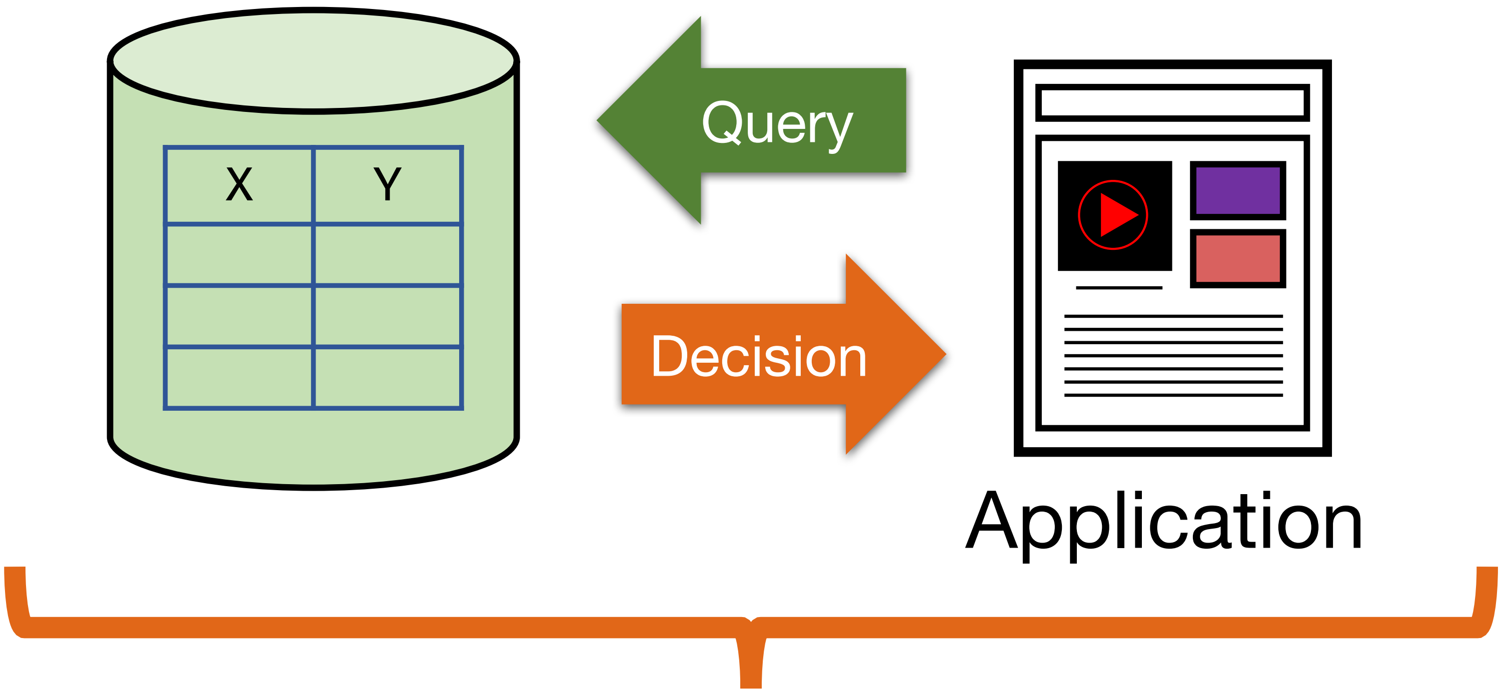
Use existing systems: Offline Scoring



Batch Analytics

Use existing systems: Offline Scoring

Look up decision in datastore



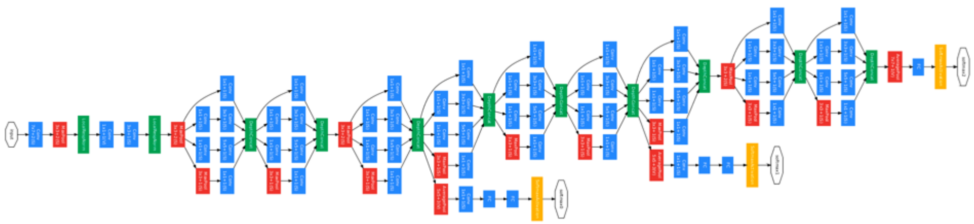
Low-Latency Serving

Use existing systems: Offline Scoring

Problems:

