

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

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeff Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng

Google Brain

Abstract

datasets, and moving them to GPUs. TensorFlow is a machine learning system that operates at first-generation system.

Spark: Cluster Computing with Working Sets

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

MapReduce/Dryad job, each job must reload the data from disk, incurring a significant performance penalty. In Spark, data is loaded once and then can be reused by successful applications. These systems are designed for that is not this paper focuses on reusing data across operations. We propose a new algorithm, which supports applications that reuse data across operations. This paper presents a new cluster computing framework that supports applications that reuse data across operations.

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

Trishul Chilimbi Yutaka Suzue Johnson Apacible Karthik Kalyanaraman

Microsoft Research

ABSTRACT

Large deep neural networks have demonstrated state-of-the-art accuracy in many computer vision tasks. However, training these models is extremely time consuming and requires a large amount of compute. We present Project Adam, an implementation of a deep learning training system co-designed for hardware and software. Project Adam achieves high efficiency and task accuracy on a wide range of models and tasks. Our results show that Project Adam is more efficient and accurate than existing training systems using current training hardware.

1. INTRODUCTION
Traditional statistical models of data and a table correspond to columns correspond to underlying data set. Algorithms can be applied to the data to extract features and learn models. Industrial applications

Caffe: Convolutional Architecture for Fast Feature Embedding*

Yangqing Jia¹, Evan Shelhamer¹, 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
{jia, shelhamer, jdonahue, sergey, 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.

Caffe is maintained and developed by the Berkeley Vision and Learning Center (BVLC) with the help of an active community of contributors on GitHub. It powers online research pipelines, from multi-modal industrial applications

1. INTRODUCTION

A key problem in multimedia data analysis is discovery of effective representations for sensory inputs—images, sound, waves, haptics, etc. While performance of conventional, handcrafted features has plateaued in recent years, new developments in deep compositional architectures have kept performance levels rising [8]. Deep models have outperformed hand-engineered feature representations in many domains, and made learning possible in domains where engineered features were lacking entirely.

We are particularly motivated by large-scale visual recognition, where a specific type of deep architecture has achieved a commanding lead on the state-of-the-art. These *Convolutional Neural Networks*, or CNNs, are discriminatively trained via back-propagation through layers of convolutional filters and other operations such as rectification and pooling. Following the early success of digit classification in the 90's, these models have recently surpassed all known methods for large-scale visual recognition and have been adopted for

GraphLab: A New Framework For Parallel Machine Learning

Yucheng Low
Carnegie Mellon University
ylow@cs.cmu.edu

Joseph Gonzalez
Carnegie Mellon University
jgonzal@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

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

Abstract

PageRank Connected K-Means Triangle Count

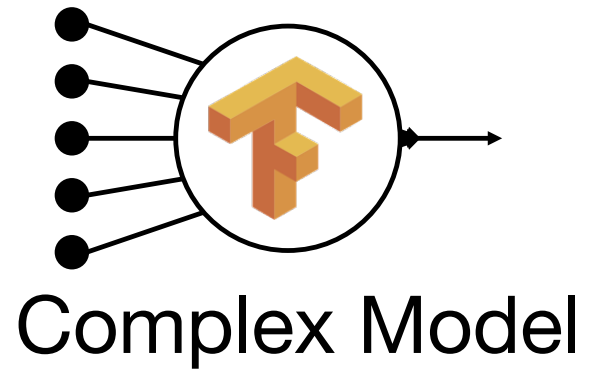
Parameter Server for Distributed Machine Learning

Mu Li¹, Li Zhou¹, Zichao Yang¹, Aaron Li¹, Fei Xia¹, David G. Andersen¹ and Alexander Smola^{1,2}
¹Carnegie Mellon University
²Google Strategic Technologies
{muli, lizhou, zichao, aaronli, fetxia, 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

