

# The New DBfication of ML/AI

Arun Kumar



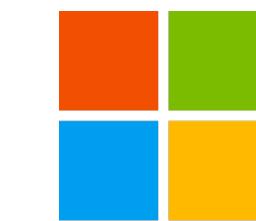
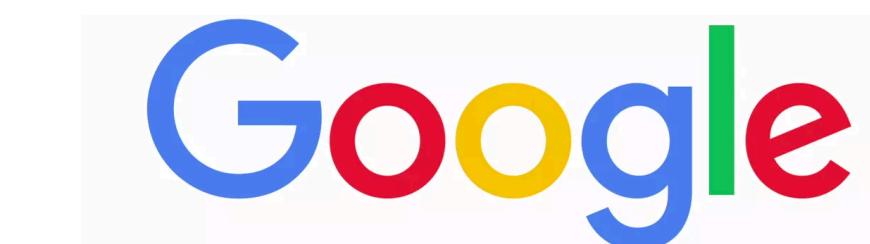
**UC San Diego**  
JACOBS SCHOOL OF ENGINEERING  
Computer Science and Engineering

**UC San Diego**  
HALİCİOĞLU DATA SCIENCE INSTITUTE

# Golden Age of ML/AI



FACEBOOK



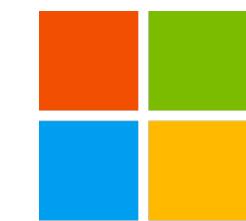
Microsoft

# Golden Age of ML/AI

**amazon**

**FACEBOOK**

**Google**



**Microsoft**

**MAYO CLINIC**

**Healthcare**

**AMERICAN FAMILY INSURANCE**

**Insurance**

**Walmart** A yellow five-pointed star logo.

**Retail**

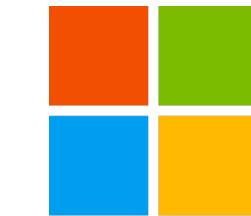


**Sciences**

# Golden Age of ML/AI



FACEBOOK



Microsoft



Healthcare



Insurance



Retail



Sciences

\$ 38 billion  
in 2019\*



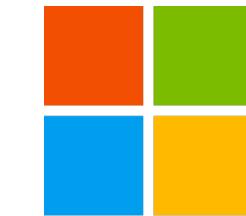
\$ 500 billion!  
by 2024\*

\*International Data Corporation

# Golden Age of ML/AI



FACEBOOK



Healthcare



Insurance



Retail



Sciences

\$ 38 billion  
in 2019\*



\$ 500 billion!  
by 2024\*

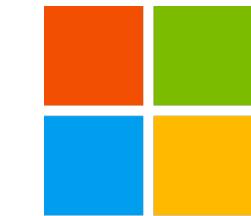


\*International Data Corporation

# Golden Age of ML/AI



FACEBOOK



Healthcare



Insurance



Retail



Sciences

\$ 38 billion  
in 2019\*



\$ 500 billion!  
by 2024\*

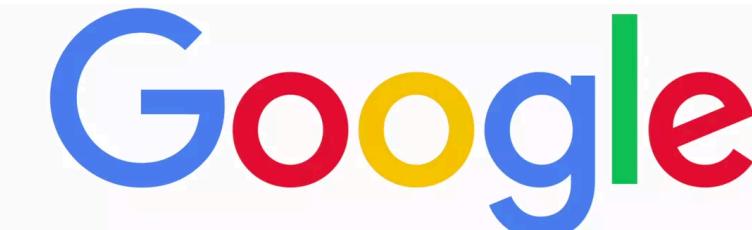


\*International Data Corporation

# Golden Age of ML/AI



FACEBOOK



Healthcare



Insurance



Retail



Sciences

\$ 38 billion  
in 2019\*



\$ 500 billion!  
by 2024\*

Still, fundamental efficiency and usability bottlenecks in the  
end-to-end process of building and deploying ML applications

\*International Data Corporation

# My Research

New abstractions, algorithms, and software systems  
to “*democratize*” ML/AI-based data analytics from  
a data management/systems standpoint

# My Research

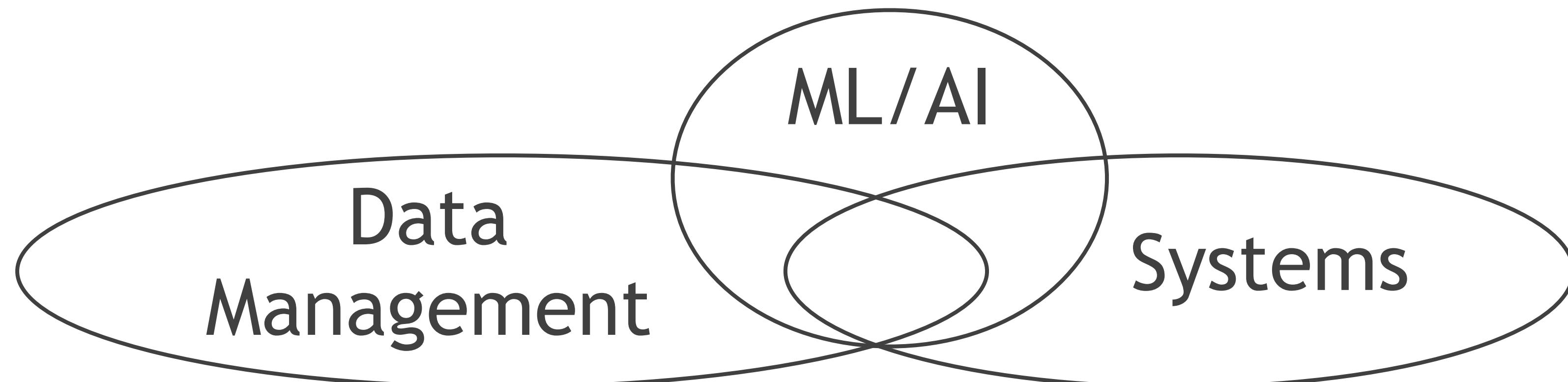
New abstractions, algorithms, and software systems  
to “*democratize*” ML/AI-based data analytics from  
a data management/systems standpoint

$$\text{Democratization} = \text{System Efficiency (Reduce costs)} + \text{Human Efficiency (Improve productivity)}$$

# My Research

New abstractions, algorithms, and software systems  
to “*democratize*” ML/AI-based data analytics from  
a data management/systems standpoint

$$\text{Democratization} = \text{System Efficiency} \quad (\text{Reduce costs}) + \text{Human Efficiency} \quad (\text{Improve productivity})$$



# My Research

New abstractions, algorithms, and software systems  
to “*democratize*” ML/AI-based data analytics from  
a data management/systems standpoint

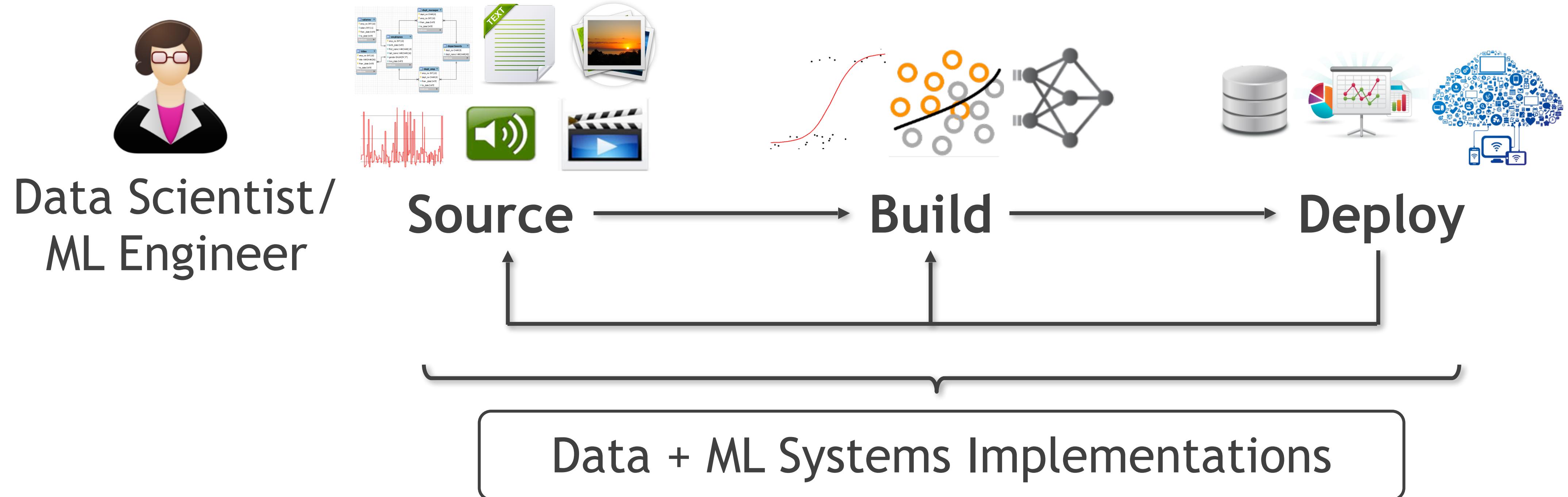
$$\text{Democratization} = \text{System Efficiency} \quad (\text{Reduce costs}) + \text{Human Efficiency} \quad (\text{Improve productivity})$$

Practical and scalable data systems for ML/AI analytics

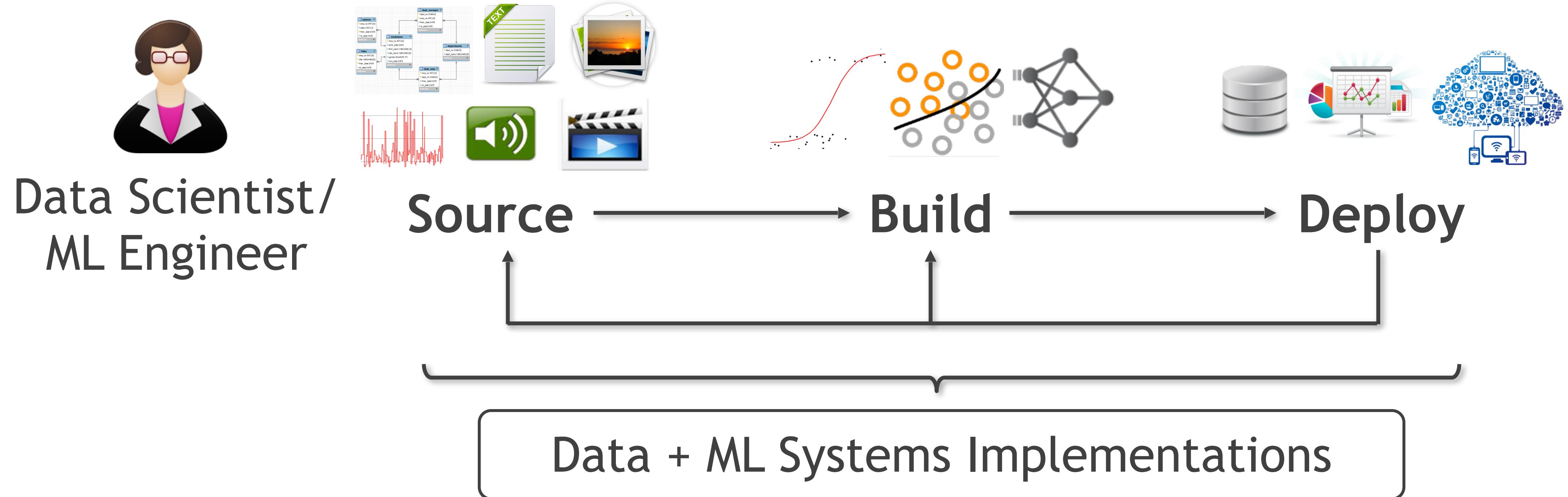
Inspired by *relational database systems* principles

Exploit insights from *learning theory* and *optimization theory*

# End-to-End ML Application Lifecycle



# End-to-End ML Application Lifecycle



**Research Approach :** *Abstract* key steps + *Formalize* computation + *Automate* grunt work + *Optimize* execution

# Outline

- | The New DBfication of ML/AI
- | Two Examples from My Research
- | Accelerating the DBfication of ML/AI

# DB-style Practical Concerns in ML

# DB-style Practical Concerns in ML

**Key concerns in ML:**

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

# DB-style Practical Concerns in ML

**Key concerns in ML:**

Accuracy

Runtime efficiency (sometimes)

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

*Q: What if the dataset is larger than single-node RAM?*

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

Scalability (and efficiency at scale)

*Q: What if the dataset is larger than single-node RAM?*

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

Scalability (and efficiency at scale)

*Q: How are the features and models configured?*

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

Scalability (and efficiency at scale)

Usability

*Q: How are the features and models configured?*

# DB-style Practical Concerns in ML

**Key concerns in ML:**

Accuracy

Runtime efficiency (sometimes)

**Additional key *practical* concerns in ML Systems:**

Scalability (and efficiency at scale)

Usability

*Q: How does it fit within production systems and workflows?*

# DB-style Practical Concerns in ML

**Key concerns in ML:**

Accuracy

Runtime efficiency (sometimes)

**Additional key *practical* concerns in ML Systems:**

Scalability (and efficiency at scale)

Usability

Manageability

*Q: How does it fit within production systems and workflows?*

# DB-style Practical Concerns in ML

**Key concerns in ML:**

Accuracy

Runtime efficiency (sometimes)

**Additional key *practical* concerns in ML Systems:**

Scalability (and efficiency at scale)

Usability

Manageability

*Q: How to simplify the implementation of such systems?*

# DB-style Practical Concerns in ML

**Key concerns in ML:**

Accuracy

Runtime efficiency (sometimes)

**Additional key *practical* concerns in ML Systems:**

Scalability (and efficiency at scale)

Usability

Manageability

Developability

*Q: How to simplify the implementation of such systems?*

# DB-style Practical Concerns in ML

**Key concerns in ML:**

Accuracy

Runtime efficiency (sometimes)

**Additional key *practical* concerns in ML Systems:**

Scalability (and efficiency at scale)

Usability

Manageability

Developability

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

Scalability (and efficiency at scale)

Usability

Manageability

Developability

*Long-standing  
concerns in the  
DB systems  
world!*

# DB-style Practical Concerns in ML

Key concerns in ML:

Accuracy

Runtime efficiency (sometimes)

Additional key *practical* concerns in ML Systems:

Scalability (and efficiency at scale)