- Requires full set of queries ahead of time
  - Small and bounded input domain
- Wasted computation and space
  - Can render and store unneeded predictions
- Costly to update
  - Re-run batch job

Prediction-Serving Challenges



Support low-latency, high-throughput serving workloads

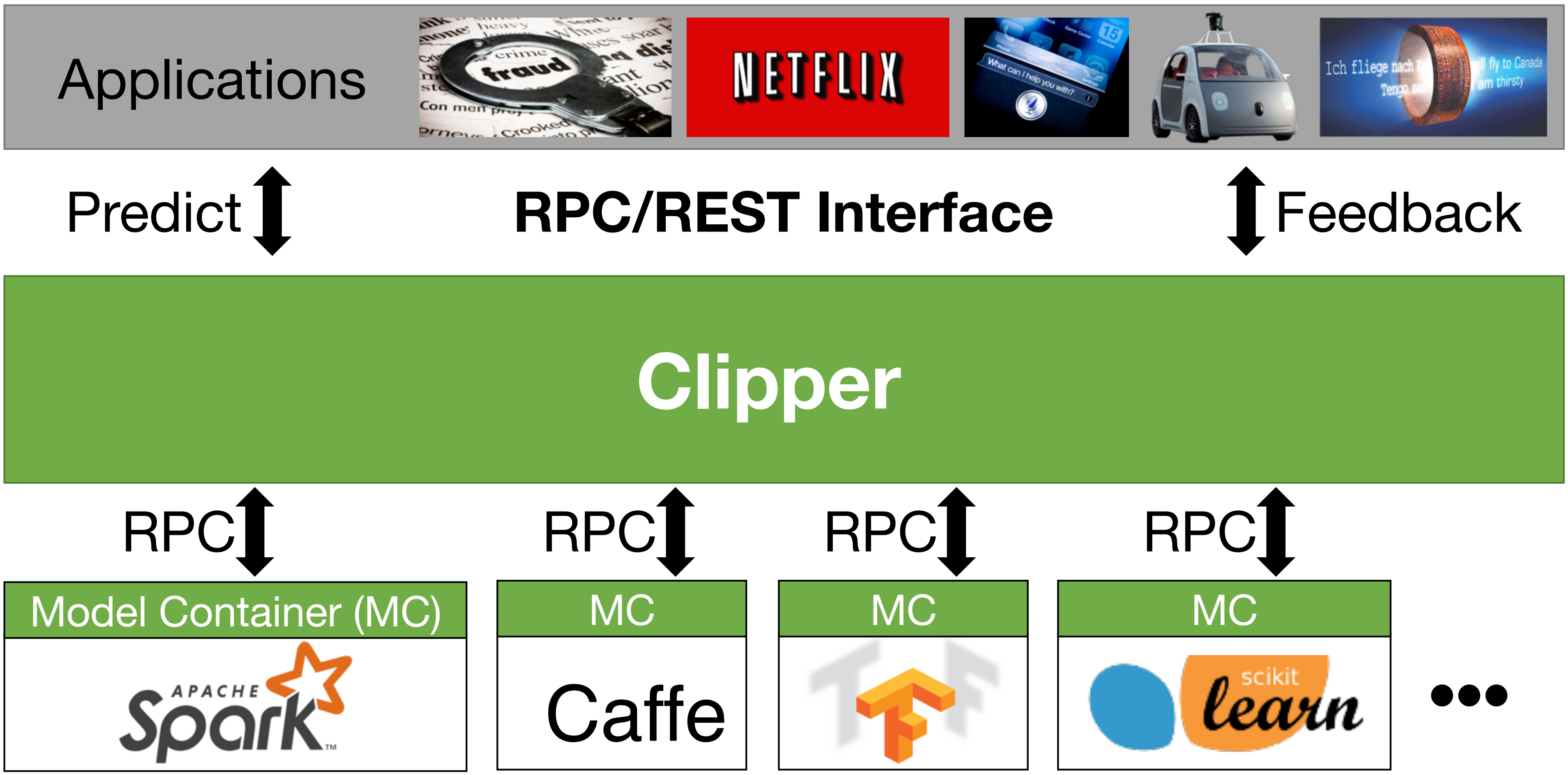
Large and growing ecosystem of ML models and frameworks

How does Clipper address these challenges?