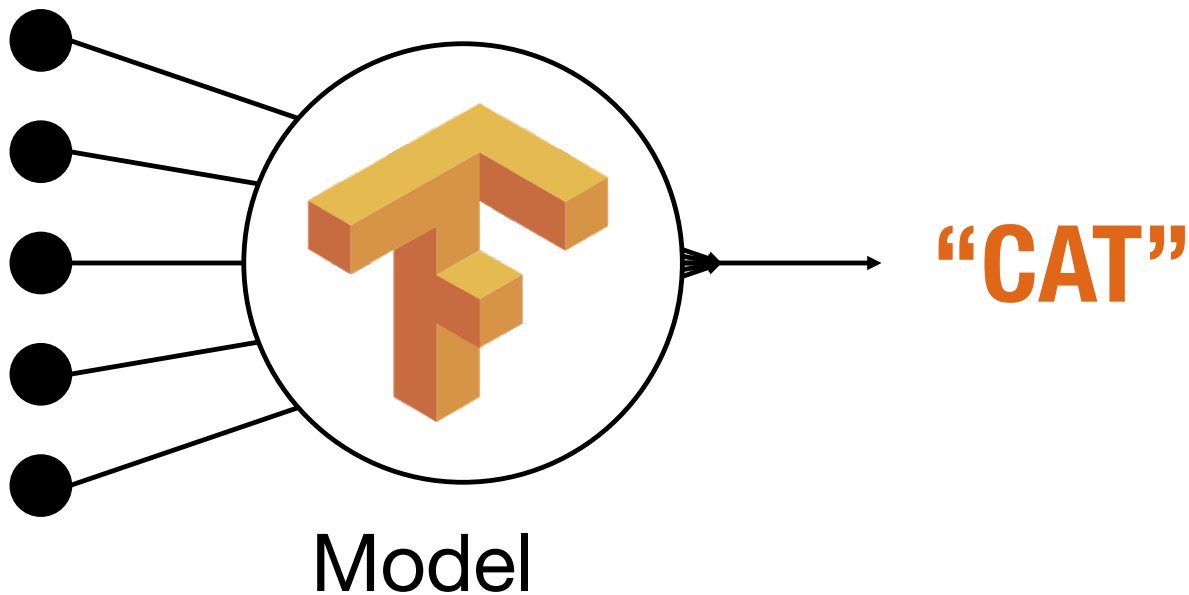


Learning Produces a Trained Model

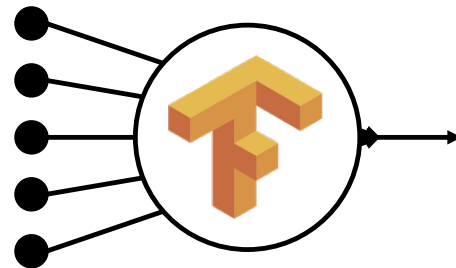
Query



Decision

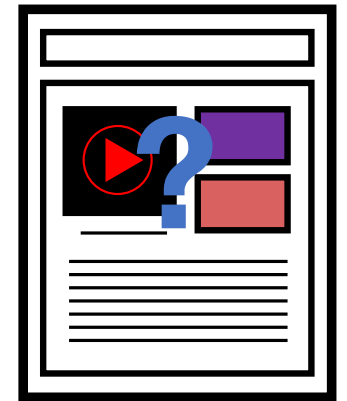
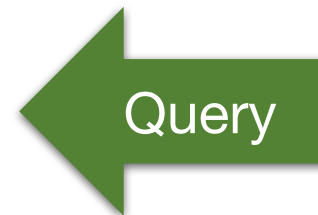


