How can **machine learning** techniques be used to address **systems** challenges?

How can **systems** techniques be used to address **machine learning** challenges?
How can **machine learning** techniques be used to address **systems** challenges?

Learning **Systems**

How can **systems** techniques be used to address **machine learning** challenges?
How can **machine learning** techniques be used **to address systems challenges**?

Systems are getting increasing complex:

- Resource Disaggregation ➔ growing diversity of system configurations and freedom to add resources as needed

- New Pricing Models ➔ dynamic pricing and potential to bid for different types of resources

- Data-centric Workloads ➔ performance depends on interaction between system, algorithms, and data
What vm-type should I use to run my experiment?
Paris
Performance Aware
Runtime Inference System

What vm-type should I use to run my experiment?

AWS
54 Instance Types
Paris
Performance Aware Runtime Inference System

What vm-type should I use to run my experiment?

- **AWS**
  - 54

- **Azure**
  - 25

- **Google**
  - 18

**Answer:** workload specific and depends on **cost** & **runtime** goals
Best vm-type depends on workload as well as cost & runtime goals

Which VM will cost me the least?

m1.small is cheapest?
Paris
Performance Aware Runtime Inference System

- Best vm-type depends on workload as well as cost & runtime goals

 Requires accurate runtime prediction.
Paris
Performance Aware Runtime Inference System

Goal: Predict the runtime of workload $w$ on VM type $v$

Challenge: How do we model workloads and VM types

Insight:
- Extensive benchmarking to model relationships between VM types
  - Costly but run once for all workloads
- Lightweight workload “fingerprinting” by on a small set of test VMs
- Generalize workload performance on other VMs

Results: Runtime prediction 17% Relative RMSE (56% Baseline)
Hemingway*
Modeling Throughput and Convergence for ML Workloads

- What is the best algorithm and level of parallelism for an ML task?
- **Trade-off:** Parallelism, Coordination, & Convergence
- **Research challenge:** Can we model this trade-off explicitly?

We can estimate $I$ from data on many systems
We can estimate $L$ from data for our problem

$I(p)$ Iterations per second as a function of cores $p$

$L(i, p)$ Loss as a function of iterations $i$ and cores $p$

*follow-up work to Shivaram’s Ernest paper
Hemingway*  
Modeling Throughput and Convergence for ML Workloads

- What is the best algorithm and level of parallelism for an ML task?
- **Trade-off:** Parallelism, Coordination, & Convergence
- **Research challenge:** Can we model this trade-off explicitly?

\[
L(i, p) \quad \text{Loss as a function of iterations } i \text{ and cores } p \\
I(p) \quad \text{Iterations per second as a function of cores } p
\]

\[
\text{loss}(t, p) = L(t \ast I(p), p)
\]

- How long does it take to get to a given loss?
- Given a time budget and number of cores, which algorithm will give the best result?

*follow-up work to Shivaram’s Ernest paper*
Deep Code Completion
Neural architectures for reasoning about programs

- **Goals:**
  - Smart naming of variables and routines
  - Learn coding styles and patterns
  - Predict large code fragments

- **Char and Symbol LSTMs**

- Programs are more tree shaped...

```python
def fib(x):
    if x < 2:
        return x
    else:
        y = fib(x-1) + fib(x-2)
        return y
```
Deep Code Completion
Neural architectures for reasoning about programs

- **Goals:**
  - Smart naming of variables and routines
  - Learn coding styles and patterns
  - Predict large code fragments

- Char and Symbol LSTMs
  - Programs are more tree shaped…

```python
def fib( ):
    if x < 2:
        return x
    y = fib(x-2) + fib(x-1)
    return y
```
Deep Code Completion
Neural architectures for reasoning about programs

- **Goals:**
  - Smart naming of variables and routines
  - Learn coding styles and patterns
  - Predict large code fragments

- **Char and Symbol LSTMs**

- **Exploring Tree LSTMs**
  - Issue: dependencies flow in both directions

```python
def fib(x):
    if x < 2:
        return x
    return y + fib(x-2) + fib(x-1)
```