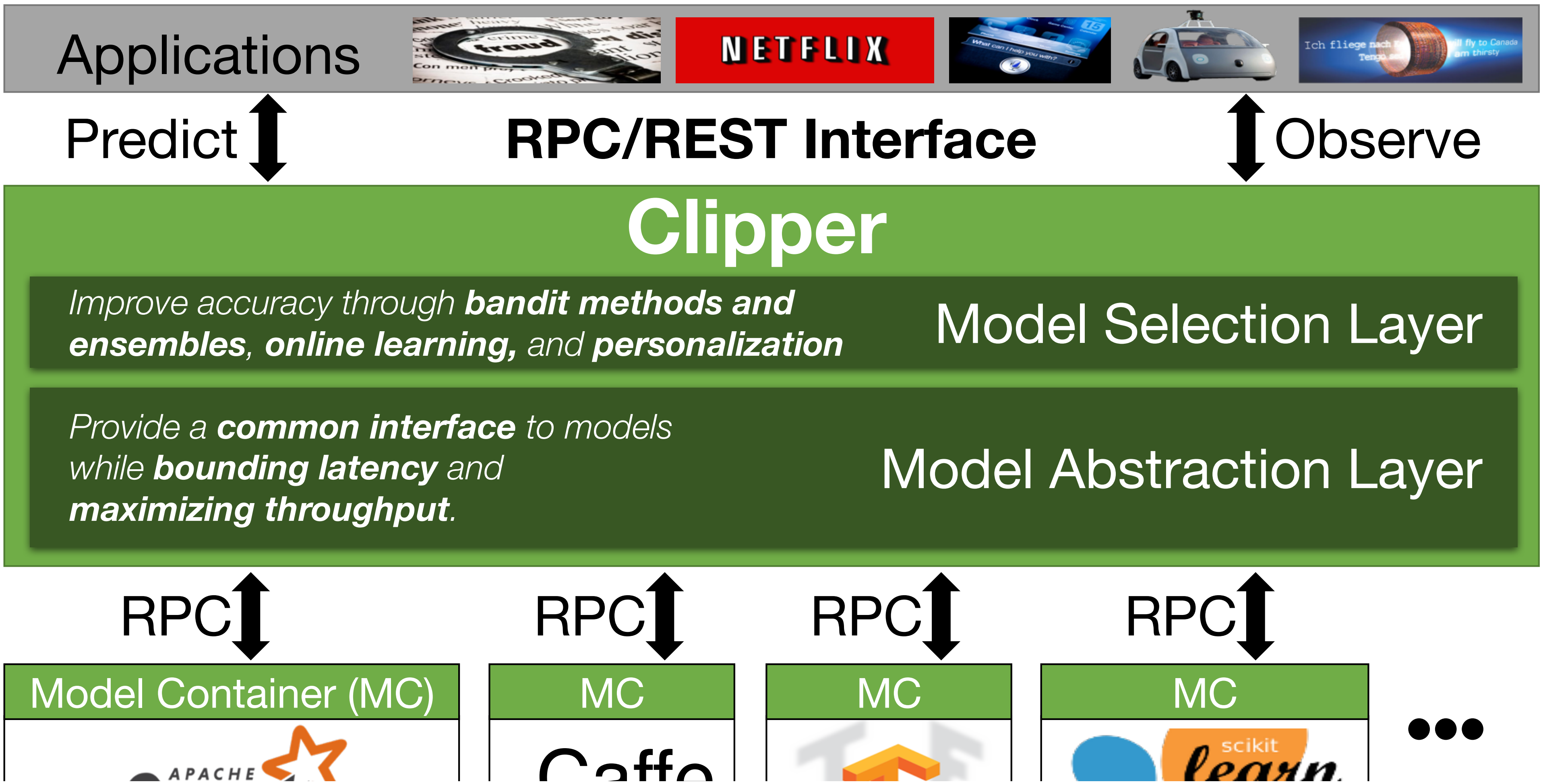
Clipper Solutions

- *Simplifies deployment through layered architecture*
- *Serves many models across ML frameworks concurrently*