Learning

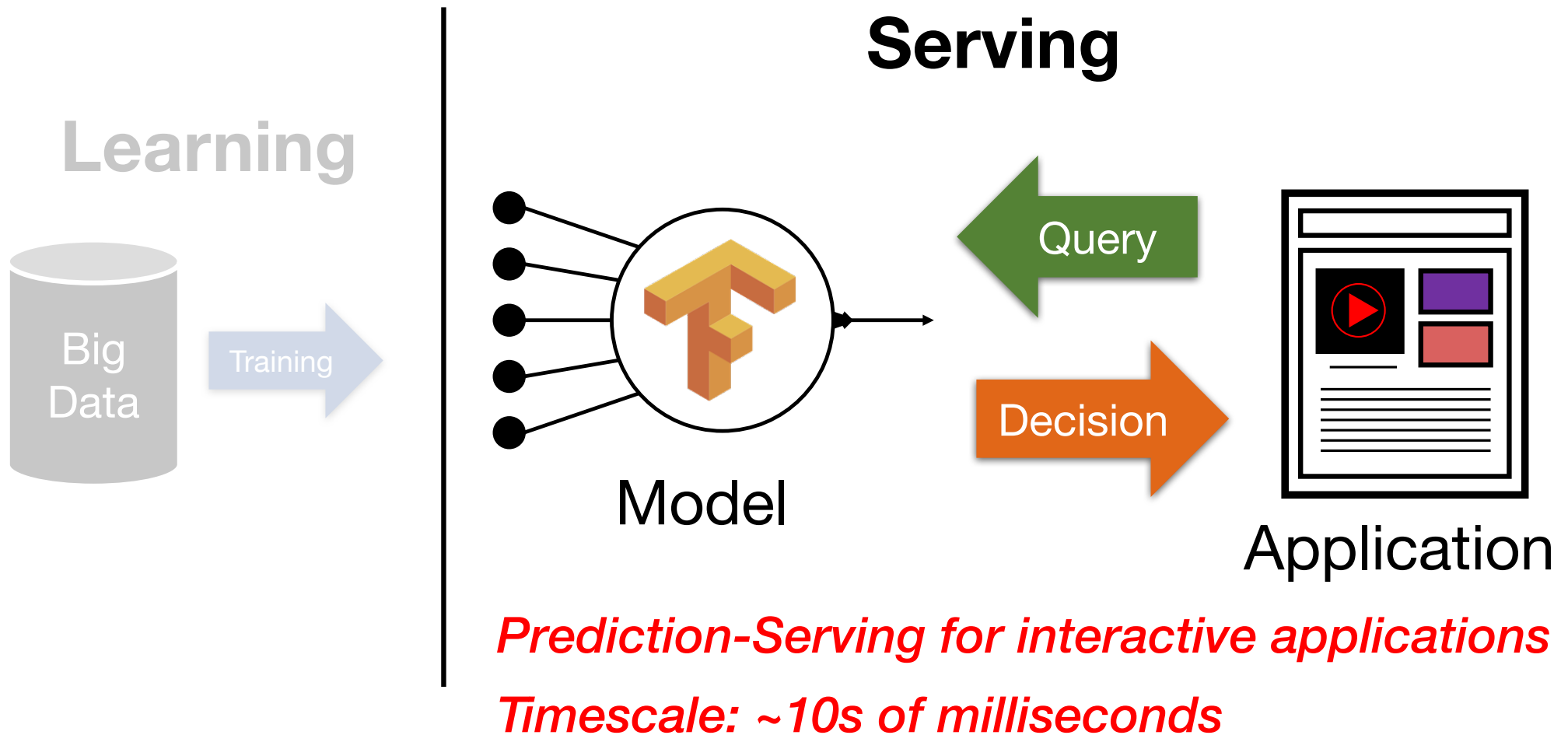


Model

Serving



Application



Google Translate

Serving



82,000 GPUs
running 24/7

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

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

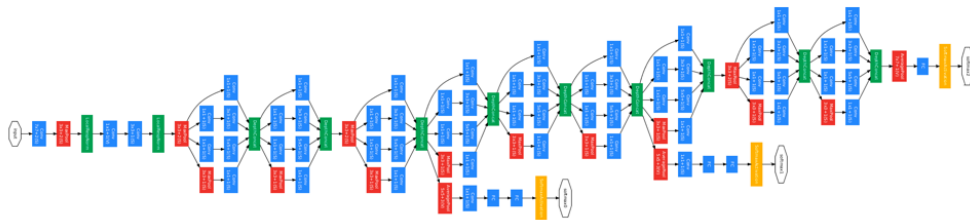
Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi
yonghui,schuster,zhifengc,qvl,mnorouzi@google.com

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

***Invented New Hardware!
Tensor Processing Unit
(TPU)***

Prediction-Serving Raises New Challenges

Prediction-Serving Challenges

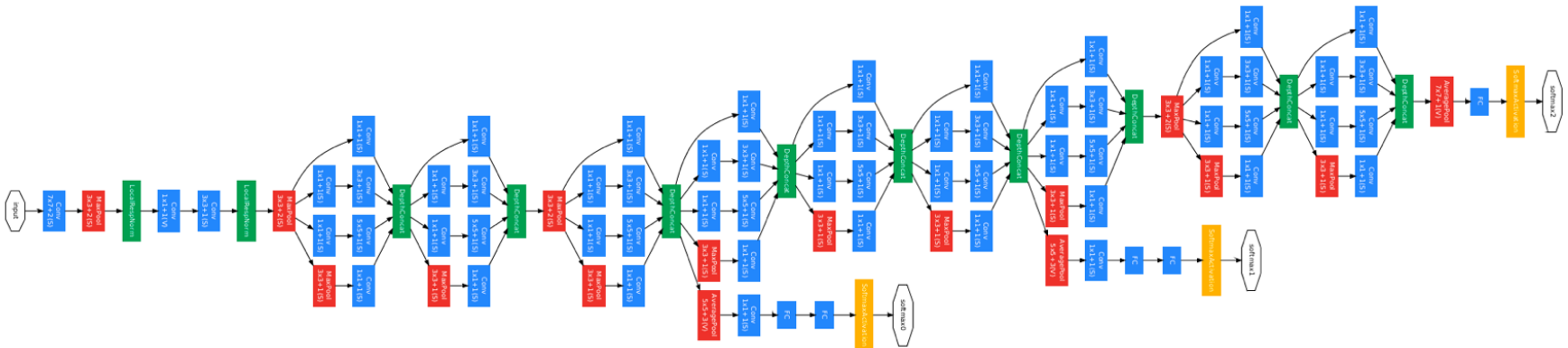


Support low-latency, high-throughput serving workloads



Large and growing ecosystem of ML models and frameworks

Support low-latency, high-throughput serving workloads

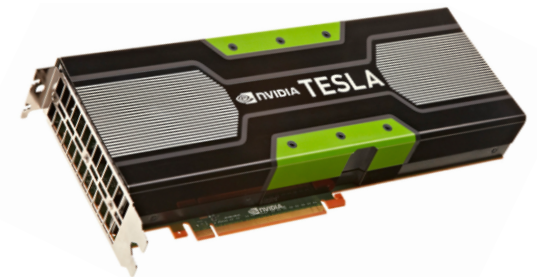


Models getting more complex

- 10s of GFLOPs [1]