Parse Tree

Xin Wang
Chang Liu
Dawn Song

Deep Code Completion
Neural architectures for reasoning about computer programs

- **Goals:**
  - Smart naming of variables and routines
  - Learn coding styles and patterns
  - Predict large code fragments
- Current studying Char-LSTM and Tree-LSTM on benchmark C++ code and JavaScript code.
- Plan to extend Tree-LSTM with downward information flow.
For now, the neural network can learn some code patterns like matching the parenthesis, if-else block, etc but the variable name issue still hasn’t been solved.

*this is trained on the leetcode OJ code submissions from Github.
How can **machine learning** techniques be used to **address systems challenges**?

How can **systems** techniques be used to **address machine learning challenges**?
How can **machine learning** techniques be used to address systems challenges?

How can **systems** techniques be used to address **machine learning** challenges?
Systems for Machine Learning

**Timescale:** minutes to days

**Systems:** offline and batch optimized

*Heavily studied ... primary focus of the ML research*
Big Data

Training

Big Model

MLC
TensorFlow
Caffe
GraphLab
Spark

MLbase
KeystoneML
GraphX
Splash
CoCoA

Please make a Logo!
Please make a Logo!
Temgine
A Scalable Multivariate Time Series Analysis Engine

Challenge:
- Estimate second order statistics
  - E.g. Auto-correlation, auto-regressive models, …
- for high-dimensional & irregularly sampled time series

Regularly Sampled
Samples are easy to align (requires sorting)

Irregularly Sampled
Difficult to align!
Temgine
A Scalable Multivariate Time Series Analysis Engine

Challenge:
- Estimate second order statistics
  - E.g. Auto-correlation, auto-regressive models, …
- for high-dimensional & irregularly sampled time series

Irregularly Sampled
Difficult to align!

Solution:
- Project onto Fourier basis
  - does not require data alignment
- Infer statistics in frequency domain
  - equivalent to kernel smoothing
  - analysis of bias – variance tradeoff
Temgine
A Scalable Multivariate Time Series Analysis Engine

Challenge:
- Estimate second order statistics
  - E.g. Auto-correlation, auto-regressive models, …
- for high-dimensional & irregularly sampled time series

Solution:
- Project onto Fourier basis
  - does not require data alignment
- Infer statistics in frequency domain
  - equivalent to kernel smoothing
  - analysis of bias – variance tradeoff

Define an operator DAG (like TF) and then rely on query-optimization to define efficient execution.
Learning

Big Data

Training

Big Model
Learning

Big Data → Training

Big Model

Inference

Query

Decision

Application

Timescale: ~10 milliseconds
Systems: online and latency optimized
Less Studied …
why is **Inference** challenging?

Need to render **low latency** (< 10ms) predictions for **complex** Models under **heavy load** with system **failures**.

**Queries**

```
SELECT * FROM users JOIN items, click_logs, pages WHERE ...
```
**Claim:**
next big area of research in scalable ML systems

**Timescale:** \(~10\) milliseconds

**Systems:** *online* and *latency* optimized

*Less studied …*
Big Data → Training → Big Model → Feedback → Application

Learning

Inference

Query → Decision
Big Data Training Application

Learning

Inference

Timescale: hours to weeks

Issues: No standard solutions … implicit feedback, sample bias, …

Feedback
Why is **Feedback** challenging?

