Deploying Interactive Machine Learning Applications with Clipper

Joseph E. Gonzalez
Co-director of the RISE Lab
jegonzal@cs.berkeley.edu
Managing the Machine Learning Lifecycle

Joseph E. Gonzalez
Co-director of the RISE Lab
jegonzal@cs.berkeley.edu
About Me

- Co-director of the RISE Lab
- Co-founder of Turi Inc.
- Member of the Apache Spark PMC

Research

- Artificial Intelligence
- Data Science
- Distributed Data Systems
- Graph Processing Systems
Conjecture

**Machine learning models are the next “big data”**

Evidence

1. Everyone is talking about models but few have them.
2. They have the opportunity to transform industries.
3. They are a consequence of mastering big data.
4. Today, their full value is only realized with advanced skills and technologies
Conjecture

Machine learning models are the next “big data”

Corollaries

Data Engineers will need to manage data & machine learning models

We need new technologies to manage the machine learning lifecycle
What is the Machine Learning Lifecycle?

Model Development

Offline Training Data → Data Collection → Cleaning & Visualization → Feature Eng. & Model Design → Training & Validation

Training

Training Pipelines → Trained Models → Validation

Inference

Prediction Service → Logic → Feedback → End User Application

Data Scientist

Data Engineer

Data Engineer

Training Pipelines

Data

Prediction

Query

Live Data
Model Development

Data Collection → Cleaning & Visualization

Training & Validation ← Feature Eng. & Model Design

Help!

Identifying potential sources of data

Joining data from multiple sources

Addressing **missing values** and **outliers**

Plotting trends to identify **anomalies**
Model Development

Identifying potential sources of data

Joining data from multiple sources

Addressing missing values and outliers

Plotting trends to identify anomalies

Offline Training Data

Data Collection → Cleaning & Visualization

Training & Validation ← Feature Eng. & Model Design

Help!

Happy to JOIN

Data Scientist

Data Engineer
In Data Science, 80% of time spent prepare data, 20% of time spent complain about need for prepare data.
Model Development

Data Collection ➔ Cleaning & Visualization

- Identifying potential sources of data
- Joining data from multiple sources
- Addressing missing values and outliers
- Plotting trends to identify anomalies

Data Collection ➔ Training & Validation

- Offline Training Data

Help!

Feature Eng. & Model Design

Happy to JOIN

Data Scientist

Data Engineer
Model Development

- Offline Training Data
- Data Collection → Cleaning & Visualization
- Feature Eng. & Model Design → Training & Validation

I was born for this!

Building informative features functions

Designing new model architectures

Tuning training algos.

Validating prediction accuracy

Data Scientist
Model Development Technologies

- Offline Training Data
- Data Collection → Cleaning & Visualization
- Training & Validation ← Feature Eng. & Model Design
- ClearML
- TensorFlow
- R
- PyTorch
- Keras
- Jupyter
- MXNet
- DMLC
- XGBoost
- Matplotlib
- Caffe2
- Pandas
- NumPy
- Apache Spark
- Dask
- Hive
What is the output of Model Development

Offline Training Data → Data Collection → Cleaning & Visualization → Training & Validation ← Feature Eng. & Model Design

Trained Model → Reports & Dashboards

(insights …)
A learned function from a **query** to a **prediction**

Trained Model

“CAT”

consisting of **parameters** and **model structure**.

**Data** (10B to 10GB)

How to use the parameters...
What is the output of Model Development

- Offline Training Data
- Data Collection
- Cleaning & Visualization
- Training & Validation
- Feature Eng. & Model Design
- Trained Model
- Reports & Dashboards (insights …)

Bad Idea
Why is it a **Bad Idea** to directly produce trained models from model development?

With just a trained model we are **unable to**

1. retrain models with new data
2. track data and code for **debugging**
3. capture **dependencies** for deployment
4. audit training for **compliance** (e.g., GDPR)
What is the output of Model Development

- Offline Training Data
- Data Collection
- Cleaning & Visualization
- Feature Eng. & Model Design
- Training & Validation
- Trained Models

Reports & Dashboards

(Insights …)

Bad Idea
What is the output of Model Development

- Offline Training Data
- Data Collection
- Cleaning & Visualization
- Feature Eng. & Model Design
- Training & Validation
- Training Pipelines

Reports & Dashboards
(Insights …)
Training Pipelines Capture the **Code and Data Dependencies**

- Description of how to train the model from data sources

---

**Training Pipelines**

- Training Data → Training Pipelines
- Trained Models → Trained Models → Binaries

---

**Software Engineering Analogy**

- Training Pipelines → Code
- Trained Models → Binaries
What is the output of Model Development

- Offline Training Data
- Data Collection → Cleaning & Visualization
- Training & Validation ← Feature Eng. & Model Design

Reports & Dashboards
(Insights ...)