Deployed on critical path

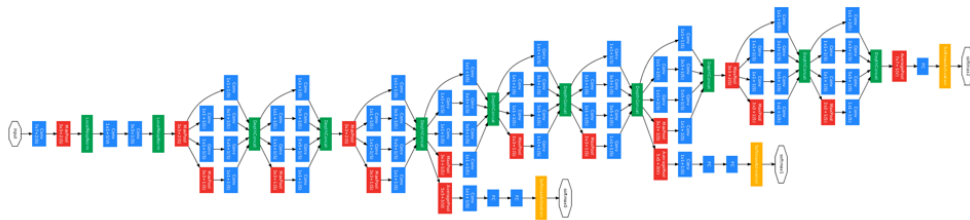
- Maintain SLOs under heavy load



Using specialized hardware
for predictions

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

Prediction-Serving Challenges

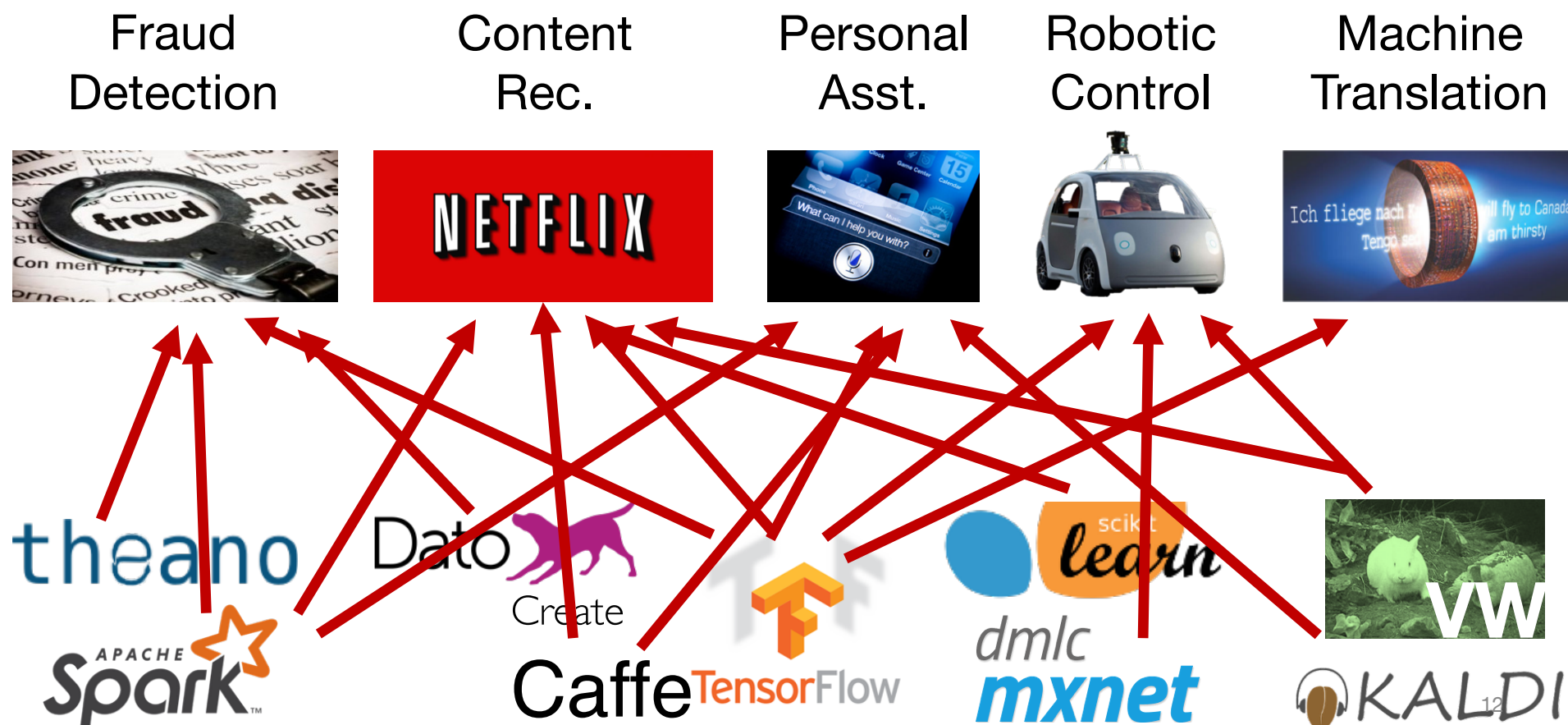


Support low-latency, high-throughput serving workloads



Large and growing ecosystem of ML models and frameworks

Large and growing ecosystem of ML models and frameworks



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



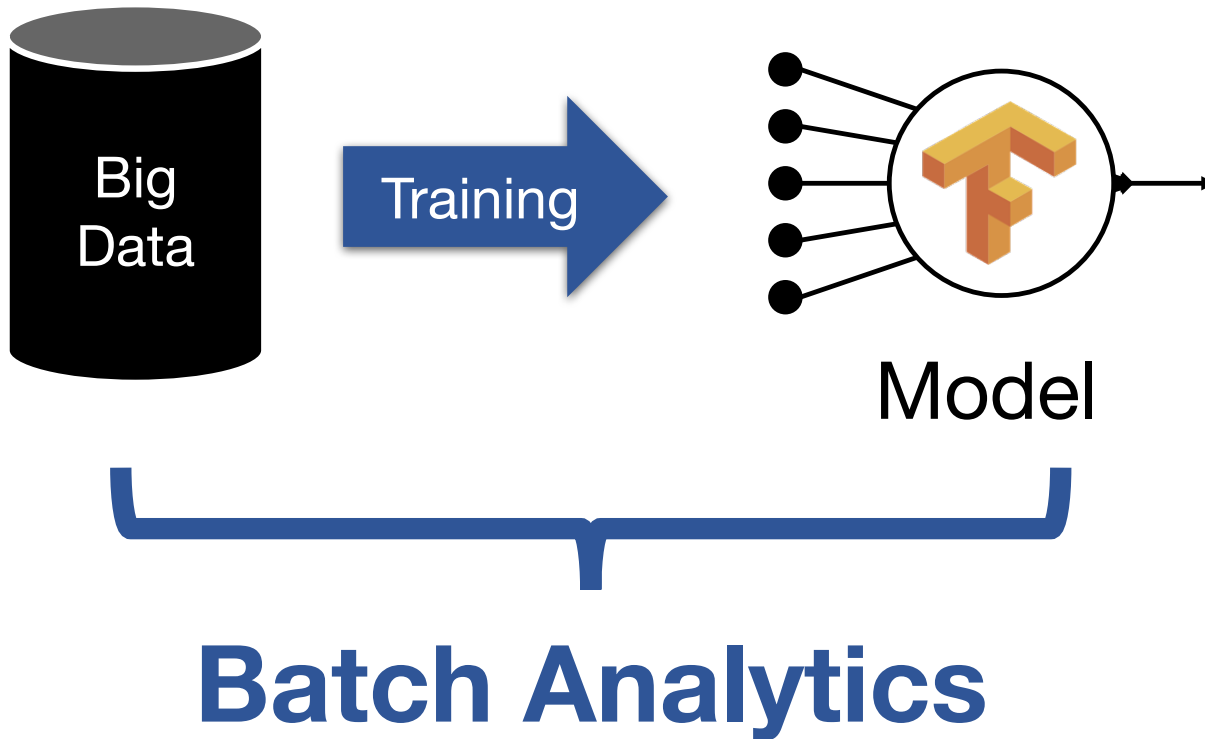
Large and growing ecosystem of ML models and frameworks