Usability

Manageability

Developability

*Long-standing  
concerns in the  
DB systems  
world!*

*Can often trade off accuracy a bit to gain on the rest!*

# The Push to “Platformize” ML/AI

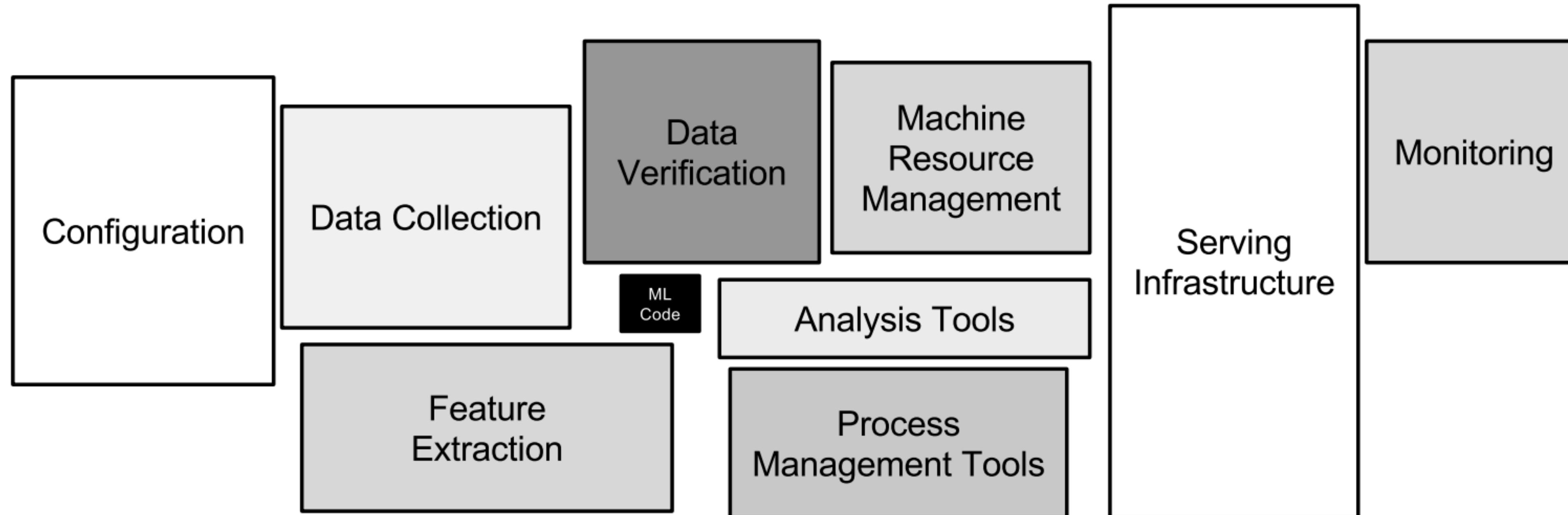


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Google

# A Brief History of “Platformizing” ML

# A Brief History of “Platformizing” ML

1980s



S

# A Brief History of “Platformizing” ML



Mid  
1990s

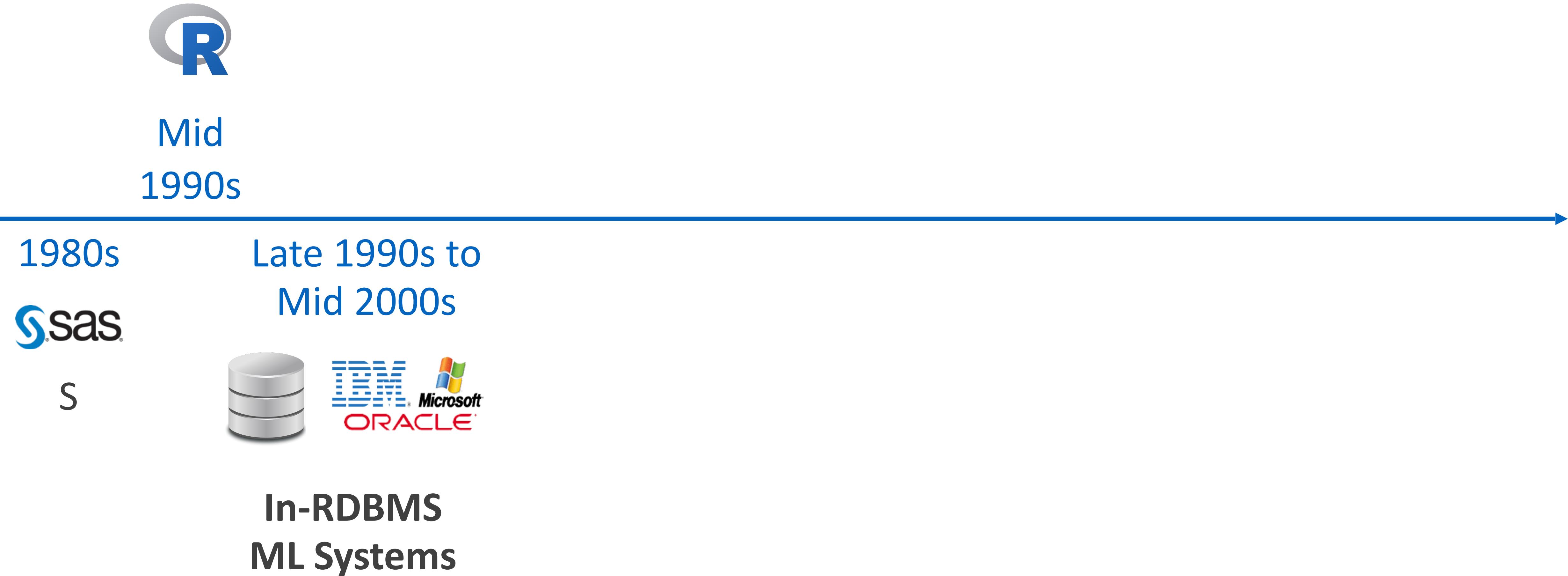
1980s

---

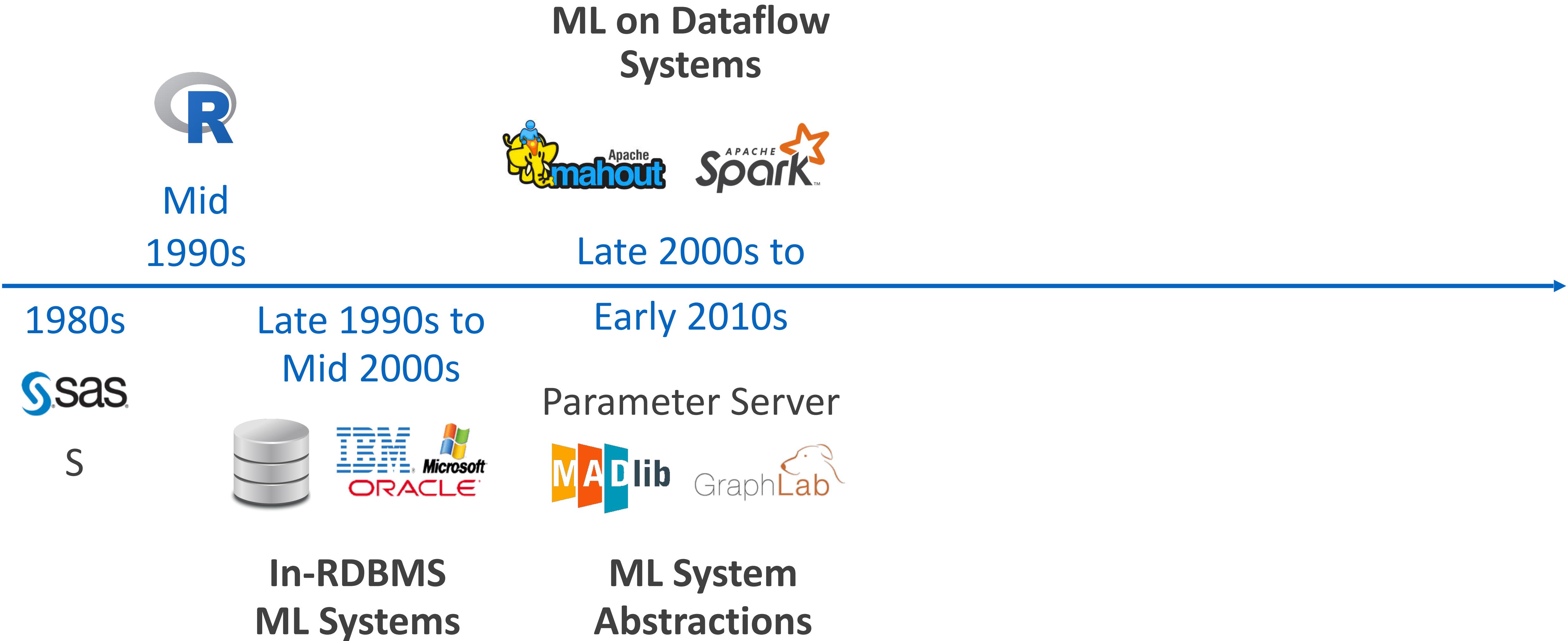


S

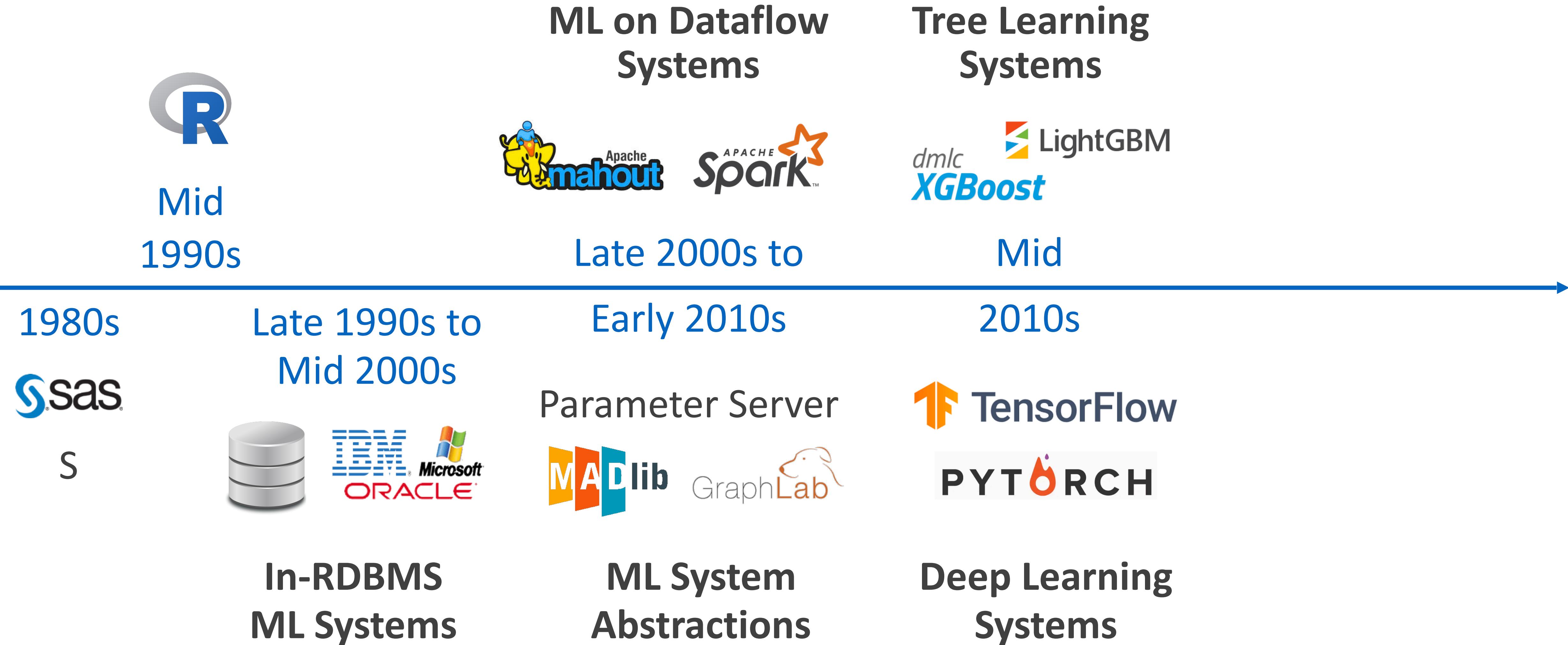
# A Brief History of “Platformizing” ML



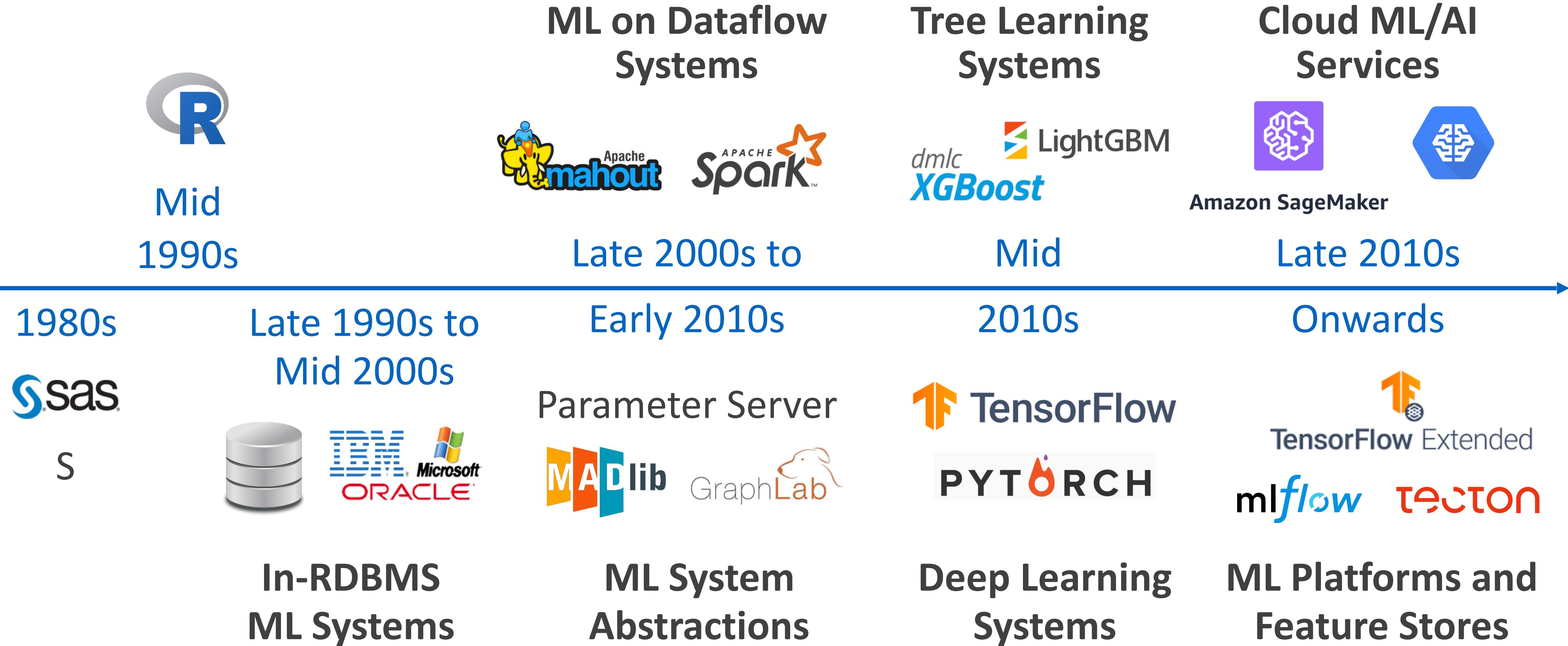
# A Brief History of “Platformizing” ML



# A Brief History of “Platformizing” ML

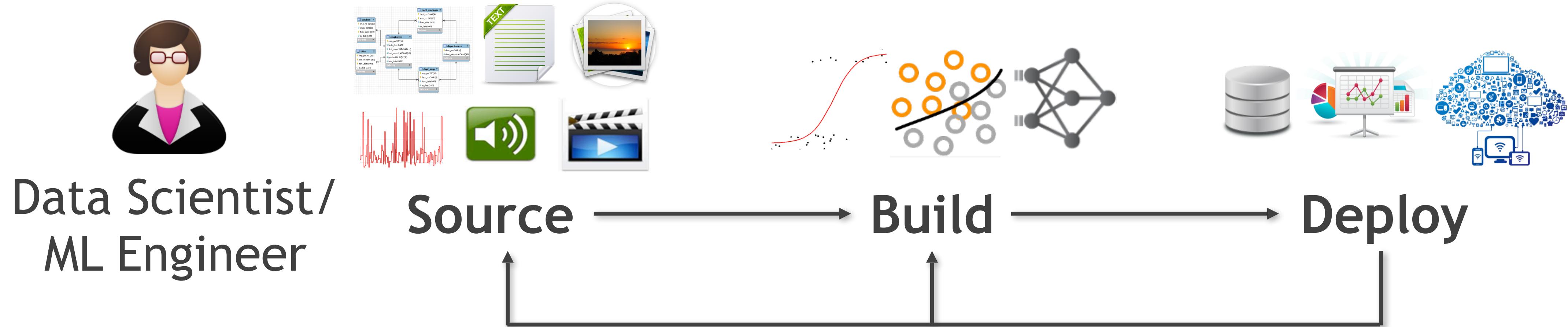


# A Brief History of “Platformizing” ML

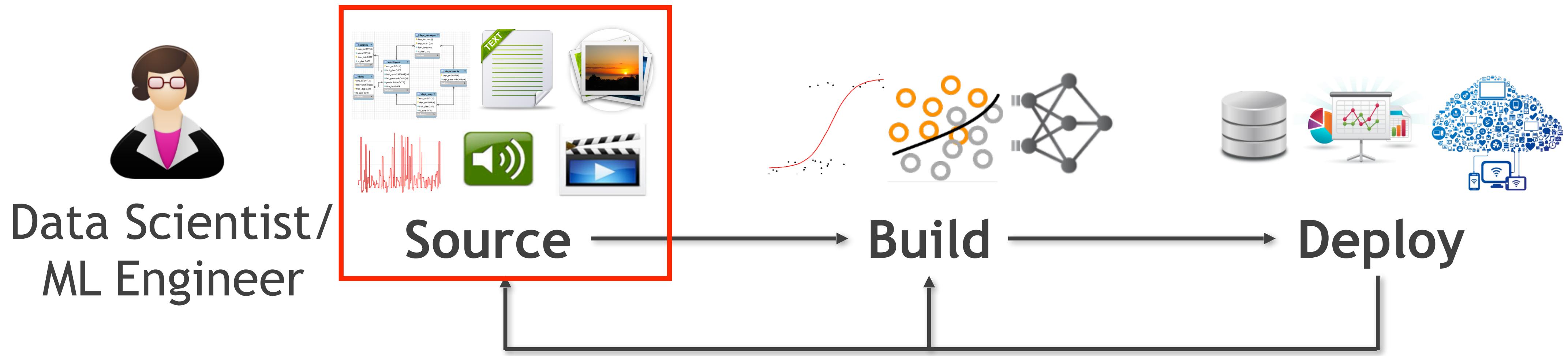


*The way I see it, the rise of ML systems/  
platforms today resembles the rise of RDBMSs  
circa early 1980s*

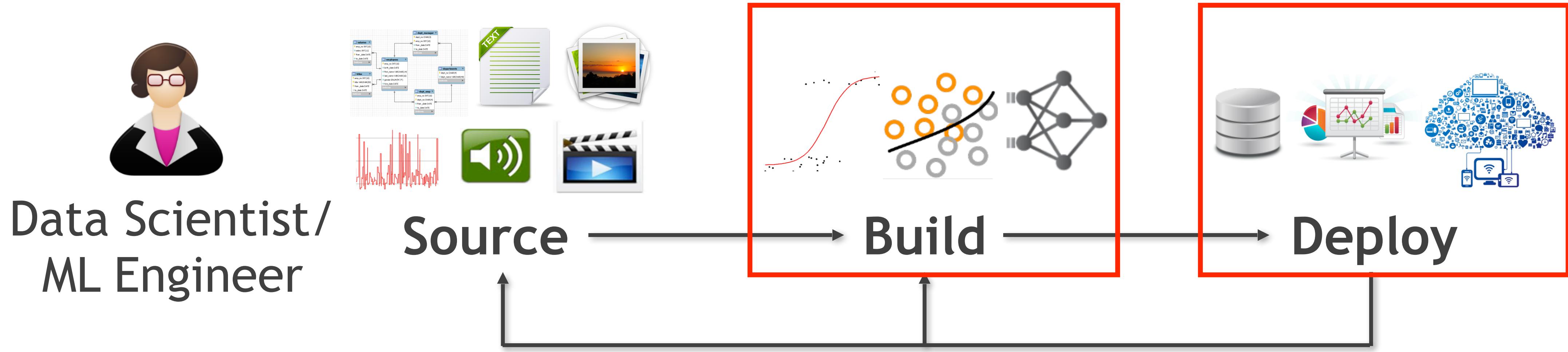
# The New DBfication of ML/AI



# The New DBfication of ML/AI



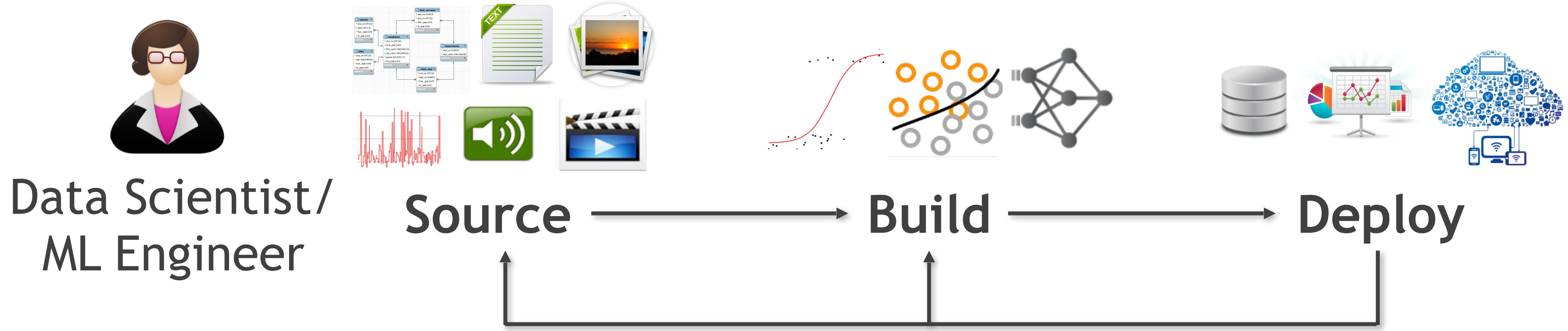
# The New DBfication of ML/AI



...

...

# The New DBfication of ML/AI



Metadata Management for ML  
Data Prep/Cleaning for ML  
Multimodal ML Query Models  
Data Search, Labeling, etc.

...

Benchmark Frameworks and Data  
Fairness, Transparency, Privacy, etc.

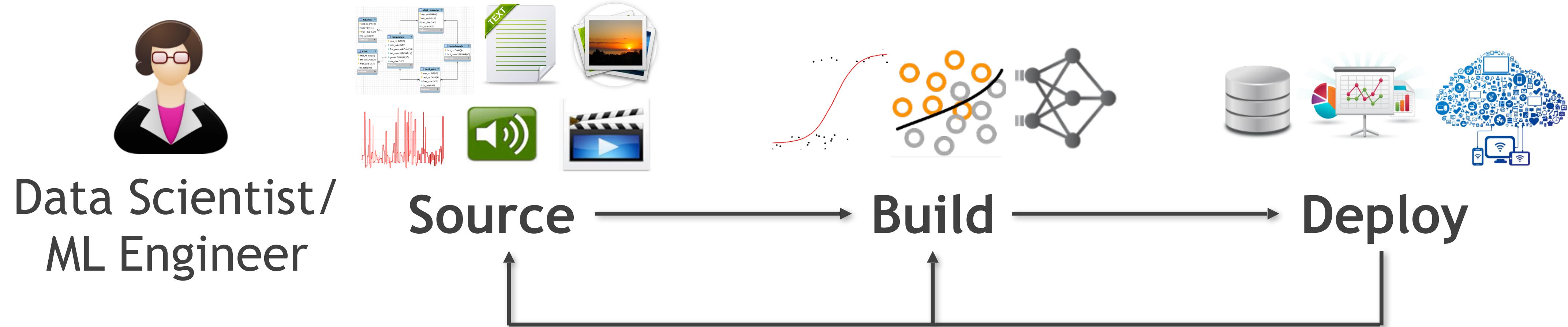
Scalable Data Systems for ML  
Query Optimization for ML  
Cloud and Streaming Infra.  
Provenance and Debugging

...