Training Pipelines
Training

Training models **at scale** on **live data**

Retraining on new data

Automatically **validate** prediction accuracy

Manage model **versioning**

Requires **minimal expertise** in machine learning
Training Technologies

Workflow Management:
- Apache Airflow
- Azkaban
- Luigi
- Oozie

Scalable Training:
- PyTorch
- TensorFlow
- XGBoost
- Spark
- mxnet
Open Problems

Context & Composition
Context  How, What, & Who?

- **How** was the model or data created?
- **What** is the latest or best version?
- **Who** is responsible? (blame...)

Partial Solution

Track relationships between
1. Code versions  ✓ git
2. Model & Data versions
3. People (versions?)
Composition

Models are being composed to solve new problems
Composition

Models are being composed to solve new problems

Puppy Detector → Yes → Cuteness Detector → Cute!
Ball Detector → Yes → }
Composition

Models are being composed to solve new problems

Puppy Detector → Yes

Wrong but helpful...

Ball Detector → Yes

Cuteness Detector → Cute!

Still Correct
Composition

Models are being composed to solve new problems

Data Scientist -> Puppy Detector -> Yes
Ball Detector -> Yes
Cuteness Detector -> Wrong but helpful...
Still Correct
Cute!
Composition

Models are being composed to solve new problems
Models are being composed to solve new problems.

Need to track composition and validate **end-to-end accuracy**.

Need **unit** and **integration** testing for models.
Active Research in the for Model Development and Training

A an open source context management service that spans multiple data systems
http://www.ground-context.org/

An experiment management designed to track data, code, and people and address reproducibility
https://github.com/ucbrise/flor
Goal: make predictions in ~10ms under heavy load

Complicated by Deep Neural Networks ➔ New ML Algorithms and Systems
Inference Technologies

Prediction Service

Logic

Query

Prediction

End User Application

Feedback

Data Engineer

PredictionIO

mxnet

TensorFlow

Clipper
Inference Technologies

Prediction Service

Query

Prediction

End User Application

Our system

Feedback

Logic

Data Engineer

Specialized in Particular Models or Frameworks

mxnet

PredictionIO

TensorFlow
Deploying Interactive Machine Learning Applications with Clipper

Joseph E. Gonzalez
Co-director of the RISE Lab
jegonzal@cs.berkeley.edu
The remainder of this talk …

- **Challenges** of prediction serving
- **Clipper architecture** overview
- **Open-source** system effort
Prediction-Serving Challenges

Support low-latency, high-throughput serving workloads

Large and growing ecosystem of ML models and frameworks
Models getting more complex

- 10s of GFLOPs [1]

Deployed on critical path

- Maintain SLOs under heavy load

Using specialized hardware for predictions

Google Translate
Serving

140 billion words a day¹

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

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,zhilengc,qv1,mnorouzi@google.com
Wolfgang Macleroy, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideko 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

Designed New Hardware!
Tensor Processing Unit (TPU)

¹ https://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html
Building Application Specific Systems
Building Application Specific Systems

Problems:
- Expensive to build and maintain
- Require ML and systems expertise
- Tightly-coupled model and application
- Difficult to change or update application
- Only supports single ML framework
Growing ecosystem of ML Frameworks

- Fraud Detection
- Content Rec.
- Personal Asst.
- Robotic Control
- Machine Translation

Frameworks:
- Theano
- Apache Spark
- Caffe
- TensorFlow
- MXNet
- scikit-learn
- VW
- KALDI
Building & maintaining separate serving systems for each framework is expensive!

Solution

Pre-materialize predictions into a low latency Data Management System
Pre-materialized Predictions

Training
- Training Pipelines
- Trained Models
- Validation
- Feedback
- Data Engineer

Inference
- Prediction Service
- Logic
- Query
- Prediction
- End User Application
- Data Engineer
Pre-materialized Predictions

Training

Training Pipelines → Trained Models → Batch Training Framework

Live Data → Validation → All Possible Queries
Pre-materialized Predictions

Training Pipelines

Trained Models

Live Data

Training

Validation

All Possible Queries

Batch Training Framework

Apache Spark

Data Management System

(Scoring)

X
Y

MySQL
Pre-materialized Predictions

Training Pipelines → Trained Models → Batch Training Framework ➔ Data Management System

Standard Data Eng. Tools

Training

Validation

Live Data

Trained Models

Batch Training Framework

All Possible Queries

(X Y)

Data Management System

MySQL®
Serving **Pre-materialized** Predictions

Data Management System

Batch Training Framework

All Possible Queries

(Scoring)

Application

Low-Latency Serving
Serving **Pre-materialized** Predictions

**Problems:**
- Requires full set of *queries ahead of time*
- Small and **bounded input domain**
- Requires substantial **computation** and **space**
- Example: scoring all content for all customers!
- Costly update → rescore everything!
Wide range of **application and frameworks**
Middle layer for prediction serving.