- Exposes system to **feedback loops**
  - Address Explore – Exploit trade-off in real-time

- Adverserial feedback
  - Opportunities for **multi-task learning** and **anomaly detection**

- Need to address **temporal variation**
  - Need to model time directly? When do we forget the past?
Learning

Big Data → Training → Big Model

Inference

Big Model → Query → Decision → Application

Feedback
Big Data

Big Model

Training

Application

Decision

Query

Learning

Responsive (~10ms)

Adaptive (~1 seconds)

Feedback

Responsive (~10ms)
Techniques we are studying (or *should be* ...):

- Multi-task Learning
- Adaptive Batching
- Approx. Caching
- Anytime Inference
- Model Switching
- Meta-Policy RL
- Online Ensemble Learning
- Load Shedding
- Model Compression
- Inference on the Edge
Prediction Serving

Clipper

Daniel Crankshaw  Xin Wang  Giulio Zhou  Michael Franklin  Ion Stoica
Learning

Big Data

Training

Inference

Query

Decision

Application

Feedback
Big Data Training Application

Learning

- Slow Changing Parameters
- Fast Changing Parameters
- Feedback

Inference

- Query
- Decision
- Application

Feedback

Slow Changing Parameters

Fast Changing Parameters
Hybrid Offline + Online Learning

Update feature functions **offline** using batch solvers
- Leverage high-throughput systems (Tensor Flow)
- Exploit slow change in population statistics

\[ f(x; \theta)^T w_u \]

Update the user weights **online**:
- Simple to train + more robust model
- Address rapidly changing user statistics
Common modeling structure

$$f(x; \theta)^T w_u$$
Clipper Online Learning for Recommendations (Simulated News Rec.)

Partial Updates: 0.4 ms
Retraining: 7.1 seconds

>4 orders-of-magnitude faster adaptation
Learning

Big Data

Fast Changing Parameters

Caffe

TensorFlow

Slow Changing Parameters

Inference

Clipper

Fast Changing Parameters

Application

Feedback

Slow

Fast Feedback
Clipper Serves Predictions across ML Frameworks

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

Clipper

Frameworks:
- theano
- Dato
- Create
- Caffe
- TensorFlow
- scikit
- learn
- VMware
- Theano ML
- Keystone ML
- Medium
- VW
- KALDI
- Intel
Clipper Architecture

Applications

Clipped

Predict

RPC/REST Interface

Observe

Clipper

RPC

Model Wrapper (MW)

Keystone

RPC

MW

Caffe

RPC

MW

RPC

MW

RPC

MW

...
Clipper Architecture

Applications

Predict

RPC/REST Interface

Observe

Clipper

Model Selection Layer

Model Abstraction Layer

Provide a common interface to models while bounding latency and maximizing throughput.

Improve accuracy through ensembles, online learning and personalization.

RPC

Model Wrapper (MW)

Keystone

Caffe

MW

RPC

RPC

RPC
Clipper Architecture

Applications

- Predict
- Observe

RPC/REST Interface

- Anytime Predictions
- Model Selection Layer
- Approximate Caching
- Model Abstraction Layer
- Adaptive Batching

Model Wrapper (MW)

- Caffe
- scikit-learn
- TensorFlow
- Keystone
- MI
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:

- be as slow as allowed...
  - Application specifies latency objective
  - Clipper uses TCP-like tuning algorithm to increase latency up to the objective
Tensor Flow Conv. Net (GPU)

- **Latency (ms)**
- **Throughput (Queries Per Second)**
- **Optimal Batch Size**

![Graph showing latency and throughput with optimal batch size highlighted.](image-url)
**Approximate Caching to Reduce Latency**

- Opportunity for caching
  - Popular items may be evaluated frequently

- Need for approximation
  - Bag-of-Words Model
  - Images
  - High Dimensional and continuous valued queries have low cache hit rate.

**Clipper Solution: Approximate Caching**

apply locality sensitive hash functions
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:
- be as slow as allowed...
  - Application specifies latency objective
  - Clipper uses TCP-like tuning algorithm to increase latency up to the objective
Throughput (Queries Per Second)

Latency (ms)

Batch Sizes (Queries)

Optimal Batch Size

Tensor Flow Conv. Net (GPU)