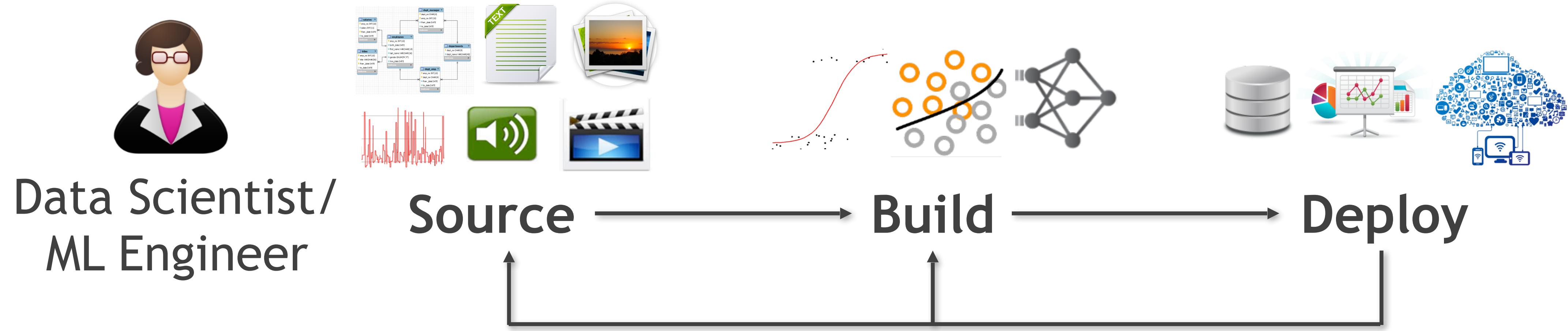
# Outline

- | The New DBfication of ML/AI
- | Two Examples from My Research
- | Accelerating the DBfication of ML/AI

# The New DBfication of ML/AI



# The New DBfication of ML/AI



Metadata Management for ML  
Data Prep/Cleaning for ML  
Multimodal ML Query Models  
Data Search, Labeling, etc.

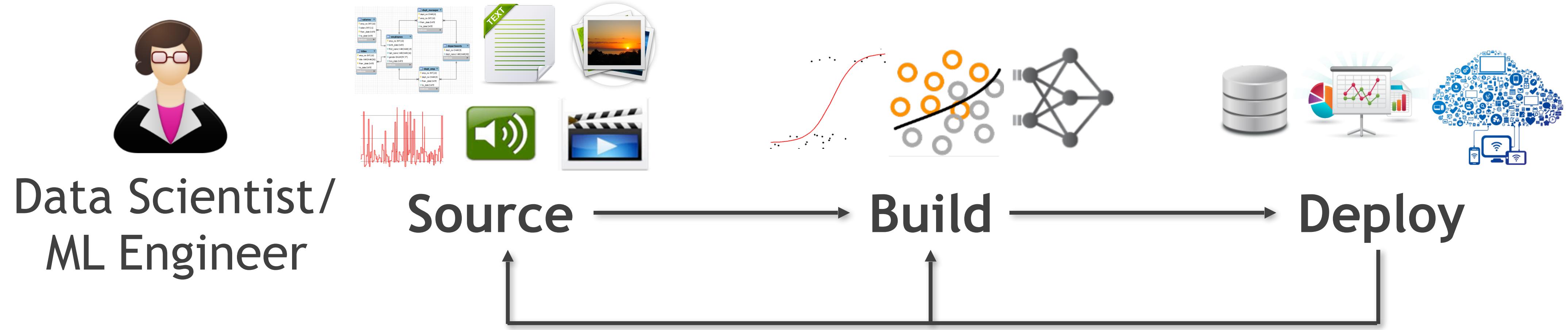
...

Scalable Data Systems for ML  
Query Optimization for ML  
Cloud and Streaming Infra.  
Provenance and Debugging

...

Benchmark Frameworks and Data  
Fairness, Transparency, Privacy, etc.

# The New DBfication of ML/AI



Metadata Management for ML  
Data Prep/Cleaning for ML  
Multimodal ML Query Models  
Data Search, Labeling, etc.

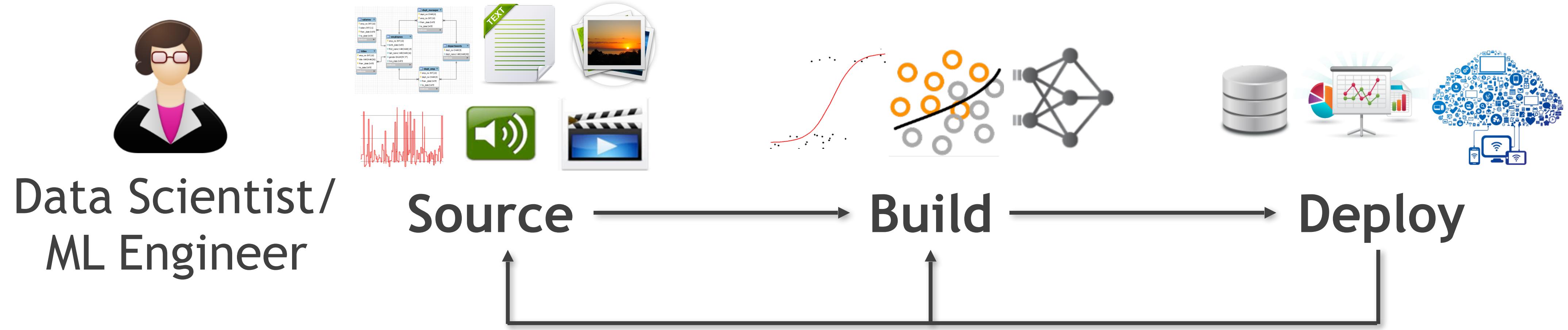
Scalable Data Systems for ML  
Query Optimization for ML  
Cloud and Streaming Infra.  
Provenance and Debugging

...

Benchmark Frameworks and Data  
Fairness, Transparency, Privacy, etc.

...

# The New DBfication of ML/AI



Metadata Management for ML  
Data Prep/Cleaning for ML  
Multimodal ML Query Models  
Data Search, Labeling, etc.

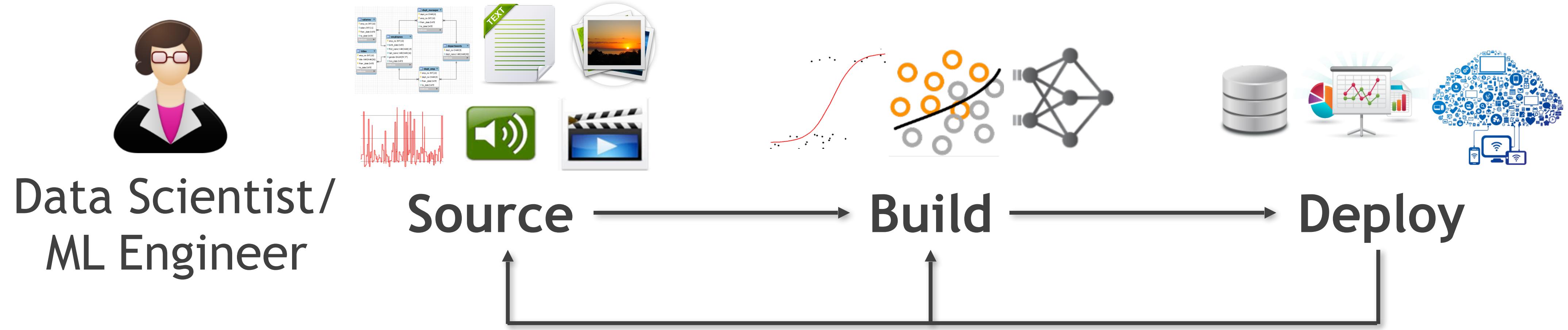
...

Scalable Data Systems for ML  
Query Optimization for ML  
Cloud and Streaming Infra.  
Provenance and Debugging

...

Benchmark Frameworks and Data  
Fairness, Transparency, Privacy, etc.

# The New DBfication of ML/AI



Metadata Management for ML  
Data Prep/Cleaning for ML  
Multimodal ML Query Models  
Data Search, Labeling, etc.

Scalable Data Systems for ML  
Query Optimization for ML  
Cloud and Streaming Infra.  
Provenance and Debugging

...

Benchmark Frameworks and Data  
Fairness, Transparency, Privacy, etc.

...

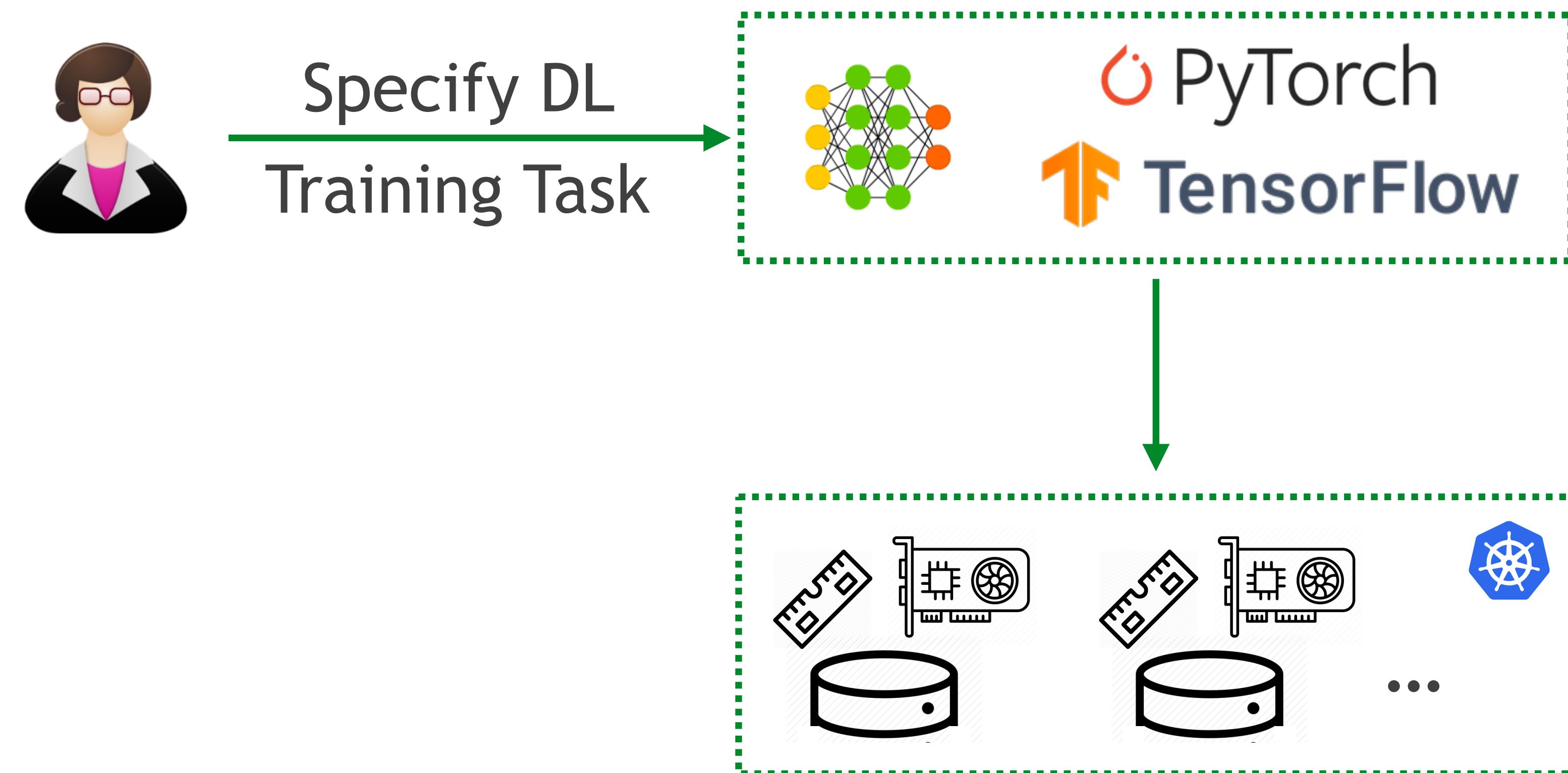
# Outline

- | The New DBfication of ML/AI
- | Two Examples from My Research
  - | Example 1: Scalable DL Systems
  - | Example 2: Auto Data Prep for ML
- | Accelerating the DBfication of ML/AI

# Outline

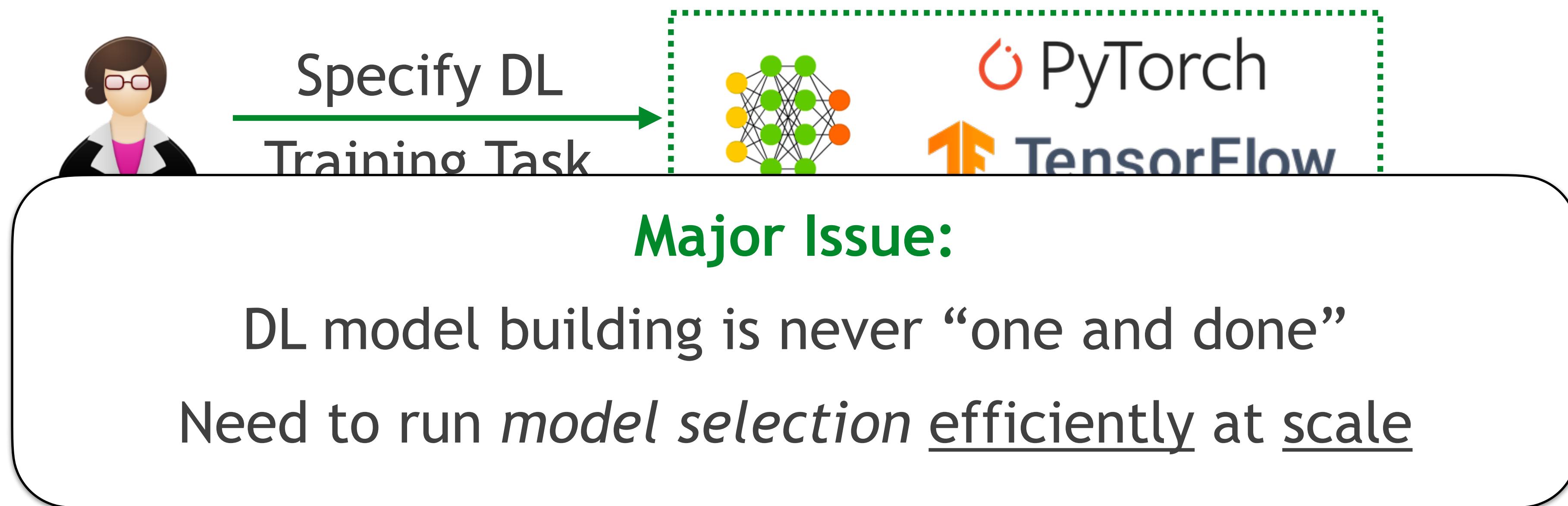
- | The New DBfication of ML/AI
- | Two Examples from My Research
  - | Example 1: Scalable DL Systems
  - | Example 2: Auto Data Prep for ML
- | Accelerating the DBfication of ML/AI