Common Abstraction

System Optimizations

theano  scikit  Caffe  TensorFlow  PyTorch  VW

Spark  dmlc  mxnet  KALDI
Clipper Decouples Applications and Models

Applications

Predict

RPC/REST Interface

Observe

Clipper

RPC

Model Container (MC)

RPC

MC

Caffe

 RPC

MC

Scikit-learn
Clipper

- **Core system:** 10K lines of C++ and 8K lines of Python
- Designed to support production level query traffic
- Deliver low + predictable latency
- Research goal: *study reality …*
Run alongside other applications with **Kubernetes**
Getting Started with Clipper


Docker images available **on DockerHub**

Clipper admin is distributed as **pip package**: `pip install clipper_admin`

Get up and running **without compiling**
Clipper Design Innovations

**Containerized frameworks:** unified abstraction and framework level isolation and scaling

**Cross-framework caching and batching:** optimize throughput and latency

**Cross-framework model composition:** improved accuracy through ensembles and bandits
Clipper Architecture

Applications

Predict \(\uparrow\)

RPC/REST Interface

\(\uparrow\) Feedback

Clipper

RPC

Model Container (MC)

RPC

MC

RPC

Caffe

RPC

MC

RPC

MC
Common Interface → Simplifies Deployment:

- Evaluate models using original code & systems
Container-based Model Deployment

Implement Model API:

```python
class ModelContainer:
    def __init__(self, model_data):
    def predict_batch(inputs)
```

- API support for many programming languages
  - Python
  - Java
  - C/C++
  - R
Model Container (MC)

class ModelContainer:
    def __init__(model_data)
    def predict_batch(inputs)
Common Interface ➔ Simplifies Deployment:

- Evaluate models using original code & systems
- Models run in separate processes as Docker containers
  - Resource isolation: ML frameworks can be buggy
Common Interface → Simplifies Deployment:

- Evaluate models using original code & systems
- Models run in separate processes as Docker containers
  - Resource isolation: ML frameworks can be buggy
  - Scale-out at the level of individual models
Clipper Architecture

Applications

Predict  

RPC/REST Interface

Observe

Clipper

RPC

Model Container (MC)

Caffe

MC

MC

MC

RPC
Clipper Architecture

Applications

Predict $\uparrow$

RPC/REST Interface

Observe $\uparrow$

Clipper

Combine predictions across frameworks

Model Selection Layer

Provide a common interface and system optimizations

Model Abstraction Layer

RPC $\uparrow$

Model Container (MC)

RPC $\uparrow$

Caffe

RPC $\uparrow$

scikit
Clipper Architecture

Optimized Batching

Caching

Common API

Model Isolation

Provide a common interface and system optimizations

Model Abstraction Layer
**Batching** to Improve Throughput

---

**Why batching helps:**

A single page load may generate many queries.

---

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

---

Throughput-optimized frameworks

---

![Graph showing the relationship between batch size and throughput](chart.png)
Latency-aware Batching to Improve Throughput

Why batching helps:

- A single page load may generate many queries.

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)
Tensor Flow Conv. Net (GPU)

Throughput (Queries Per Second)

Better
Tensor Flow Conv. Net (GPU)

Throughput (Queries Per Second)

Latency (ms)

Better

Batch Size

Deadline

Better

73
Tensor Flow Conv. Net (GPU)

Throughput (Queries Per Second)

Latency (ms)

Optimal Batch Size

Deadline

P99
Latency-aware **Batching** to Improve Throughput

**Better**

**Throughput (QPS)**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Adaptive</th>
<th>No Batch</th>
<th>No Op</th>
<th>Random Forest (SKlearn)</th>
<th>Linear SVM (PySpark)</th>
<th>Linear SVM (SKlearn)</th>
<th>Kernel SVM (SKLearn)</th>
<th>Log Regression (SKLearn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No-Op</td>
<td>48386</td>
<td>8963</td>
<td>317</td>
<td>22859</td>
<td>29350</td>
<td>1920</td>
<td>1921</td>
<td>197203</td>
</tr>
<tr>
<td>Batching</td>
<td>48934</td>
<td>47219</td>
<td>317</td>
<td>22859</td>
<td>29350</td>
<td>1920</td>
<td>1921</td>
<td>197203</td>
</tr>
</tbody>
</table>
Clipper Architecture

Applications

Predict

RPC/REST Interface

Observe

Clipper

Combine predictions across frameworks

Model Selection Layer

RPC

Model Container (MC)