***Difficult to deploy and
brittle to manage***

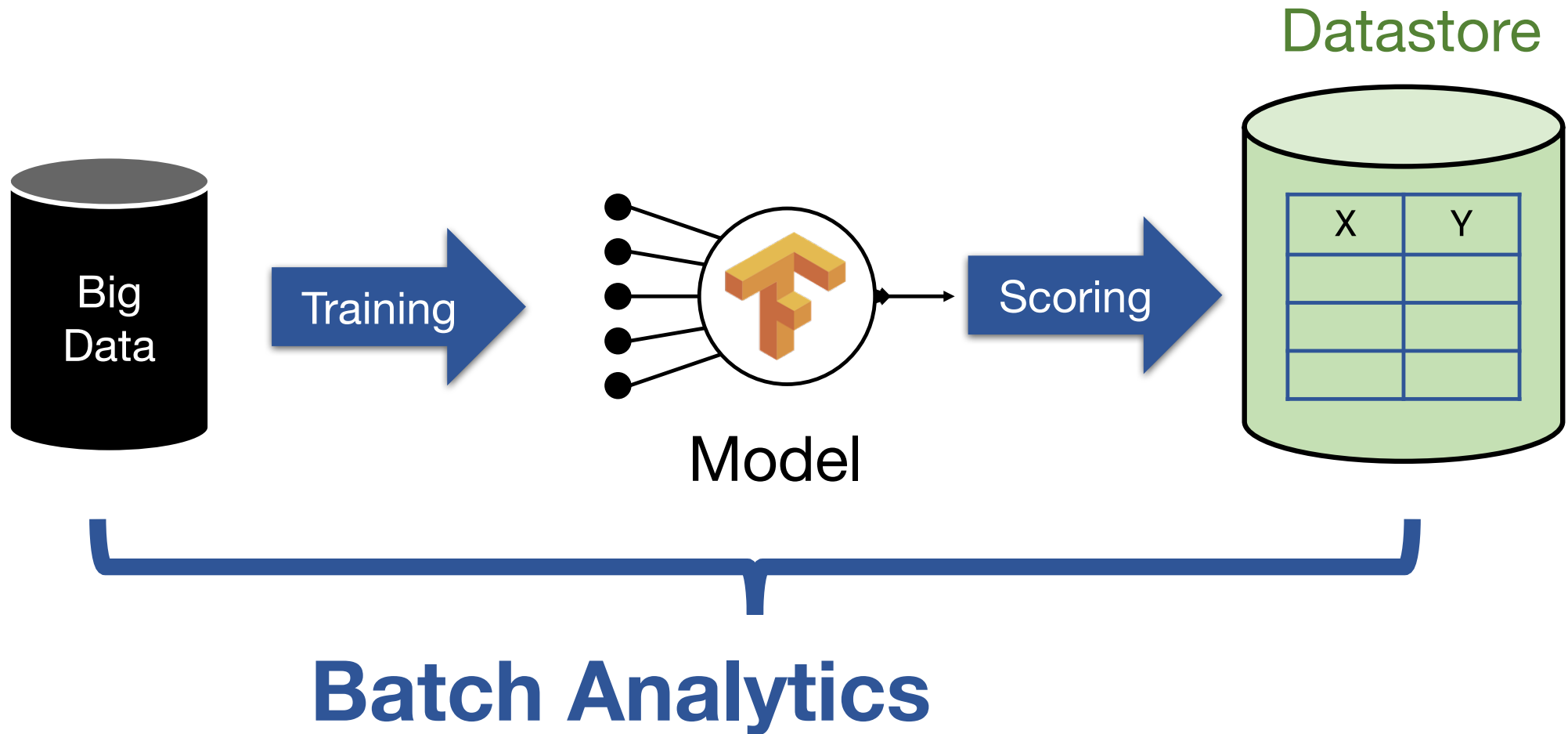


*But most companies
can't build new
serving systems...*

Use existing systems: Offline Scoring

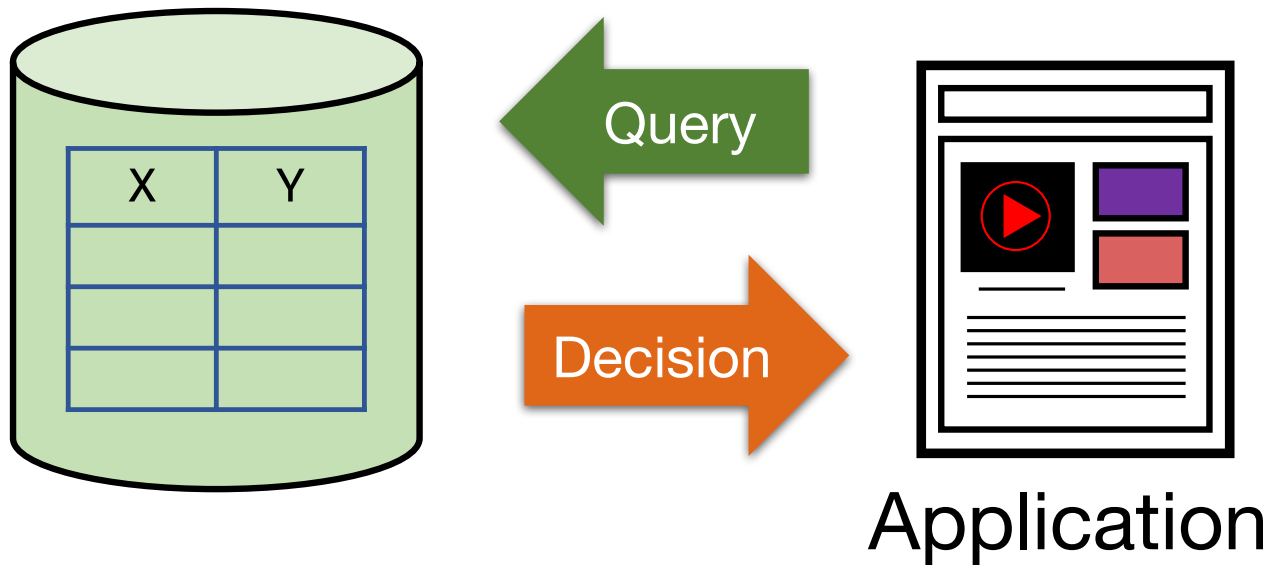


Use existing systems: Offline Scoring



Use existing systems: Offline Scoring

Look up decision in datastore



Low-Latency Serving