# Example 1: DL Systems at Scale



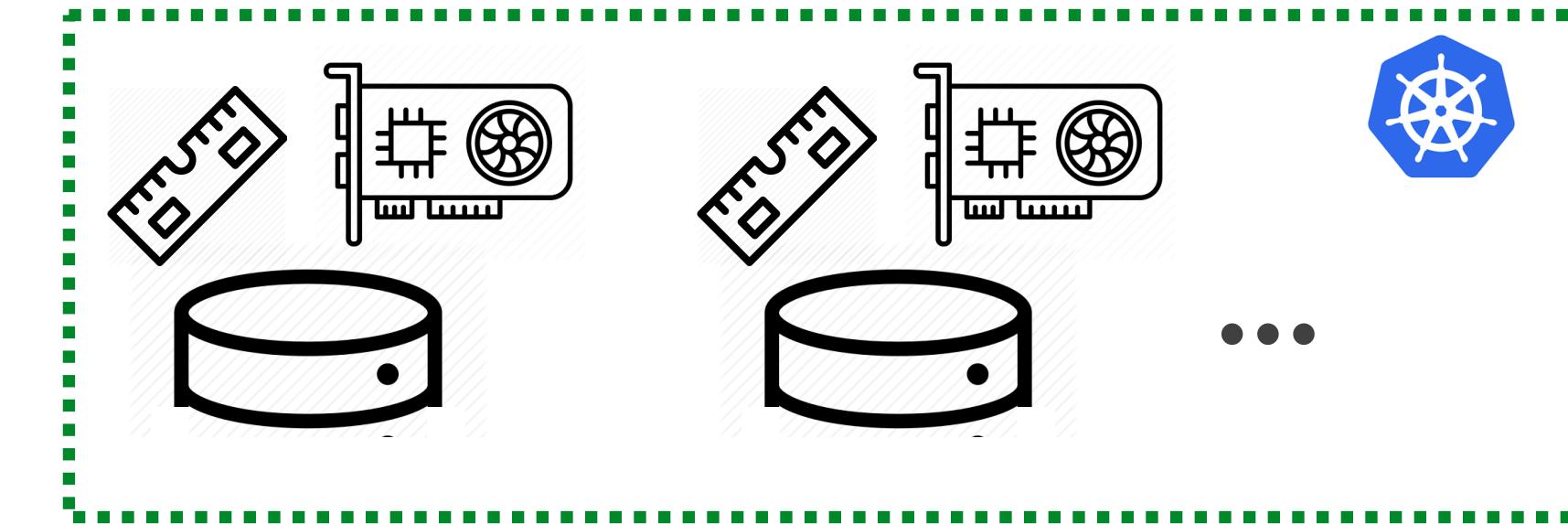
Cluster: GPU+CPU, memory, disks

# Example 1: DL Systems at Scale



Cluster: GPU+CPU, memory, disks

# Project Cerebro

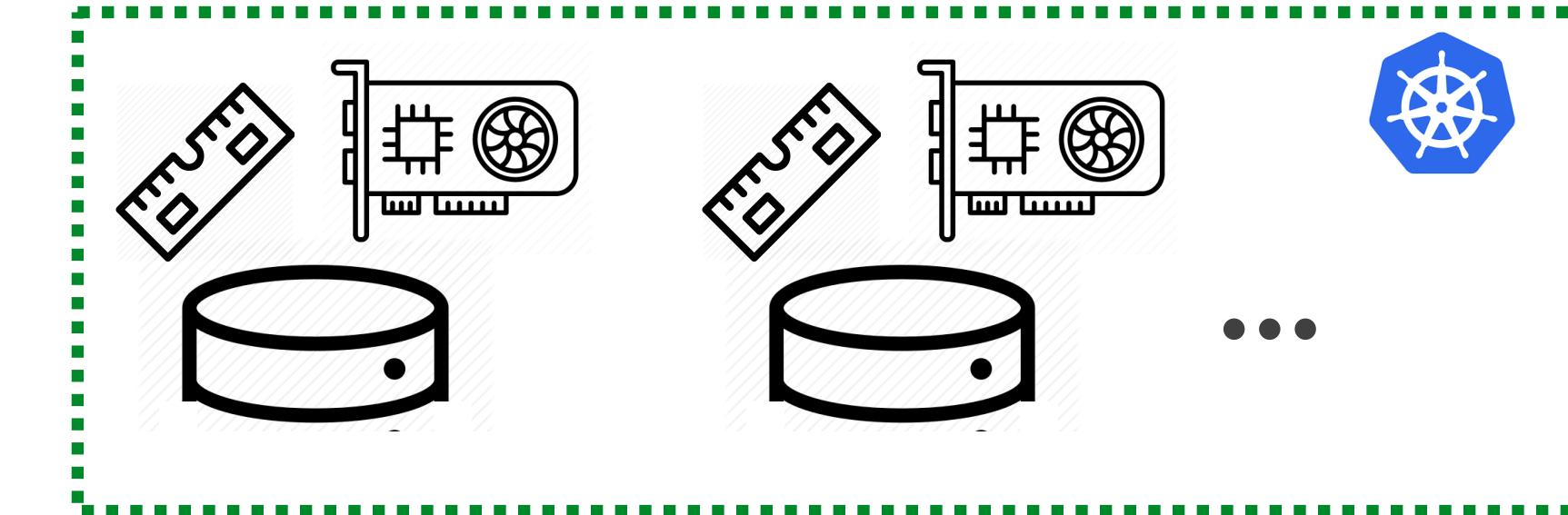


Plain/Kubernetes cluster

# Project Cerebro

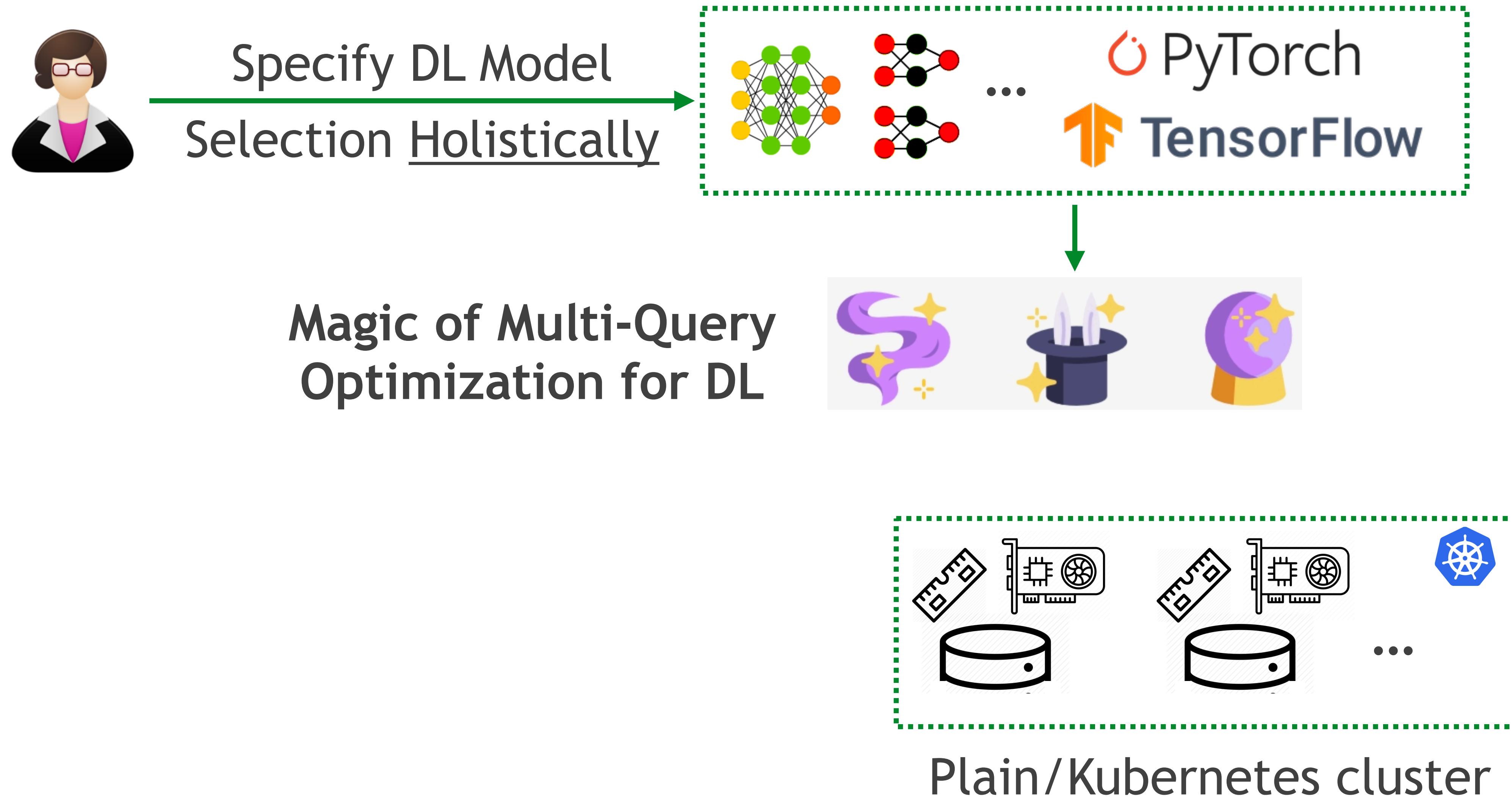


Specify DL Model  
Selection Holistically

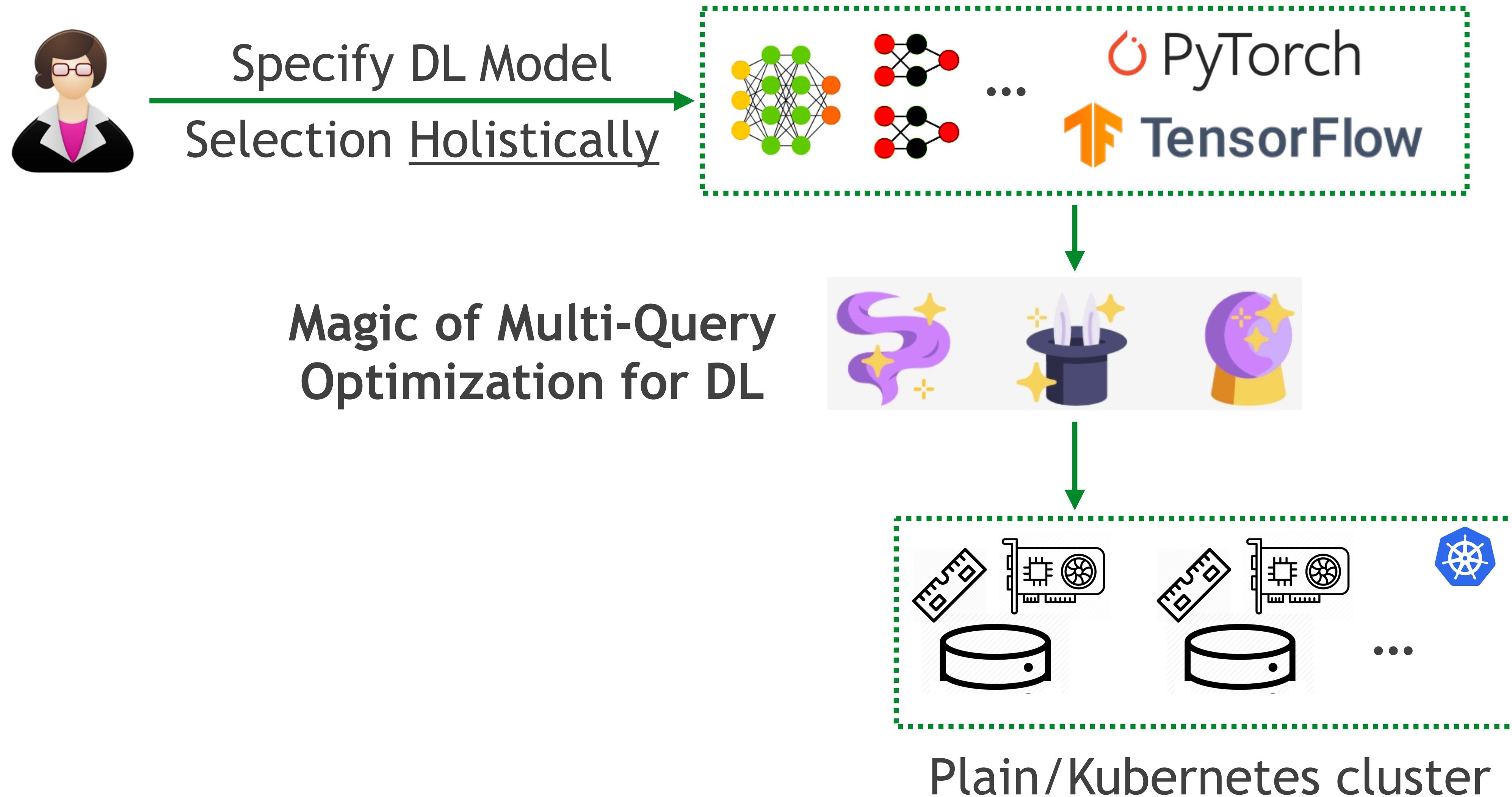


Plain/Kubernetes cluster

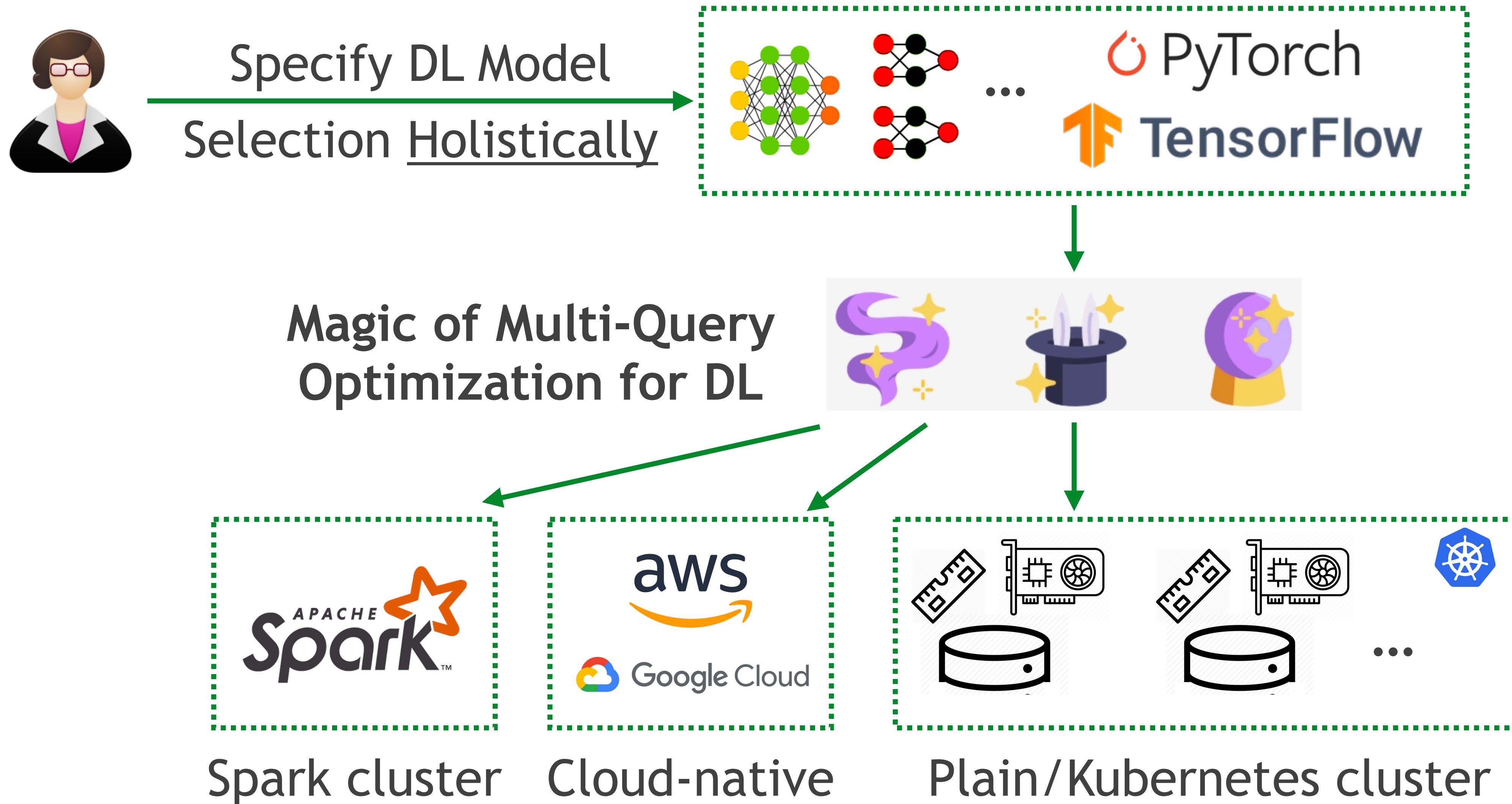
# Project Cerebro



# Project Cerebro



# Project Cerebro



*So, what kinds of scalability issues arise?  
What optimizations does Cerebro do?*

# DL Scalability Issues on Memory Hierarchy

GPU Memory

DRAM/Main Memory

Local Disk

Remote Disk(s)

# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

DRAM/Main Memory

Local Disk

Slowest data access  
Large; Cheaper

Remote Disk(s)



# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

DRAM/Main Memory

Local Disk

Slowest data access  
Large; Cheaper  
Multiplicity

Remote Disk(s)



# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

1. DL model and/or data batch  
does not fit in GPU memory!



Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)

# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

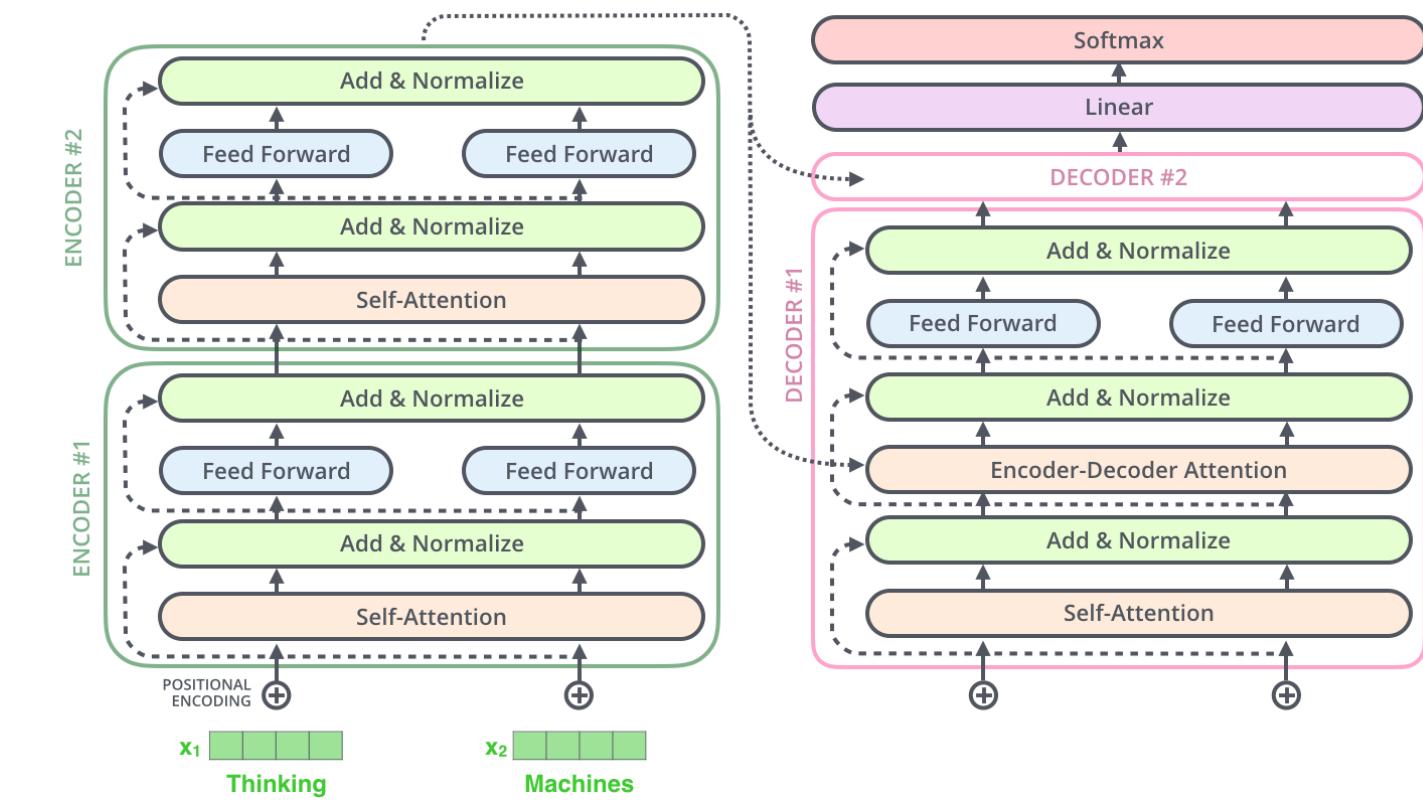
DRAM/Main Memory

Local Disk

Remote Disk(s)



1. DL model and/or data batch does not fit in GPU memory!



# What Cerebro Does (Under Submission)

Fastest data access  
Tiny; Costliest

GPU Memory

1. DL model and/or data batch  
does not fit in GPU memory!



DRAM/Main Memory

Slowest data access  
Large; Cheaper  
Multiplicity

Local Disk

Remote Disk(s)

# What Cerebro Does (Under Submission)

Fastest data access  
Tiny; Costliest

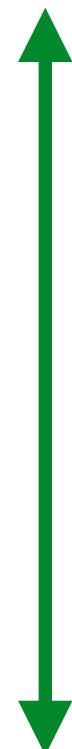
GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



1. DL model and/or data batch does not fit in GPU memory!
1. Automatic model *sharding* and *spilling*

# What Cerebro Does (Under Submission)

Fastest data access  
Tiny; Costliest

GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



1. DL model and/or data batch does not fit in GPU memory!

1. Automatic model *sharding* and *spilling*

We rewrite computational graph to break it up on DRAM

# What Cerebro Does (Under Submission)

Fastest data access  
Tiny; Costliest

GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



1. DL model and/or data batch does not fit in GPU memory!

1. Automatic model *sharding* and *spilling*

We rewrite computational graph to break it up on DRAM  
Interleave multiple model parts across GPUs to raise throughput

# What Cerebro Does (Under Submission)

Fastest data access  
Tiny; Costliest

GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



1. DL model and/or data batch does not fit in GPU memory!

1. Automatic model *sharding* and *spilling*

We rewrite computational graph to break it up on DRAM  
Interleave multiple model parts across GPUs to raise throughput  
Seamlessly scales to any size

# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

DRAM/Main Memory

Local Disk

Slowest data access  
Large; Cheaper  
Multiplicity

Remote Disk(s)



# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



2. Dataset does not fit in DRAM;  
GNNs do random I/Os to disk!

# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

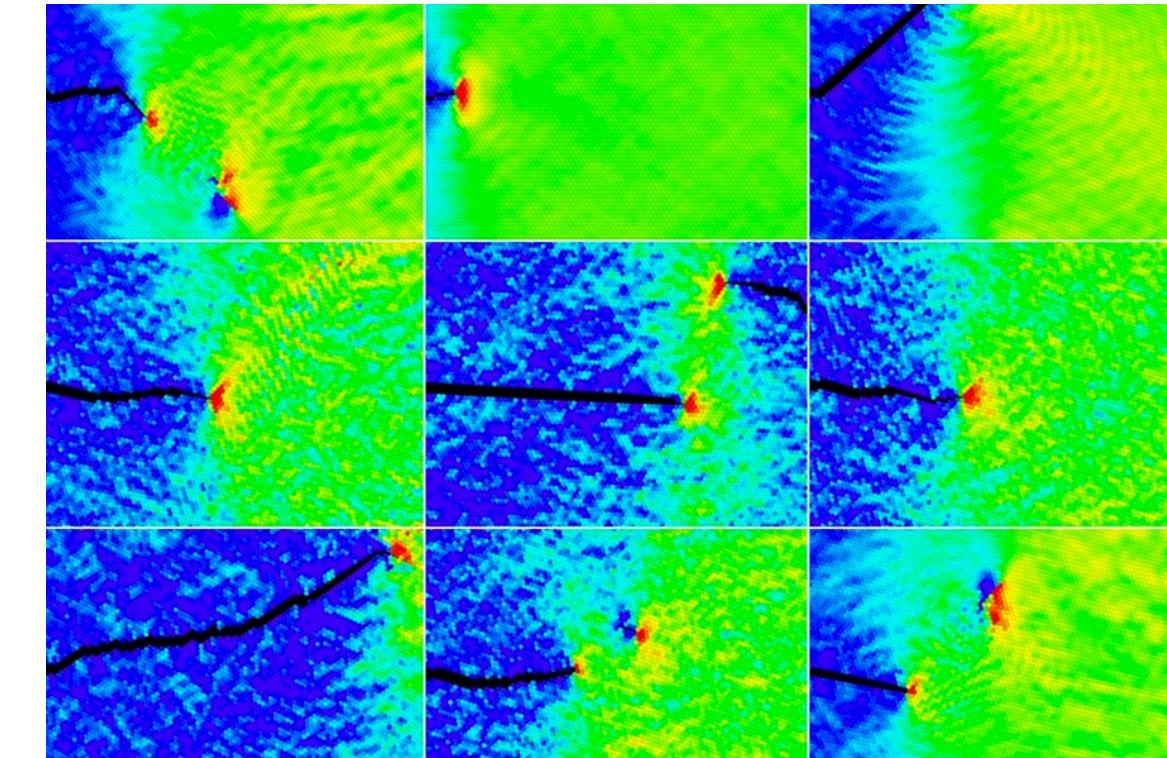
DRAM/Main Memory

Local Disk

Remote Disk(s)



2. Dataset does not fit in DRAM;  
GNNs do random I/Os to disk!



# What Cerebro Will Do (In the Works)

Fastest data access  
Tiny; Costliest

GPU Memory

DRAM/Main Memory

Local Disk

Remote Disk(s)

Slowest data access  
Large; Cheaper  
Multiplicity



2. Dataset does not fit in DRAM  
and random accesses to disk!

# What Cerebro Will Do (In the Works)

Fastest data access  
Tiny; Costliest

GPU Memory

DRAM/Main Memory

Local Disk

Remote Disk(s)

Slowest data access  
Large; Cheaper  
Multiplicity



2. Dataset does not fit in DRAM and random accesses to disk!
2. I/O optimization with staged reads and caching

# What Cerebro Will Do (In the Works)

Fastest data access  
Tiny; Costliest

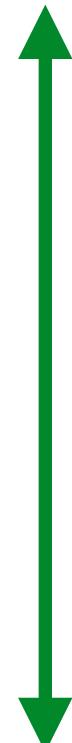
GPU Memory

Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



2. Dataset does not fit in DRAM and random accesses to disk!

2. I/O optimization with staged reads and caching

We rewrite DL computation to reduce random I/Os

Memory manager to cache data and reuse across many models

# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

DRAM/Main Memory

Local Disk

Slowest data access  
Large; Cheaper  
Multiplicity

Remote Disk(s)

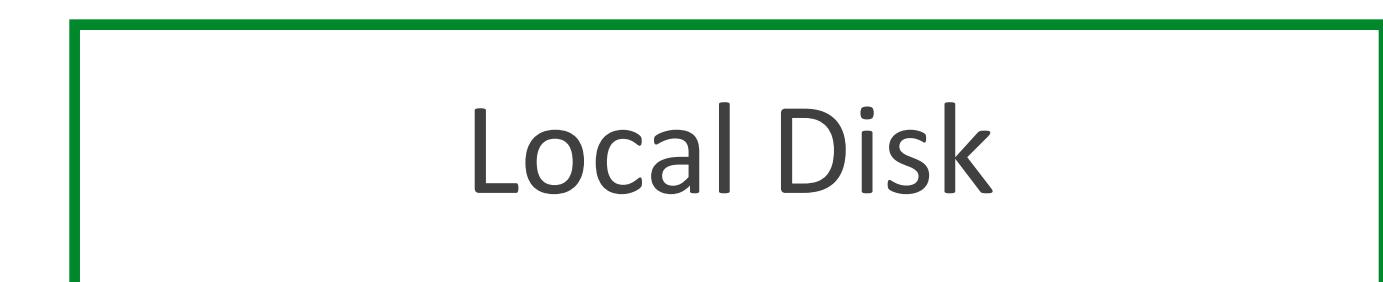


# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest



Slowest data access  
Large; Cheaper  
Multiplicity



3. Dataset is too big for one node to finish on time!  
Copying full data across all cluster nodes wastes DRAM/disk

# DL Scalability Issues on Memory Hierarchy

Fastest data access  
Tiny; Costliest

GPU Memory

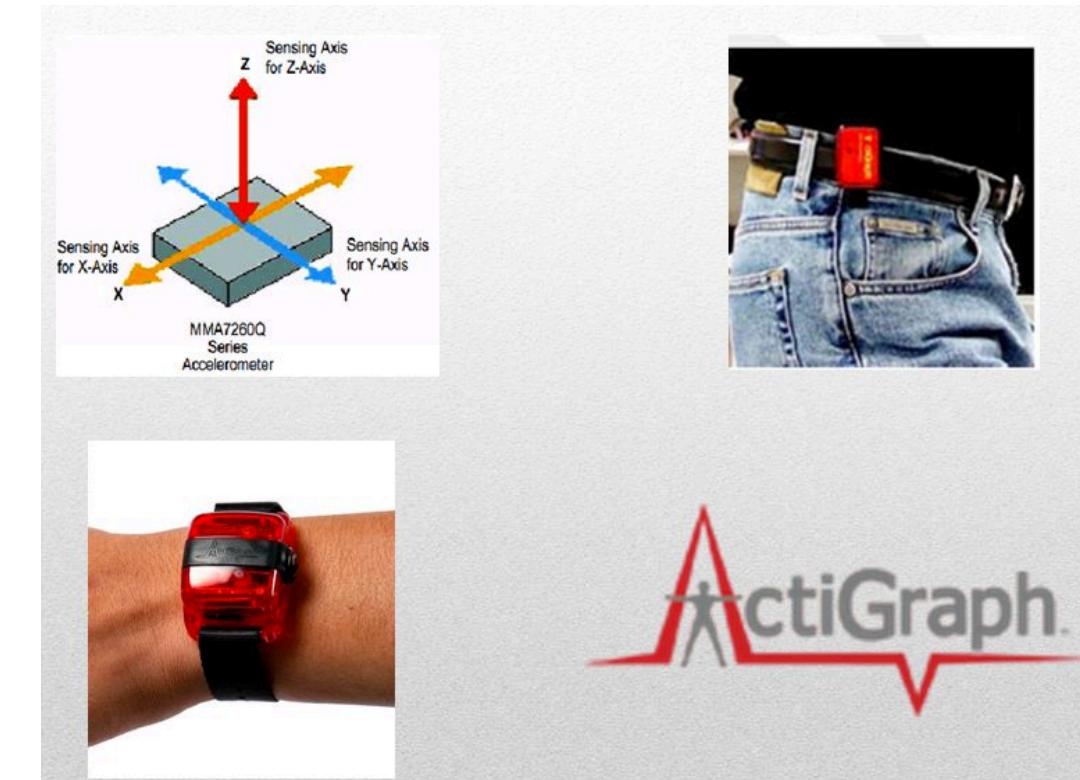


Slowest data access  
Large; Cheaper  
Multiplicity

DRAM/Main Memory

Local Disk

Remote Disk(s)



UC San Diego  
SCHOOL OF MEDICINE

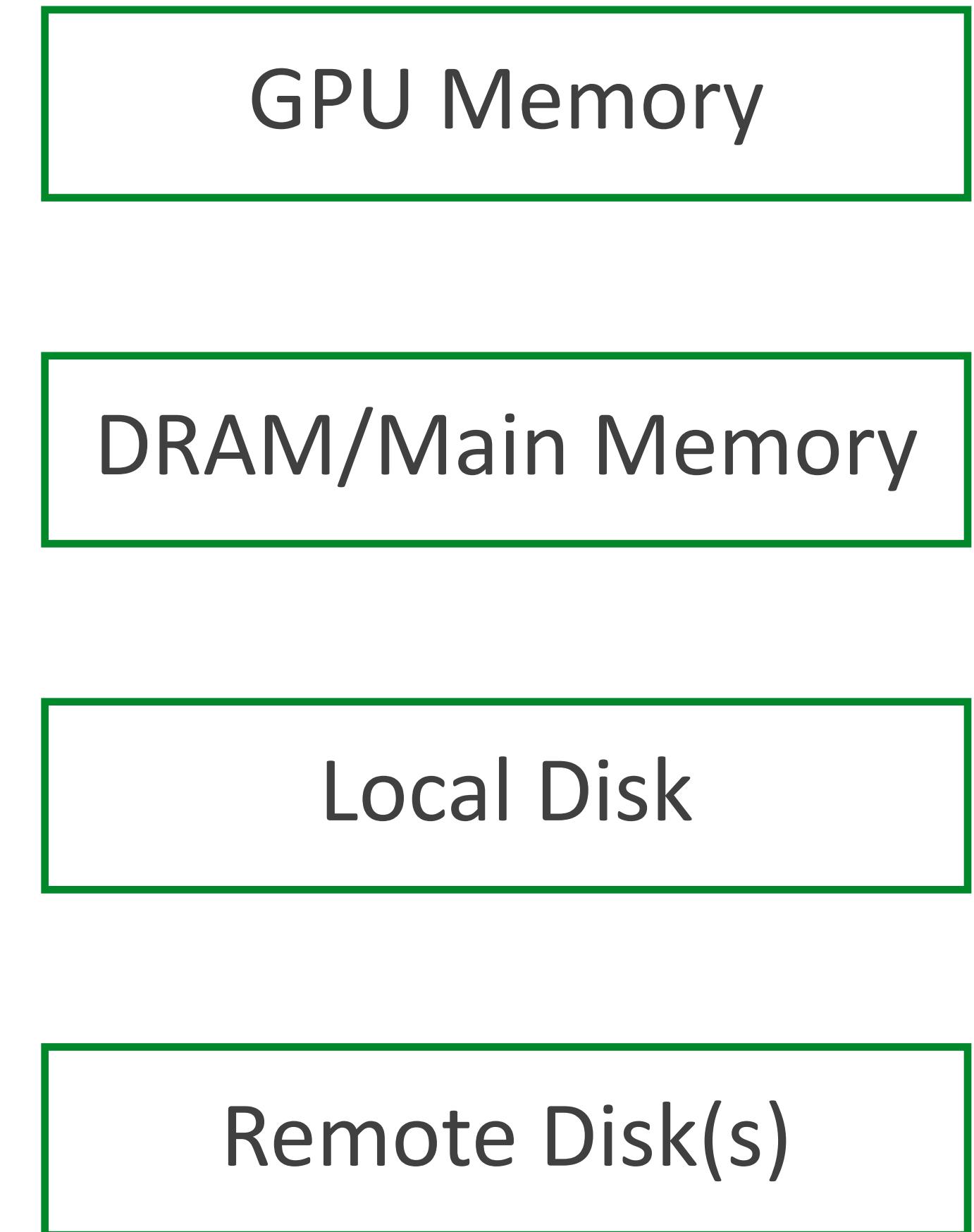
3. Dataset is too big for one node to finish on time!  
Copying full data across all cluster nodes wastes DRAM/disk

# What Cerebro Does (Already Published)

Fastest data access  
Tiny; Costliest



Slowest data access  
Large; Cheaper  
Multiplicity



3. Dataset is too big for one node to finish on time!  
Copying full data across all cluster nodes wastes DRAM/disk  
Copying wastes DRAM/disk

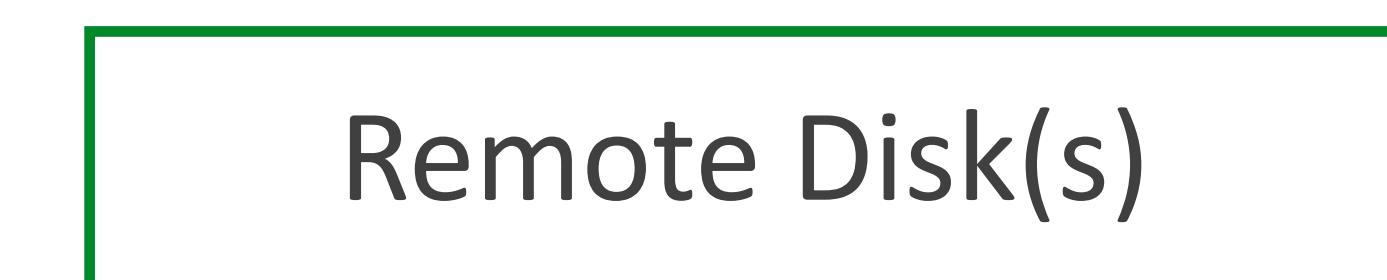
# What Cerebro Does (Already Published)

Fastest data access  
Tiny; Costliest



Slowest data access  
Large; Cheaper  
Multiplicity

A vertical double-headed arrow connects the "GPU Memory" box at the top to the "DRAM/Main Memory" box in the middle.



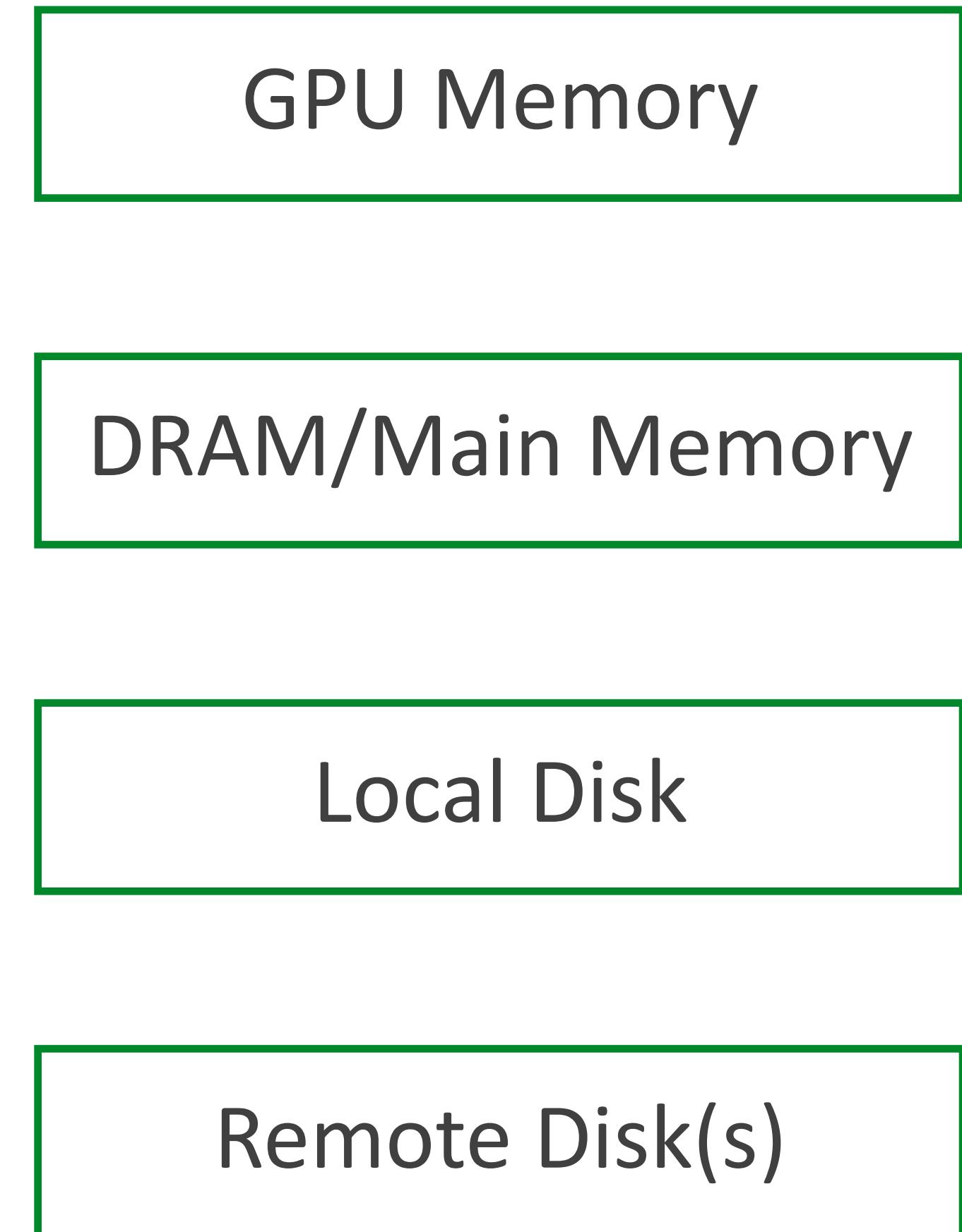
3. Dataset is too big for one node to finish on time!  
Copying full data across all cluster nodes wastes DRAM/disk  
Copying wastes DRAM/disk

3. MQO for DL Model Selection

# What Cerebro Does (Already Published)

Fastest data access  
Tiny; Costliest

Slowest data access  
Large; Cheaper  
Multiplicity



3. Dataset is too big for one node to finish on time!  
Copying full data across all cluster nodes wastes DRAM/disk  
Copying wastes DRAM/disk
3. MQO for DL Model Selection  
Training multiple models at scale on sharded data