Provide a common interface and system optimizations

Model Abstraction Layer
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

Model Container (MC)

RPC

MC

RPC

MC

RPC

MC
Clipper

Combine predictions across frameworks

Model Selection Layer

Periodic retraining

Model Abstraction Layer

Model Container (MC)

RPC/REST Interface

Model Selection Layer

Provide a common interface and system optimizations

Model Selection Layer

Combine predictions across frameworks

Caffe

Version 1

Version 2

Version 3

Periodic retraining

Experiment with new models and frameworks

Caffe

TensorFlow

Caffe

Caffe
Selection Policy can Calibrate Confidence
Selection Policy: Estimate confidence

Better

Top-5 Error Rate

ImageNet

ensemble: 0.0586
4-agree: 0.0469
5-agree: 0.0327

0.0
0.2
0.4

confident
unsure

0.3182
0.1983
Selection Policy: Estimate confidence

Better

Top-5 Error Rate

ImageNet

- ensemble: 0.0586
- 4-agree: 0.0469
- 5-agree: 0.0327

Width is percentage of query workloads
Project Status and Development

- Current development focus:
  - **stability** and **performance** improvements
  - **easy model deployment** for common ML frameworks: Pytorch, caffe2 (via onnx), tensorflow, xgboost, mxnet, pyspark, scikit learn
  - **metrics** and **monitoring** infrastructure using Prometheus

- **Development Team:** *22 active contributors*
  - Including 8 from outside Berkeley

- **Working with several organizations on production deployments**
  - SAP, Scotia Bank, Pacific AI, ARM...
I made a case for **Model** & Data Engineering and outlined some of the **opportunities** & **challenges**
Middle layer for prediction serving.

Common Abstraction

System Optimizations

Open-source prediction serving system for low-latency, high-throughput predictions across machine learning models and frameworks.

http://clipper.ai
Thank you!

jegonzal@berkeley.edu

Collaborators

Daniel Crankshaw  Rolando Garcia  Joe Hellerstein  Yika Luo  Simon Mo  Vikram Sreekanti  Ion Stoica  Alexey Tumanov  Xin Wang  Neeraja Yadwadkar  Corey Zumar

Research Sponsors

[Logos of various companies including Alibaba, Amazon, CapitalOne, ERICSSON, Facebook, Google, Huawei, Intel, Microsoft, Scotiabank, Splunk, VMware, and the U.S. government.]
Bonus!
A few of the RISE Lab projects ...

**Real-time Inference**

**IDK Predictions Cascades**
Teaching AI to think fast by learning not to overthinking

**Hardware Security** for Apache Spark

**Opaque**
- SQL
- ML
- Graph
- query optimization
- o-filter
- o-groupby
- o-join

**Catalyst**

**Spark Execution**

**SkipNets: RL for Dynamic Network Design**

**AutoPyLot**
An open platform for **Autonomous Vehicles**

**RAY**
Parallel Python for Reinforcement Learning
Ray Tune
RL and Deep Learning workloads demand different resource requirements.
New Algorithms for hyperparameter tuning require more complicated control flows

Population Based Training

HyperBand
Security
Machine Learning is on the critical path

Augmented Reality
Mobile Assistants
Home Security
Self Driving Cars
Home Automation
Personal Robotics
Machine Learning in **Sensitive Contexts**

AR/VR Systems  |  Home Monitoring  |  Voice Technologies  |  Medical Imaging

Protect the **data**, the **model**, and the **query**
Protect the **data**, the **model**, and the **query**

High-Value **Data is Sensitive**
- Medical Info.
- Home video
- Finance

**Models** capture **value** in data
- Core Asset
- “Contain” the data

**Queries** can be as sensitive as the data
Model costs are increasing much faster than gains in accuracy.

Small but significant order of magnitude gap.
Dynamic Networks
Learning Not to Overthink

IDK Prediction Cascades

Simple Model

Accurate Model

Query

Fast

I don’t Know

Slow

Prediction

Prediction Cascades

SkipNets: gated execution

Easy Images
Skip Many Layers

Hard Images
Skip Few Layers
Learn to combine fast (inaccurate) models with slow (accurate) models to maximize accuracy while reducing computational costs.
SkipNet: dynamic execution within a model
SkipNet: dynamic execution within a model
**SkipNet**: dynamic execution within a model

- Combine **reinforcement learning** with **supervised pre-training** to learn a gating policy.
SkipNet Performance

Easy Images
Skip Many Layers

Hard Images
Skip Few Layers

SVHN

FLOPs (1e8)

ResNet-152
ResNet-110
ResNet-74

No Gate
SkipNet+SP
SkipNet+SP+HRL

65% 86% 51% 81% 34% 77%