- *Employs caching, batching, scale-out for high-performance serving*

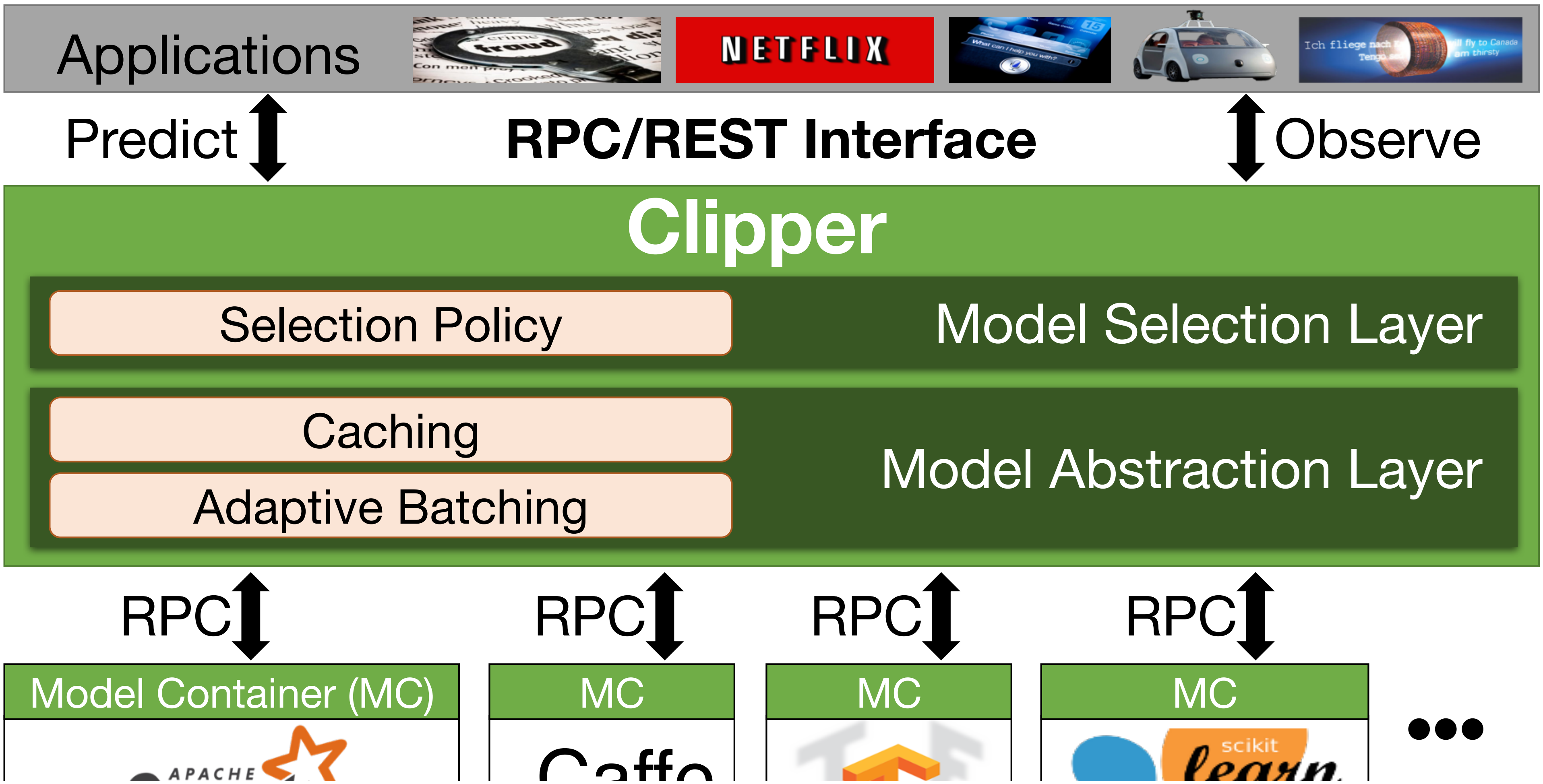
Clipper Decouples Applications and Models