Use existing systems: Offline Scoring

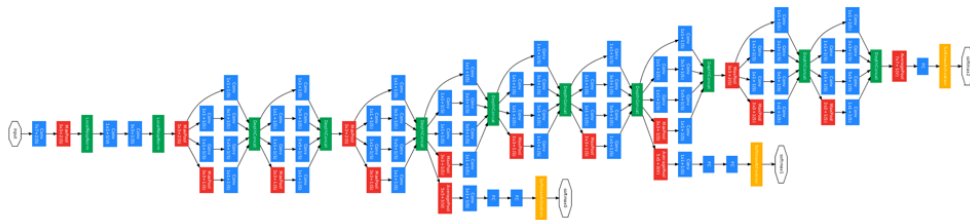
Look up decision in datastore

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

Low-Latency Serving

Prediction-Serving Challenges



Support low-latency, high-throughput serving workloads



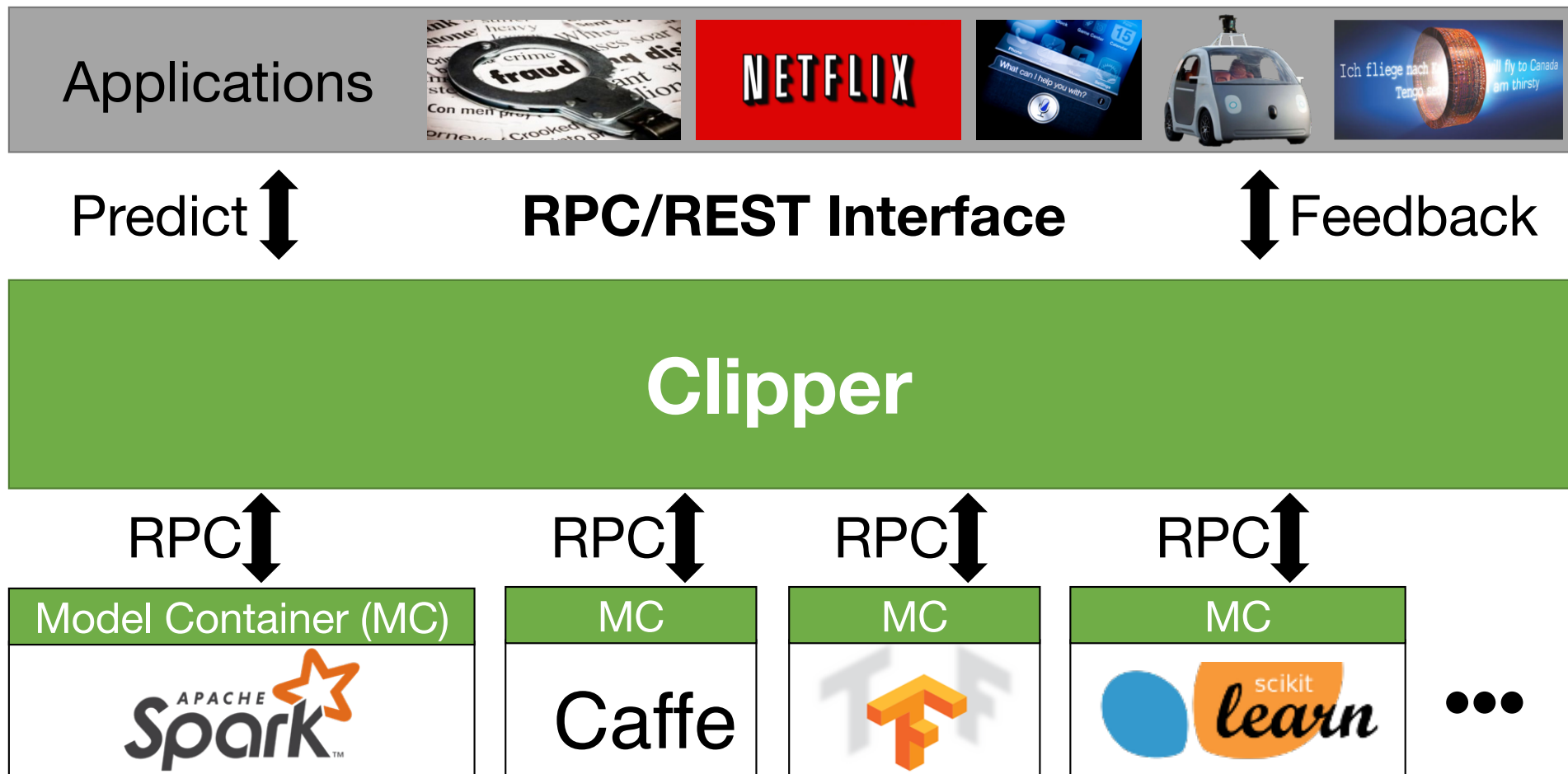
Large and growing ecosystem of ML models and frameworks