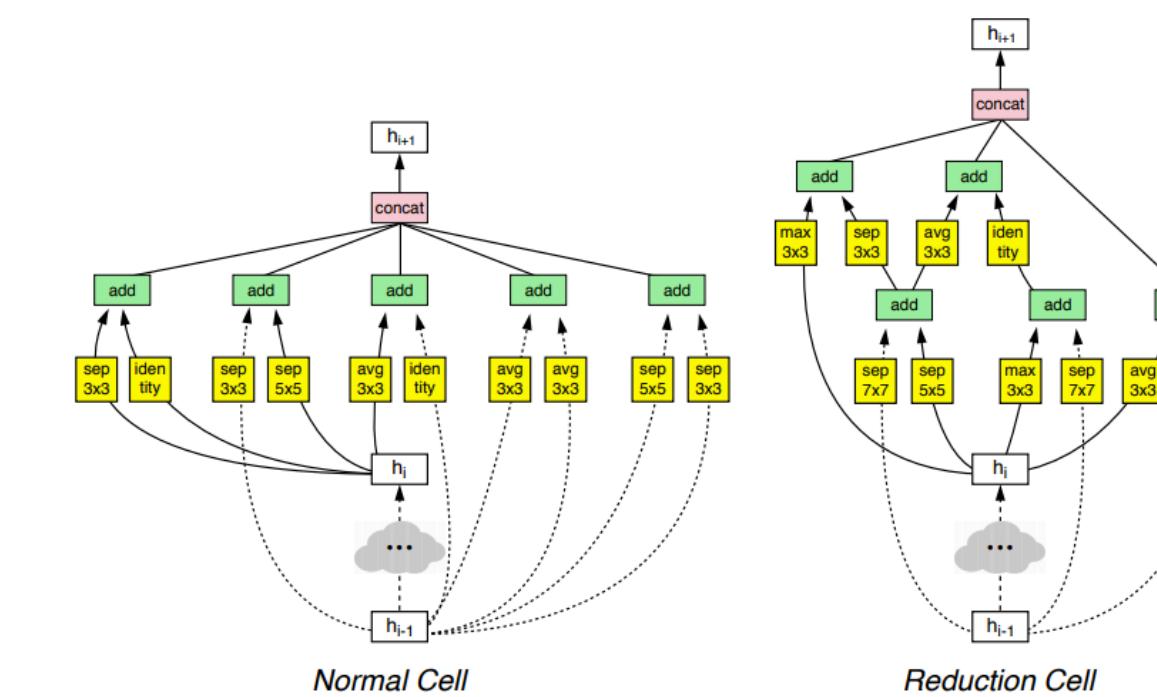
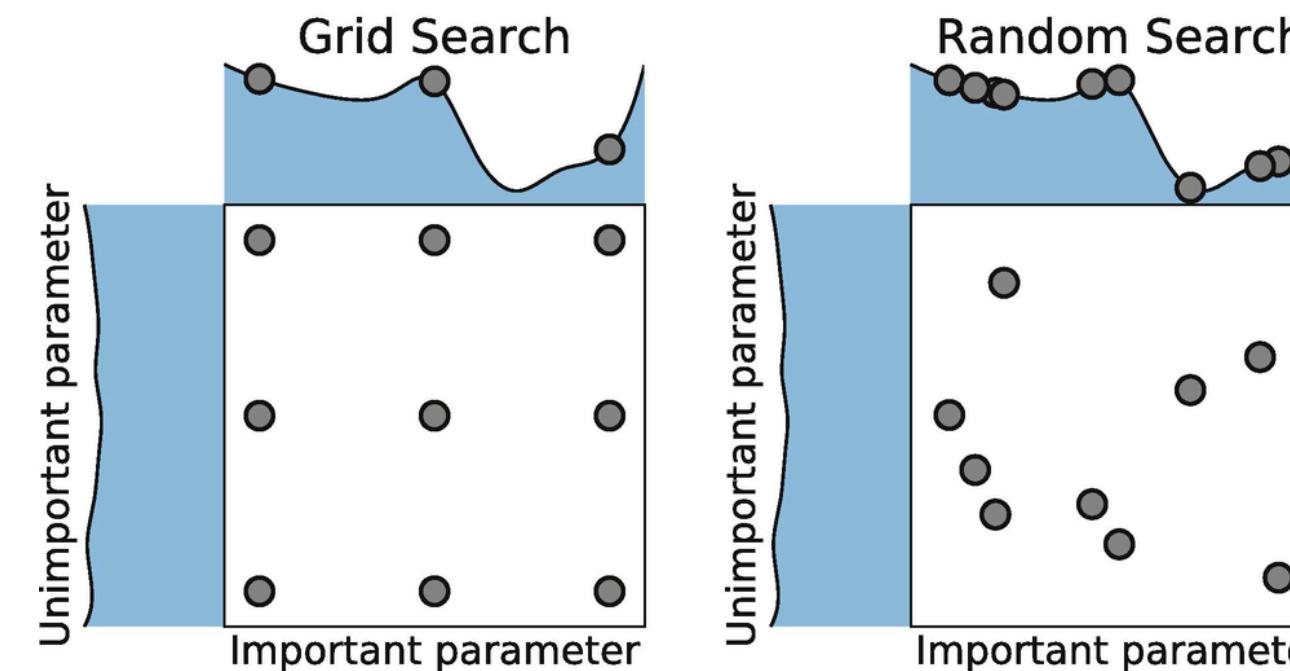
# MQO for DL Model Selection

*Cerebro: Efficient and Reproducible Model Selection on Deep Learning Systems.* **SIGMOD DEEM'19**  
*Cerebro: A Data System for Optimized Deep Learning Model Selection.* **VLDB'20**

# MQO for DL Model Selection

## Workload:

Hyper-parameter tuning and neural architecture exploration yield numerous (even 100s) of DL training configs

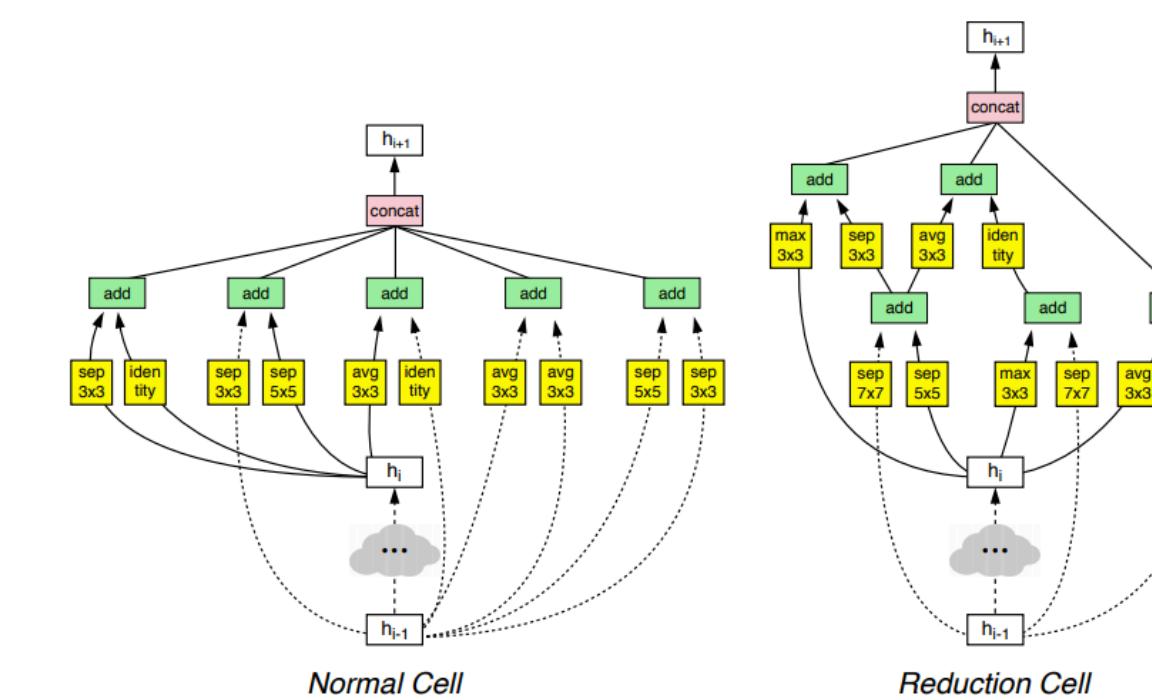
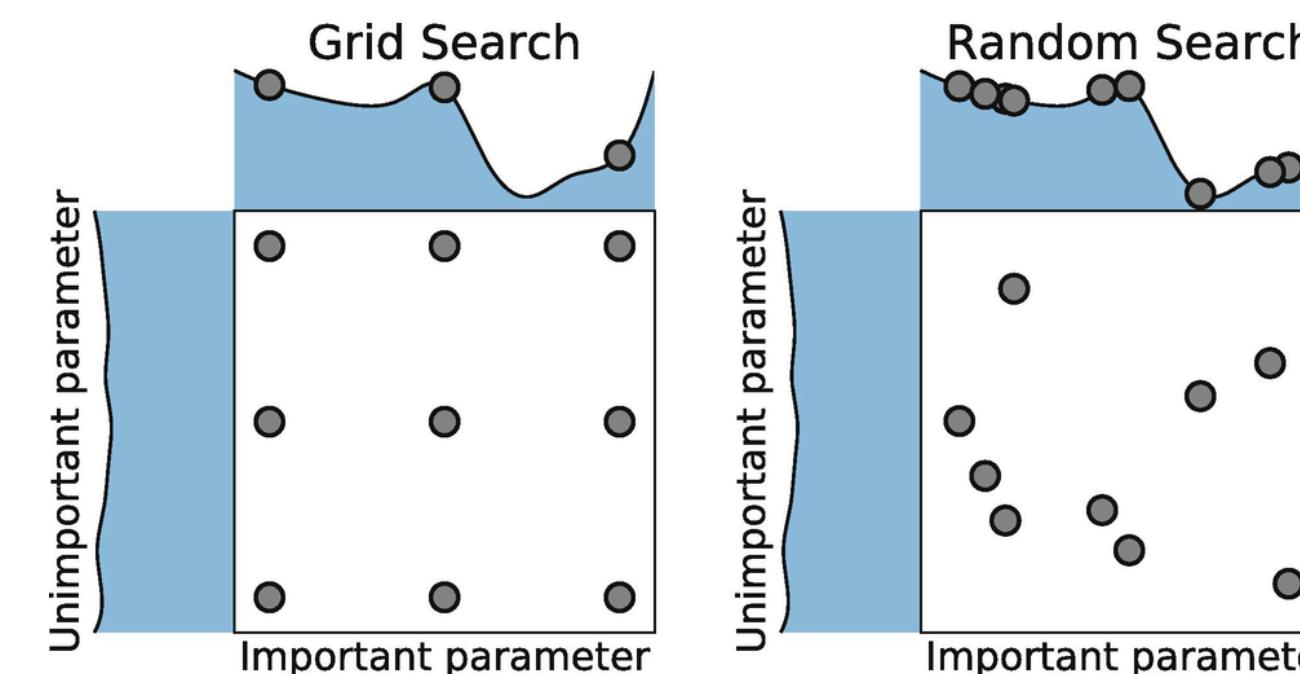


HyperBand,  
AutoKeras,  
ASHA, PBT,  
HyperOpt,  
NAS, etc.

# MQO for DL Model Selection

## Workload:

Hyper-parameter tuning and neural architecture exploration yield numerous (even 100s) of DL training configs



HyperBand,  
AutoKeras,  
ASHA, PBT,  
HyperOpt,  
NAS, etc.

## Multi-Query:

Run multiple configs on large dataset in one go

*Cerebro: Efficient and Reproducible Model Selection on Deep Learning Systems. SIGMOD DEEM'19*  
*Cerebro: A Data System for Optimized Deep Learning Model Selection. VLDB'20*

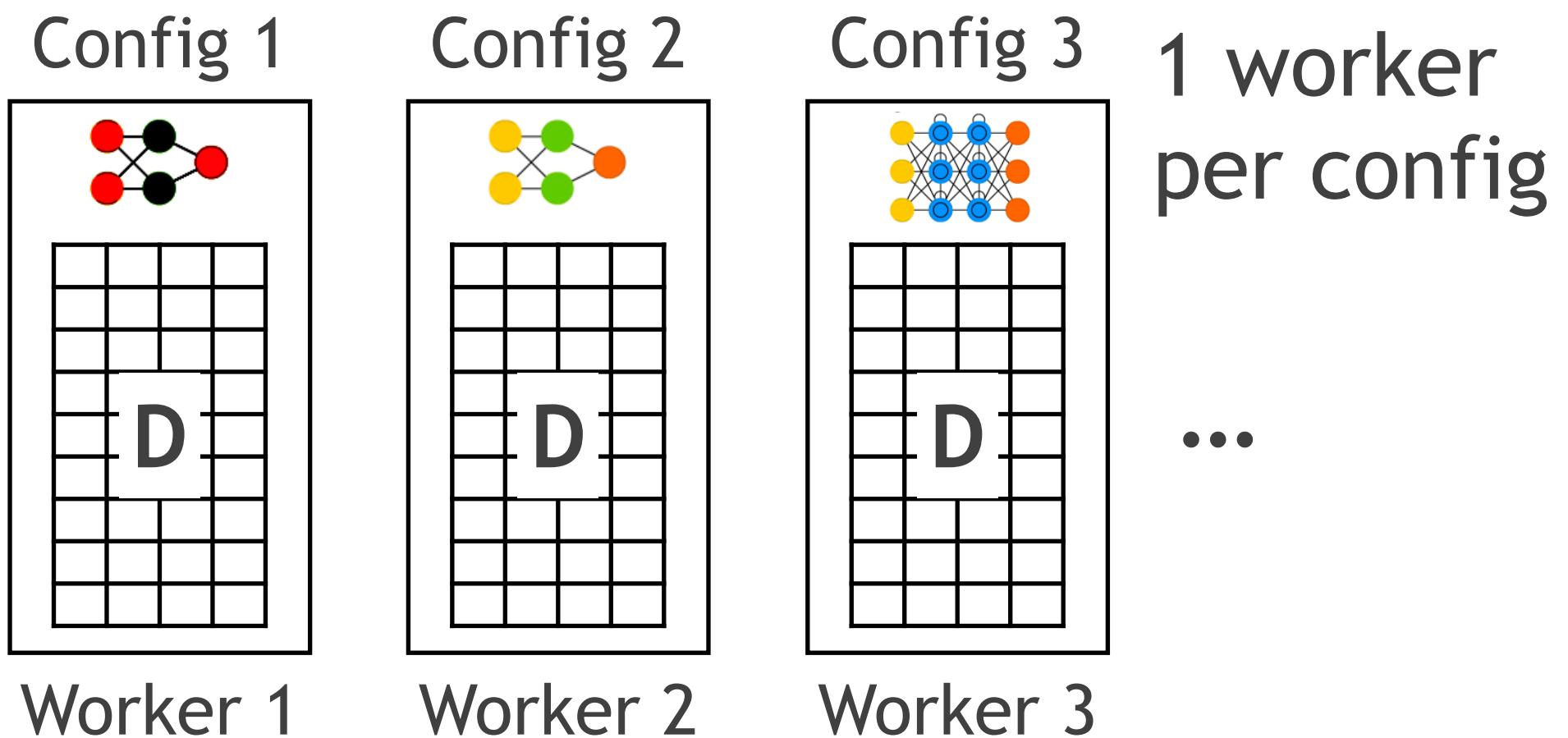
# Positioning Cerebro's Technique vs Prior Art

We devise a novel execution strategy called **Model Hopper Parallelism (MOP)**

# Positioning Cerebro's Technique vs Prior Art

We devise a novel execution strategy called **Model Hopper Parallelism (MOP)**

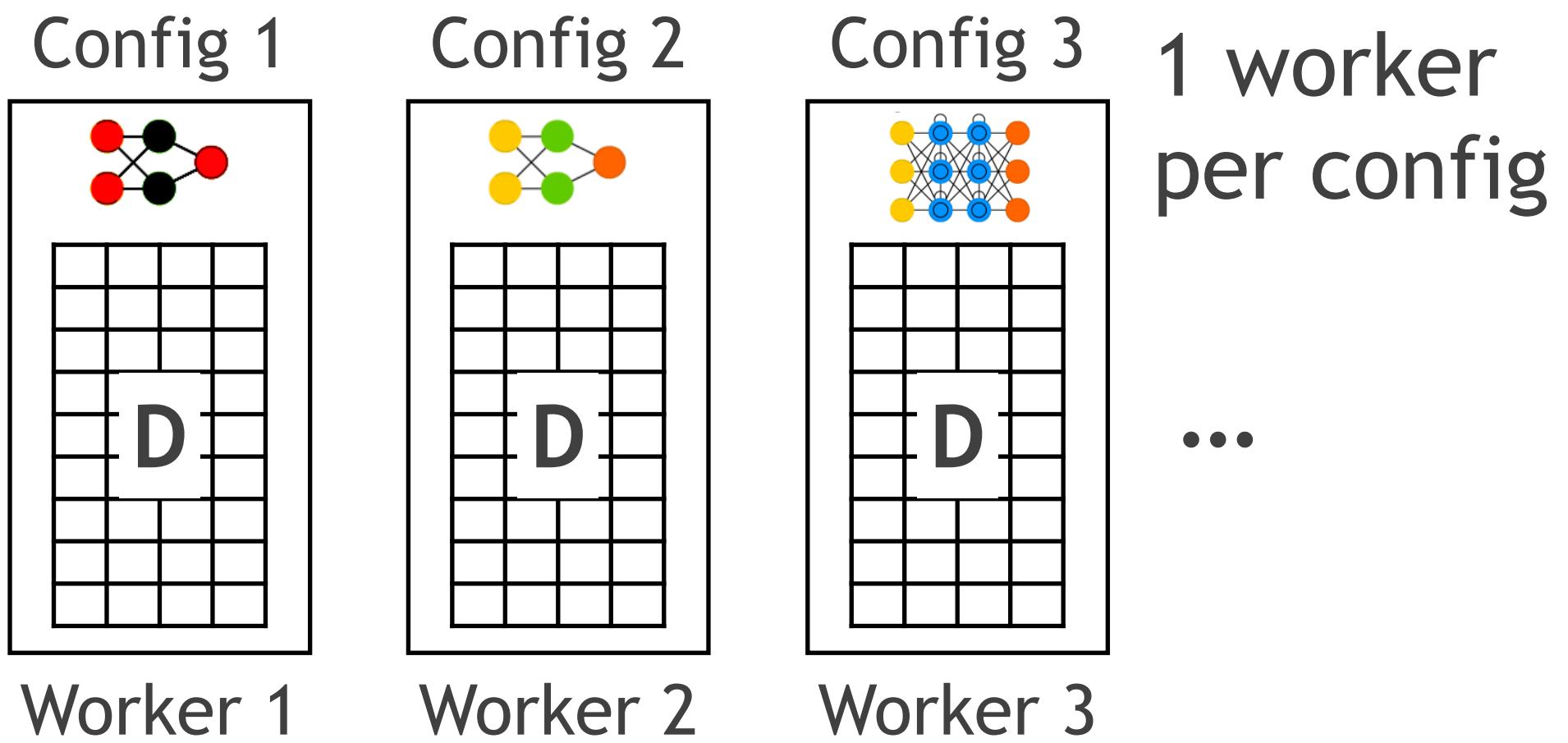
## Task Parallelism:



# Positioning Cerebro's Technique vs Prior Art

We devise a novel execution strategy called **Model Hopper Parallelism (MOP)**

## Task Parallelism:

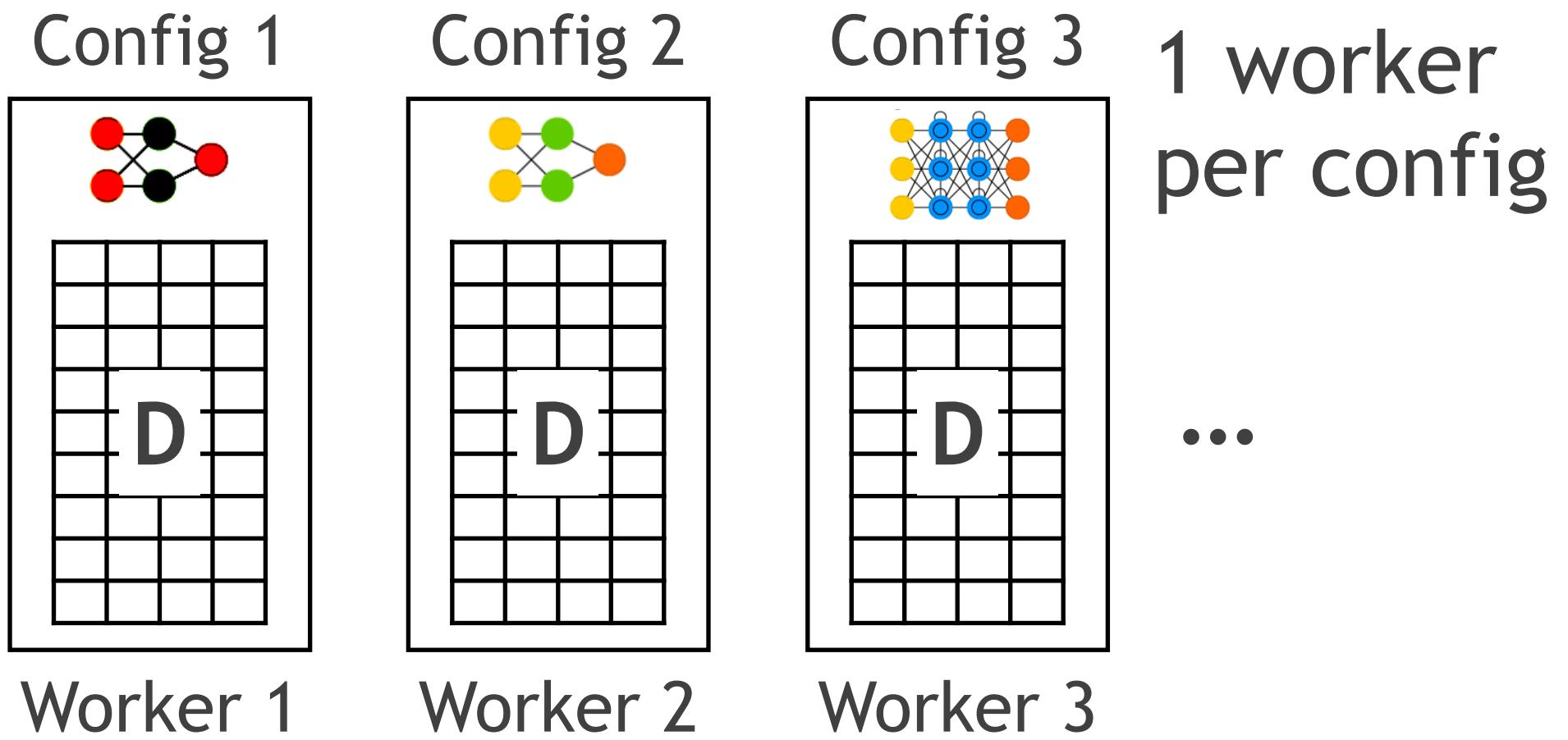


- + High throughput model selection
- + Best accuracy from Sequential SGD
- Low data scalability; wastes space  
(copy) or network (remote read)

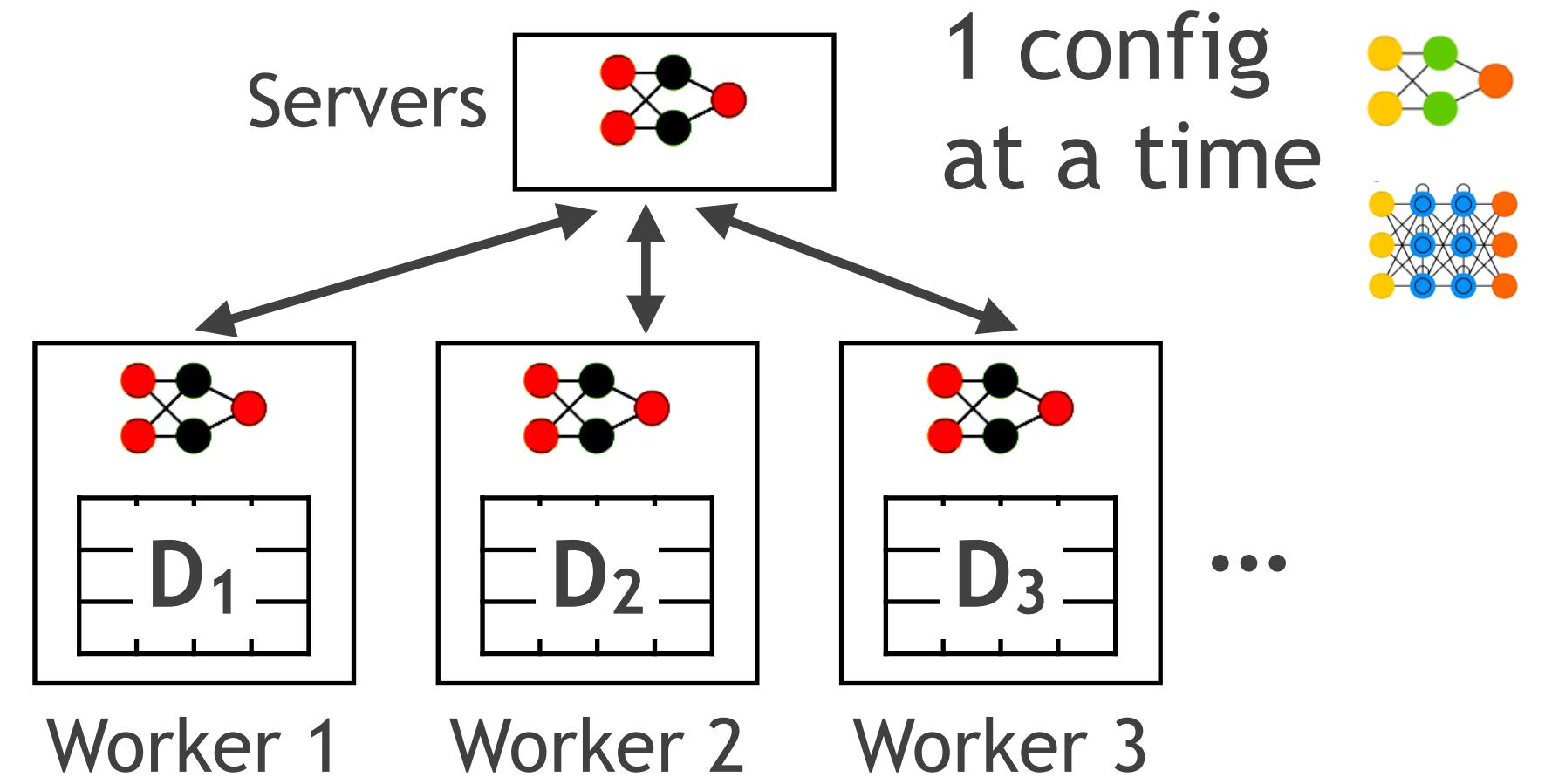
# Positioning Cerebro's Technique vs Prior Art

We devise a novel execution strategy called **Model Hopper Parallelism (MOP)**

## Task Parallelism:



## Data Parallelism:

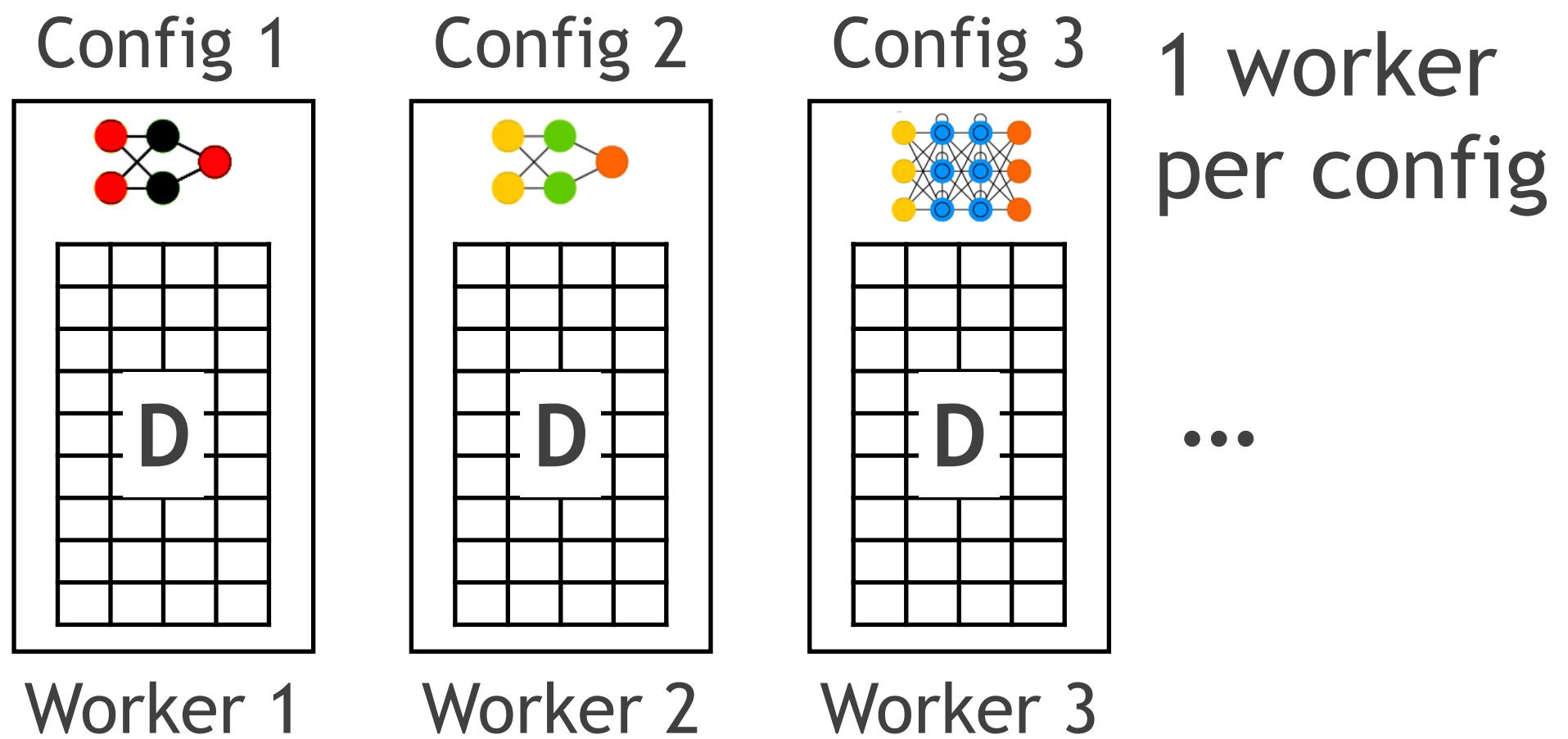


- + High throughput model selection
- + Best accuracy from Sequential SGD
- Low data scalability; wastes space  
(copy) or network (remote read)

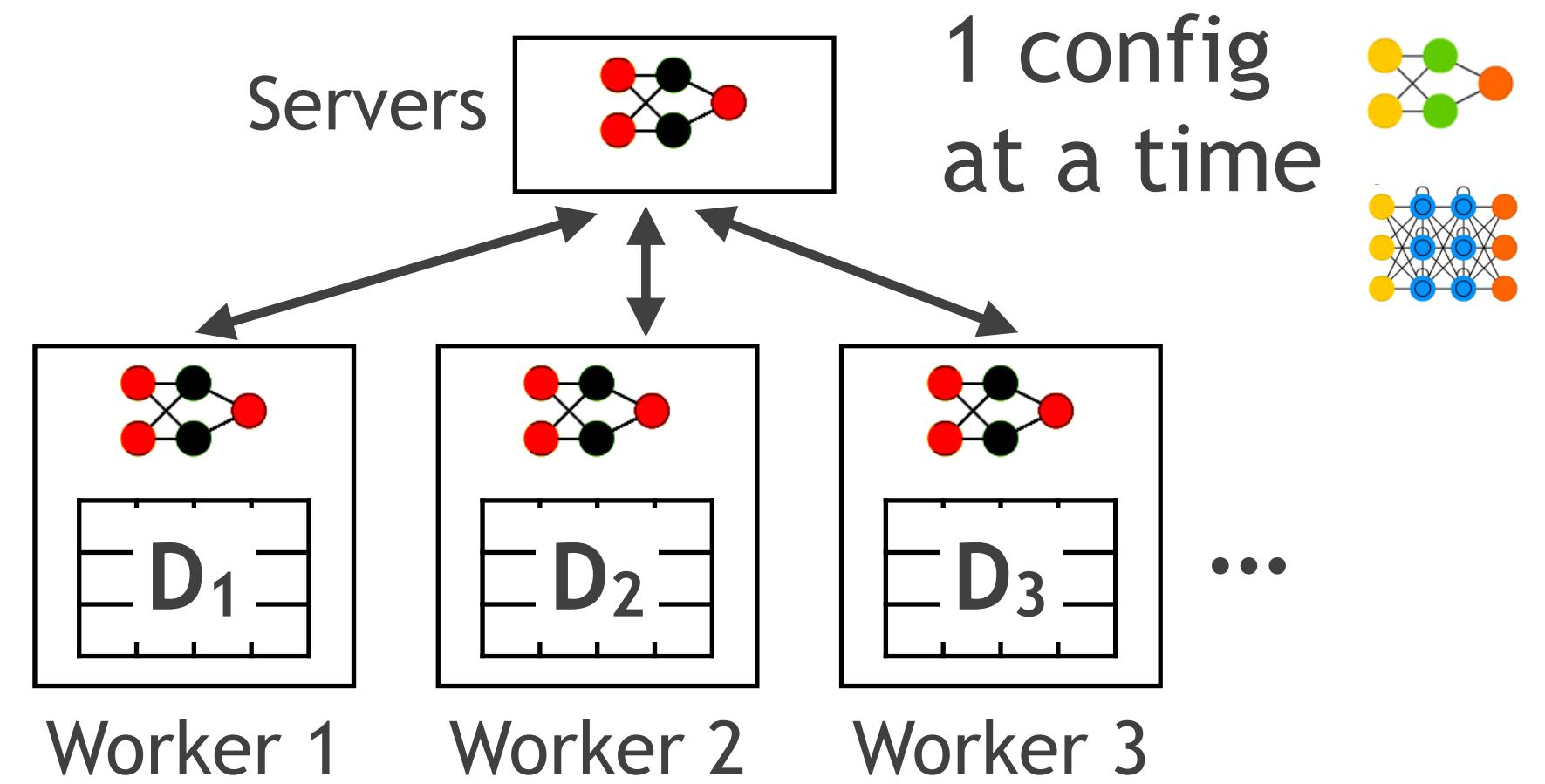
# Positioning Cerebro's Technique vs Prior Art

We devise a novel execution strategy called **Model Hopper Parallelism (MOP)**

## Task Parallelism:



## Data Parallelism:



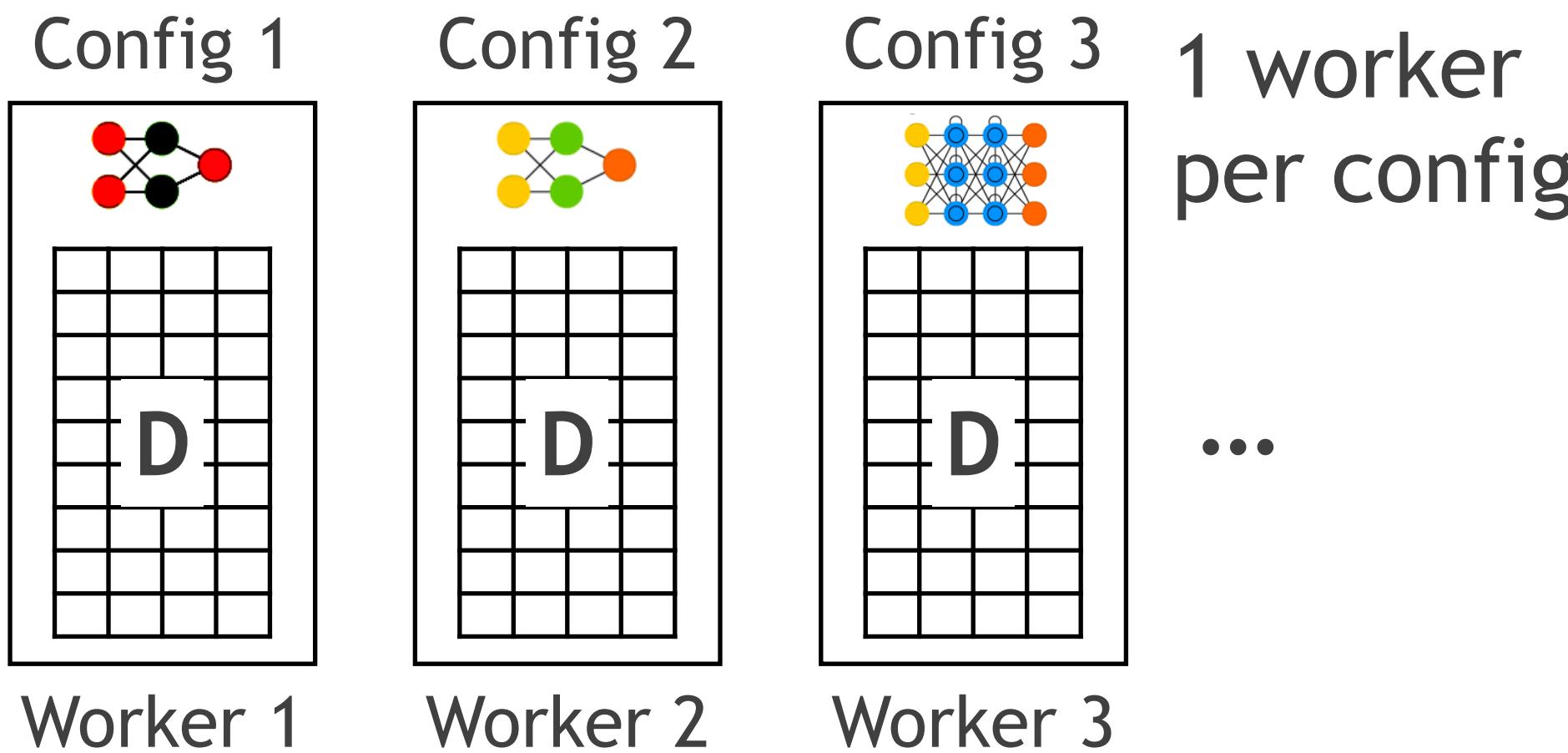
- + High throughput model selection
- + Best accuracy from Sequential SGD
- Low data scalability; wastes space (copy) or network (remote read)

- + High data scalability via sharding
- Shard-level SGD does not converge; mini-batch level is too slow
- Very low throughput overall