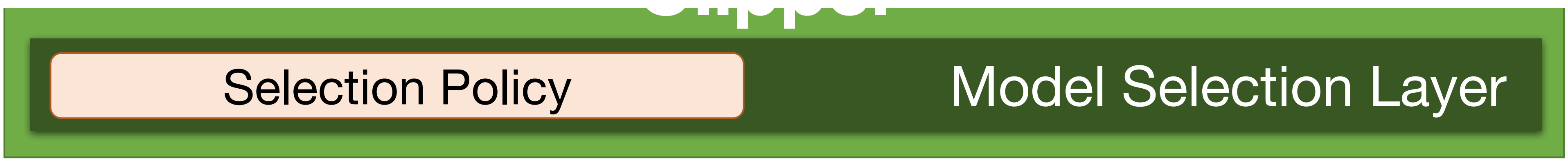


Clipper Architecture



Clipper Architecture





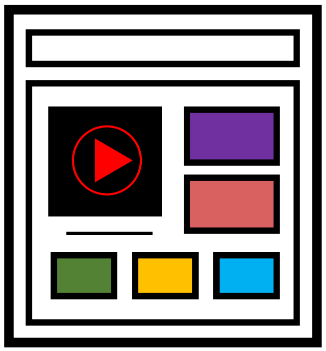
Selection policies supported by Clipper

- Exploit multiple models to estimate confidence
- Use multi-armed bandit algorithms to learn optimal model-selection online
- Online personalization across ML frameworks

**See paper for details [NSDI 2017]*

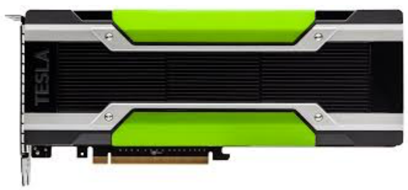
Batching to Improve Throughput

- Why batching helps:



A single page load may generate many queries

Hardware Acceleration

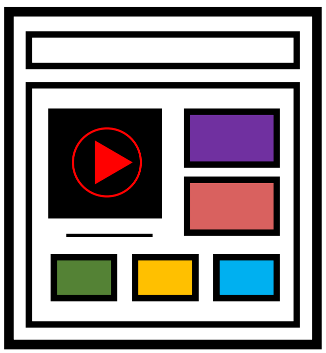


Helps amortize system overhead

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

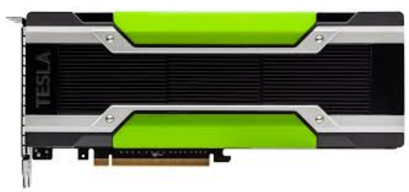
Adaptive Batching to Improve Throughput

- Why batching helps:



A single page load may generate many queries

Hardware Acceleration



Helps amortize system overhead

- 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**)

Conclusion

- *Prediction-serving* is an important and *challenging* area for *systems research*
  - Support *low-latency, high-throughput* serving workloads
  - Serve *large* and growing *ecosystem of ML frameworks*
- *Clipper* is a *first step* towards addressing these challenges
  - *Simplifies deployment* through layered architecture
  - Serves many models *across ML frameworks* concurrently
  - Employs *caching, adaptive batching, container scale-out* to meet interactive serving workload demands
- Beyond academic prototype to build a real, *open-source system*

https://github.com/ucbrise/clipper

crankshaw@cs.berkeley.edu