*How does Clipper address
these challenges?*

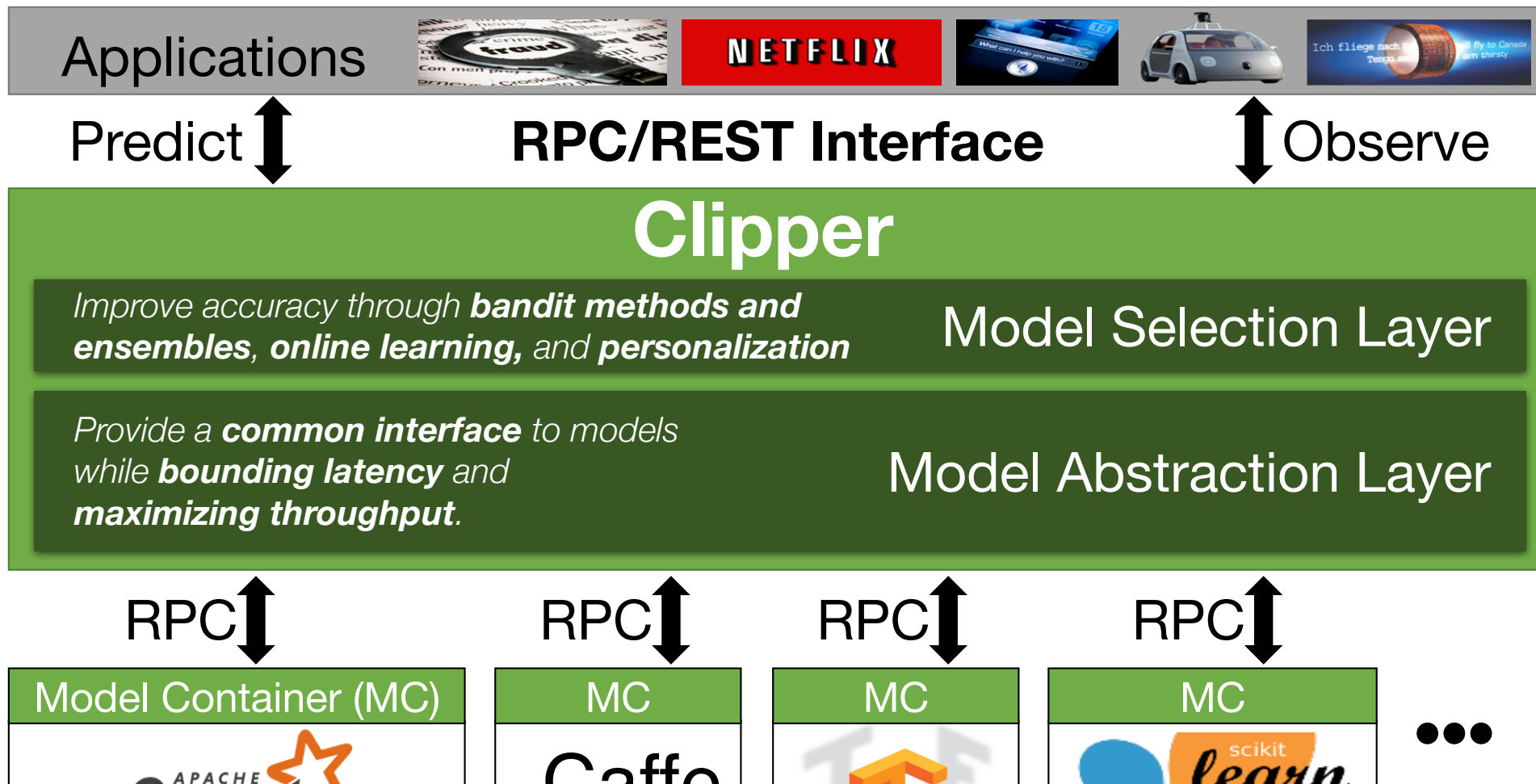
Clipper Solutions

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

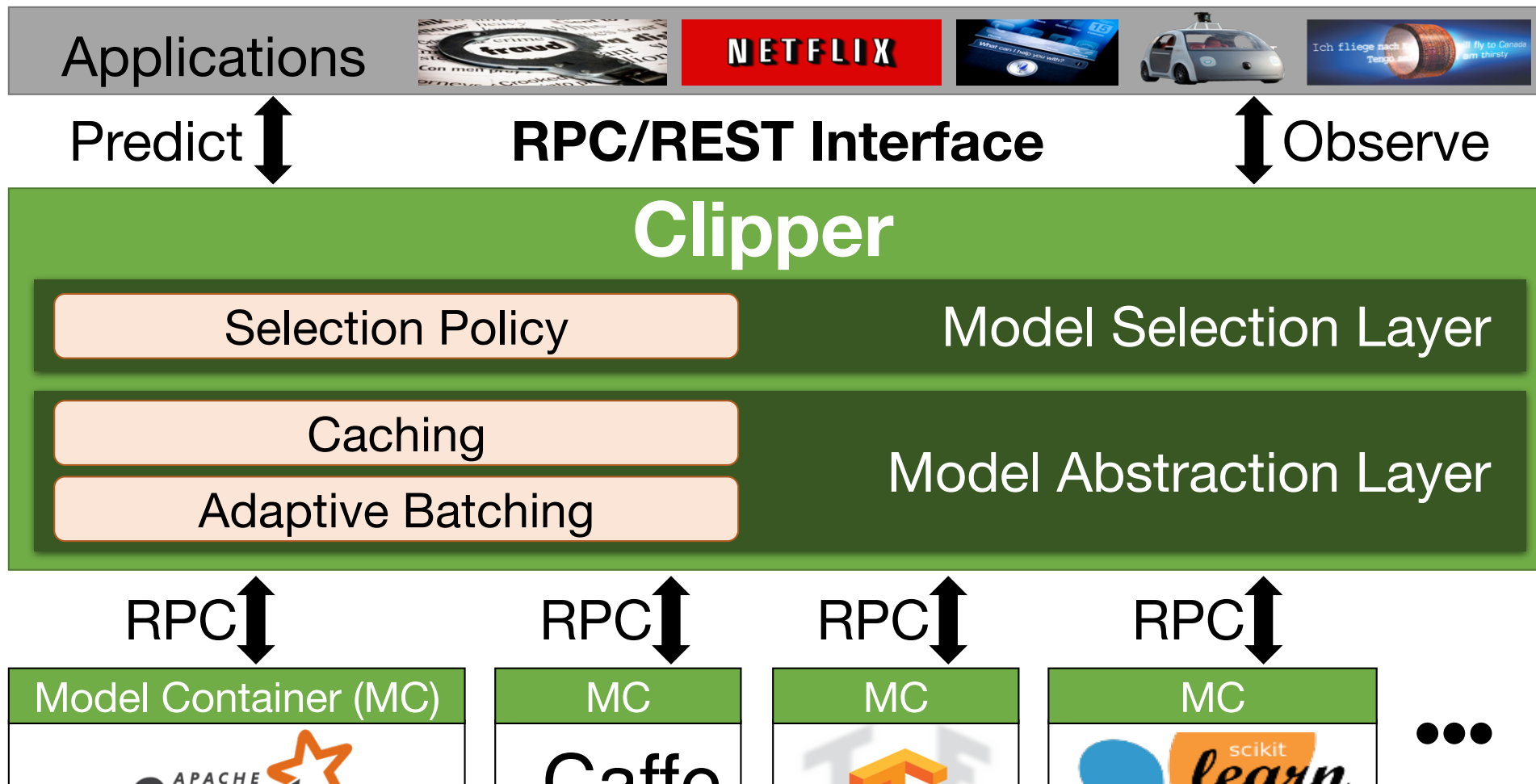
Clipper Decouples Applications and Models




Clipper Architecture



Clipper Architecture





The diagram shows a green horizontal bar at the top. Inside this bar, on the left, is a light orange rounded rectangle labeled 'Selection Policy'. To its right, within the same green bar, is a dark green rounded rectangle labeled 'Model Selection Layer'.

Selection Policy

Model Selection Layer