Optimal Batch Size

Throughput

Latency

Deadline

P99 Latency

Avg. Latency

Throughput (Queries Per Second)
Anytime Predictions

Slow Changing Model

Fast Changing Linear Model

Solution:
Replace missing prediction with an estimator

20ms

Clipper

Application

\[ E[ \hat{y}(x) ] \]
Anytime Predictions

\[ W_{\text{scikit}} f_{\text{scikit}}(x) + W_{\text{TF}} E_X [f_{\text{TF}}(X)] + W_{\text{Caffe}} f_{\text{Caffe}}(x) \]
Comparison to TensorFlow Serving

Takeaway: Clipper is able to match the average latency of TensorFlow Serving while reducing tail latency (2x) and improving throughput (2x)
Takeaway: Clipper is able to *gracefully degrade accuracy* to maintain availability under heavy load.
## Improved Prediction Accuracy (ImageNet)

<table>
<thead>
<tr>
<th>System</th>
<th>Model</th>
<th>Error Rate</th>
<th>#Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caffe</td>
<td>VGG</td>
<td>13.05%</td>
<td>6525</td>
</tr>
<tr>
<td>Caffe</td>
<td>LeNet</td>
<td>11.52%</td>
<td>5760</td>
</tr>
<tr>
<td>Caffe</td>
<td>ResNet</td>
<td>9.02%</td>
<td>4512</td>
</tr>
<tr>
<td>TensorFlow</td>
<td>Inception v3</td>
<td>6.18%</td>
<td>3088</td>
</tr>
</tbody>
</table>

sequence of pre-trained models
## Improved Prediction Accuracy (ImageNet)

<table>
<thead>
<tr>
<th>System</th>
<th>Model</th>
<th>Error Rate</th>
<th>#Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caffe</td>
<td>VGG</td>
<td>13.05%</td>
<td>6525</td>
</tr>
<tr>
<td>Caffe</td>
<td>LeNet</td>
<td>11.52%</td>
<td>5760</td>
</tr>
<tr>
<td>Caffe</td>
<td>ResNet</td>
<td>9.02%</td>
<td>4512</td>
</tr>
<tr>
<td>TensorFlow</td>
<td>Inception v3</td>
<td>6.18%</td>
<td>3088</td>
</tr>
<tr>
<td>Clipper</td>
<td>Ensemble</td>
<td>5.86%</td>
<td>2930</td>
</tr>
</tbody>
</table>

5.2% relative improvement in prediction accuracy!
Clipper prediction serving system that spans multiple ML Frameworks and is designed to

- simplify model serving
- bound latency and increase throughput
- and enable real-time learning and personalization across machine learning frameworks
Learning Systems

Joseph E. Gonzalez
773 Soda Hall
jegonzal@cs.berkeley.edu

Graduate students collaborators on this work:

Francois Billetti  Daniel Crankshaw  Ankur Dave  Xinghao Pan  Xin Wang  Neeraja Yadwadkar  Wenting Zheng
Real-time, Intelligent, and Secure Systems Lab

Joseph E. Gonzalez  jegonzal@cs.berkeley.edu
Joseph Hellerstein  hellerstein@cs.berkeley.edu
Raluka Ada Popa  raluca.popa@cs.berkeley.edu
Ion Stoica  istoica@cs.berkeley.edu
RISE Lab
From live data to real-time decisions

AMP Lab
From batch data to advanced analytics
Goal

Real-time decisions

decide in ms

on live data

the current state of the environment

with strong security

privacy, confidentiality, integrity
Real-time, Intelligent, and Secure Systems Lab

Learn More:
• **CS294 Course** on RISE Topics  
  [https://ucbrise.github.io/cs294-rise-fa16/](https://ucbrise.github.io/cs294-rise-fa16/)
• Early RISErs **Seminar** on **Mondays** at **9:30 AM**
Security: Protecting Models

Data is a core asset & models capture the value in data

- **Expensive**: many engineering & compute hours to develop
- Models can **reveal private information** about the data

How do we **protect models** from being stolen?

- Prevent them from being copied from devices (**DRM**? **SGX**?)
- Defend against **active learning** attacks on decision boundaries

How do we identify when models have been stolen?

- **Watermarks** in decision boundaries?