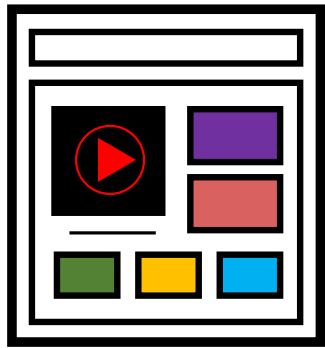
Selection policies supported by Clipper

- 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]*

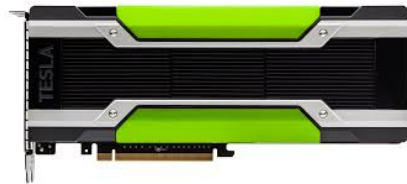
Batching to Improve Throughput

- Why batching helps:



A single page load may generate many queries

Hardware Acceleration

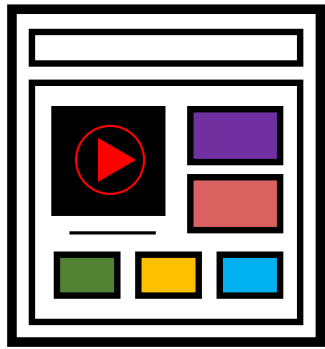


Helps amortize system overhead

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

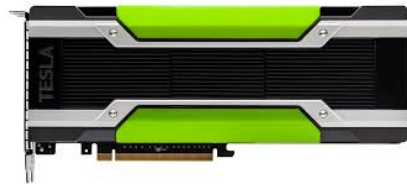
Adaptive Batching to Improve Throughput

- 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