# Positioning Cerebro's Technique vs Prior Art

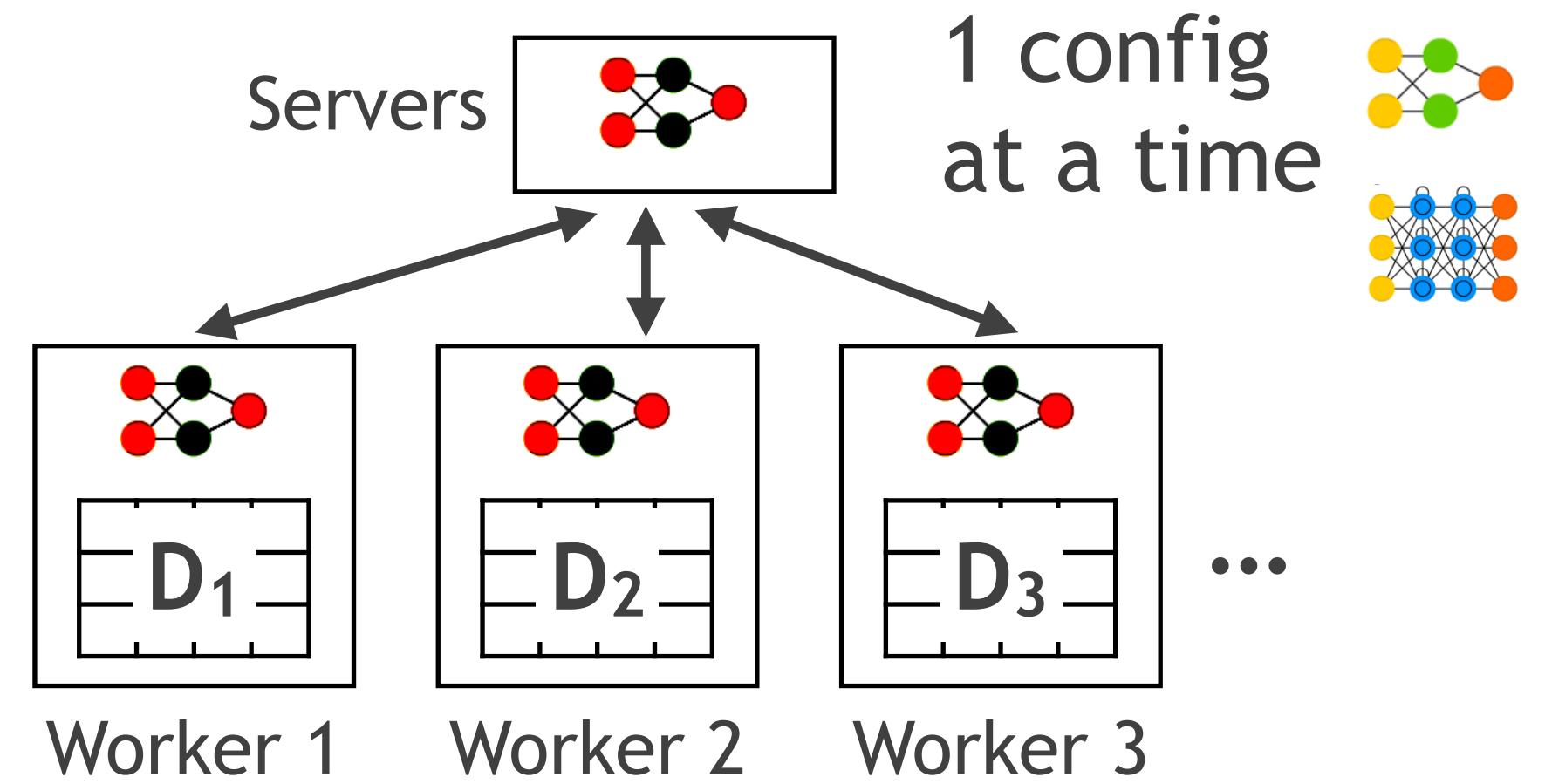
We devise a novel execution strategy called **Model Hopper Parallelism (MOP)**

## Task Parallelism:



- + High throughput model selection
- + Best accuracy from Sequential SGD
- Low data scalability; wastes space (copy) or network (remote read)

## Data Parallelism:



## MOP:

New hybrid of data and task parallelism  
**Best of both worlds!**

- + High data scalability via sharding
- Shard-level SGD does not converge; mini-batch level is too slow
- Very low throughput overall



# Model Hopper Parallelism (MOP)

**Insight from Optimization Theory:**

SGD is robust to *data ordering randomness*

# Model Hopper Parallelism (MOP)

**Insight from Optimization Theory:**

SGD is robust to *data ordering randomness*

Shuffle and partition dataset

Run  $n$  DNNs on  $n$  workers

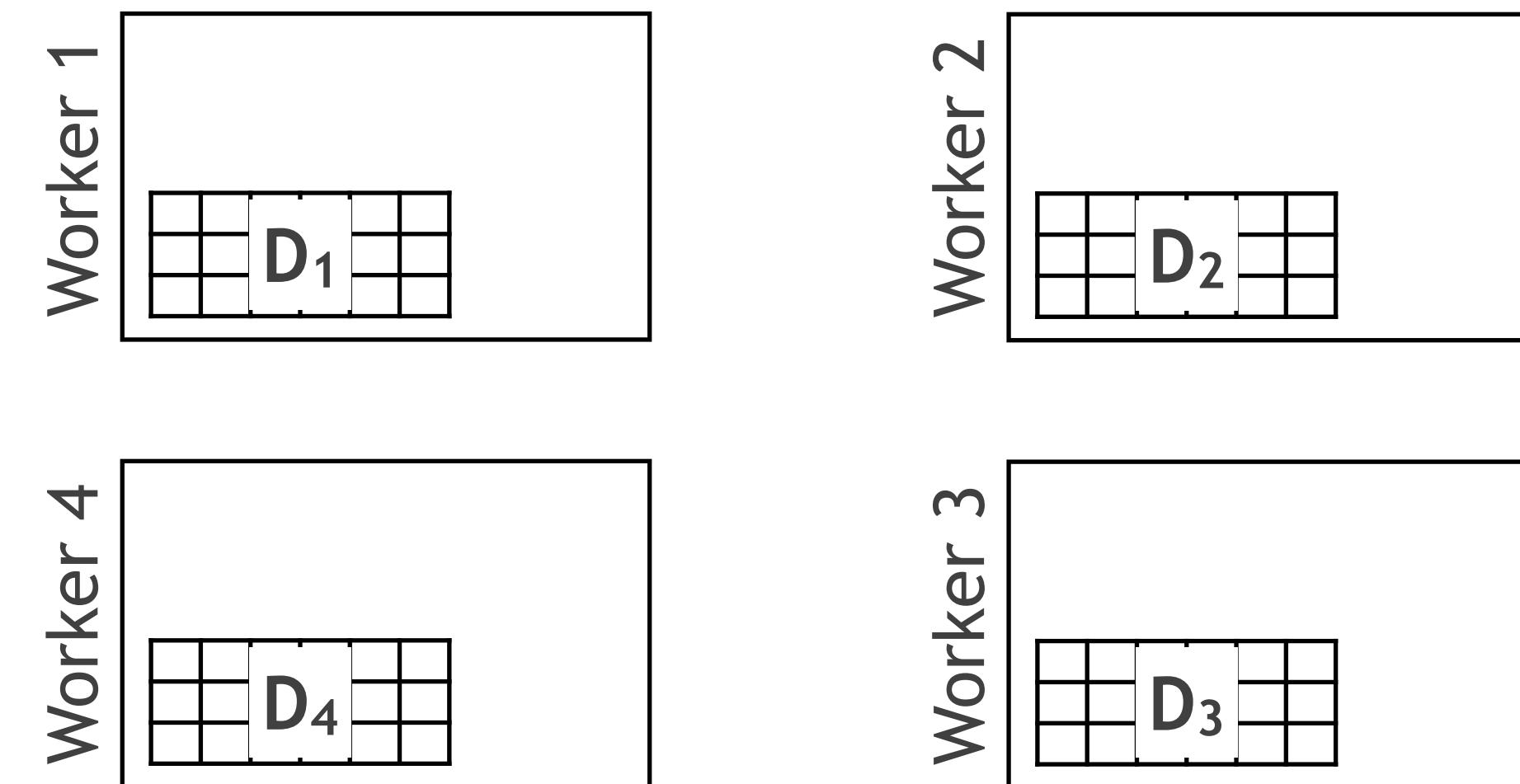
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset

Run  $n$  DNNs on  $n$  workers



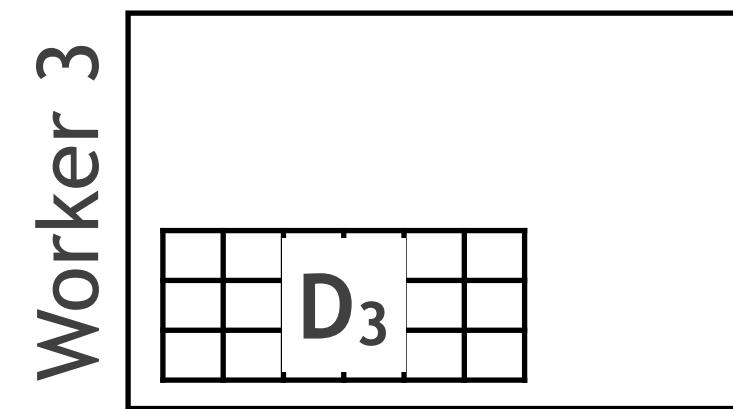
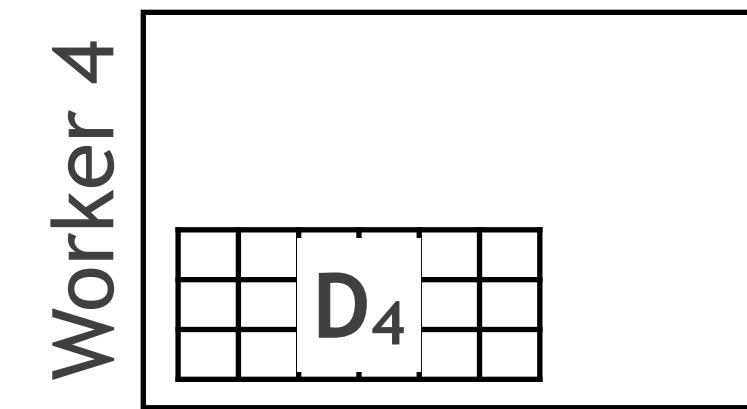
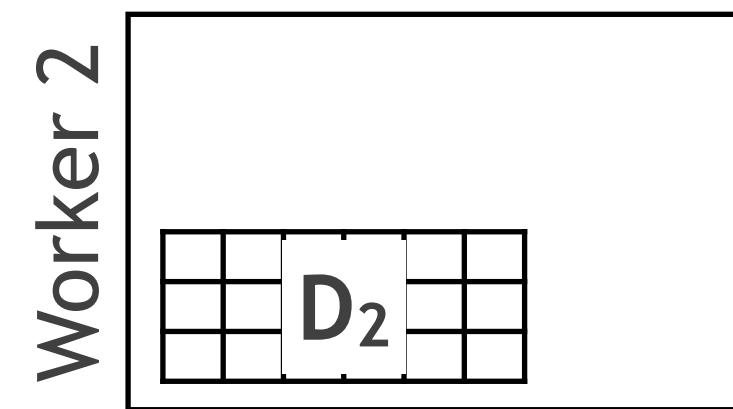
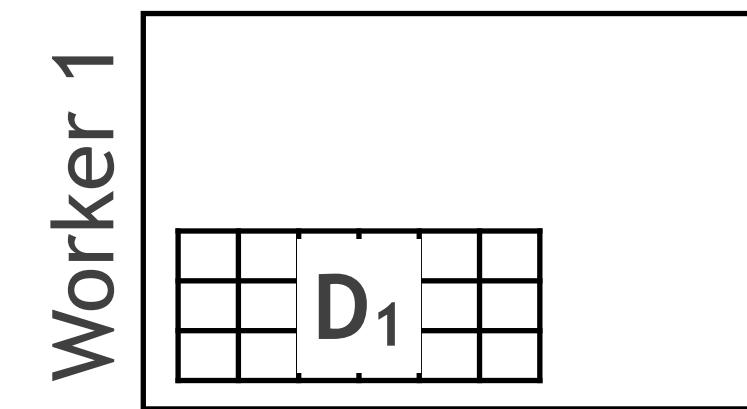
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Epoch 1.1 starts in parallel



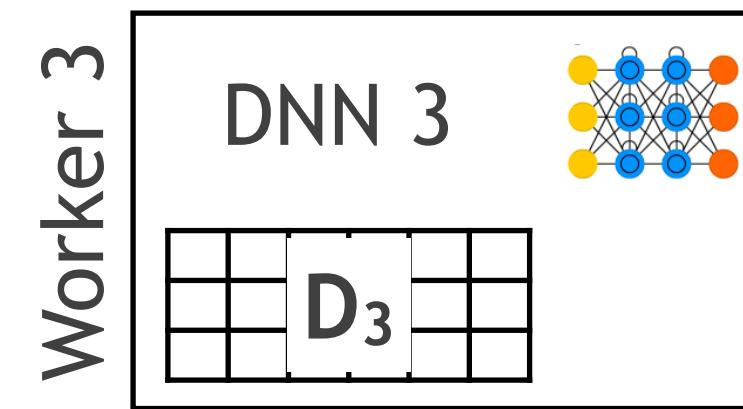
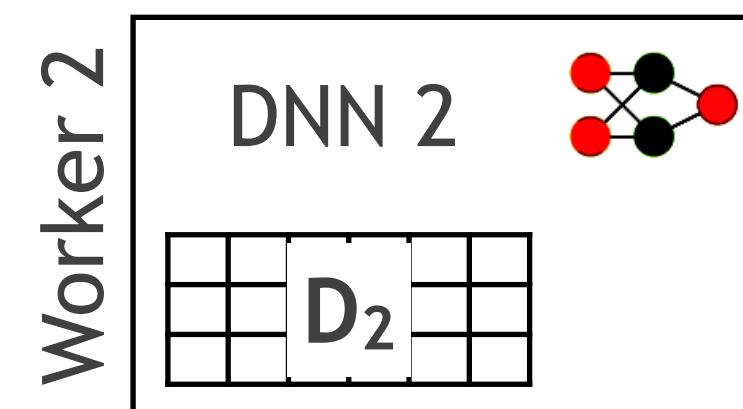
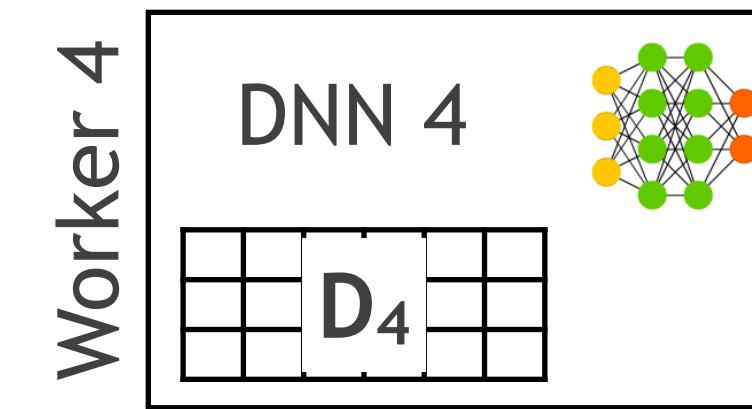
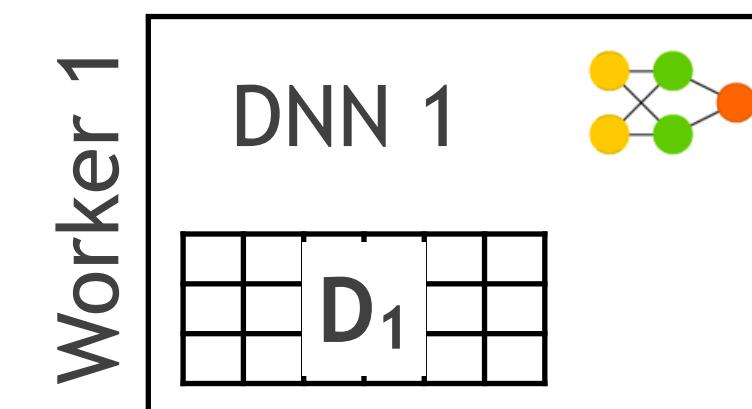
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Epoch 1.1 starts in parallel



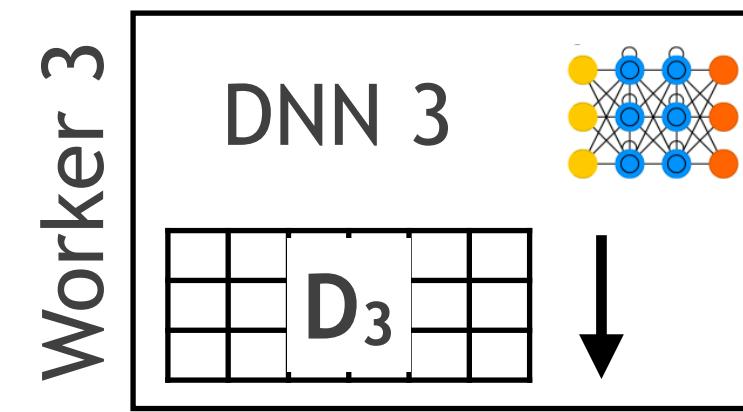
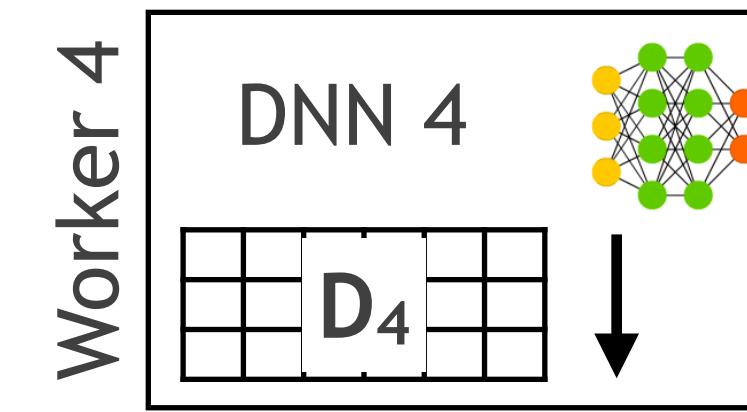
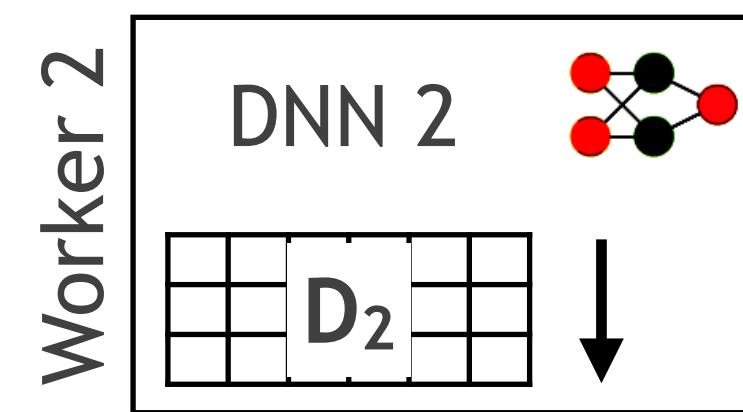
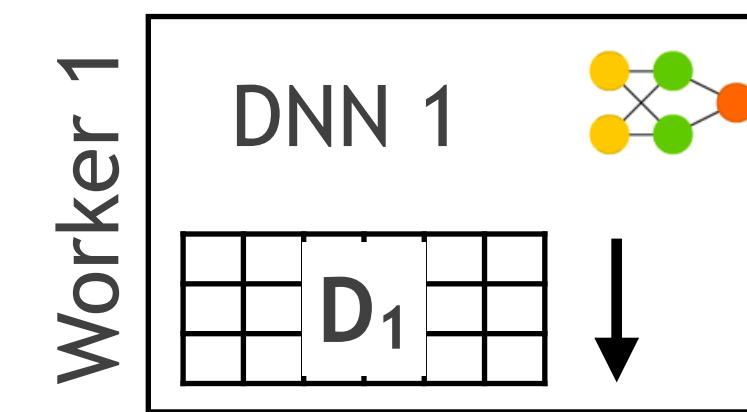
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Epoch 1.1 starts in parallel



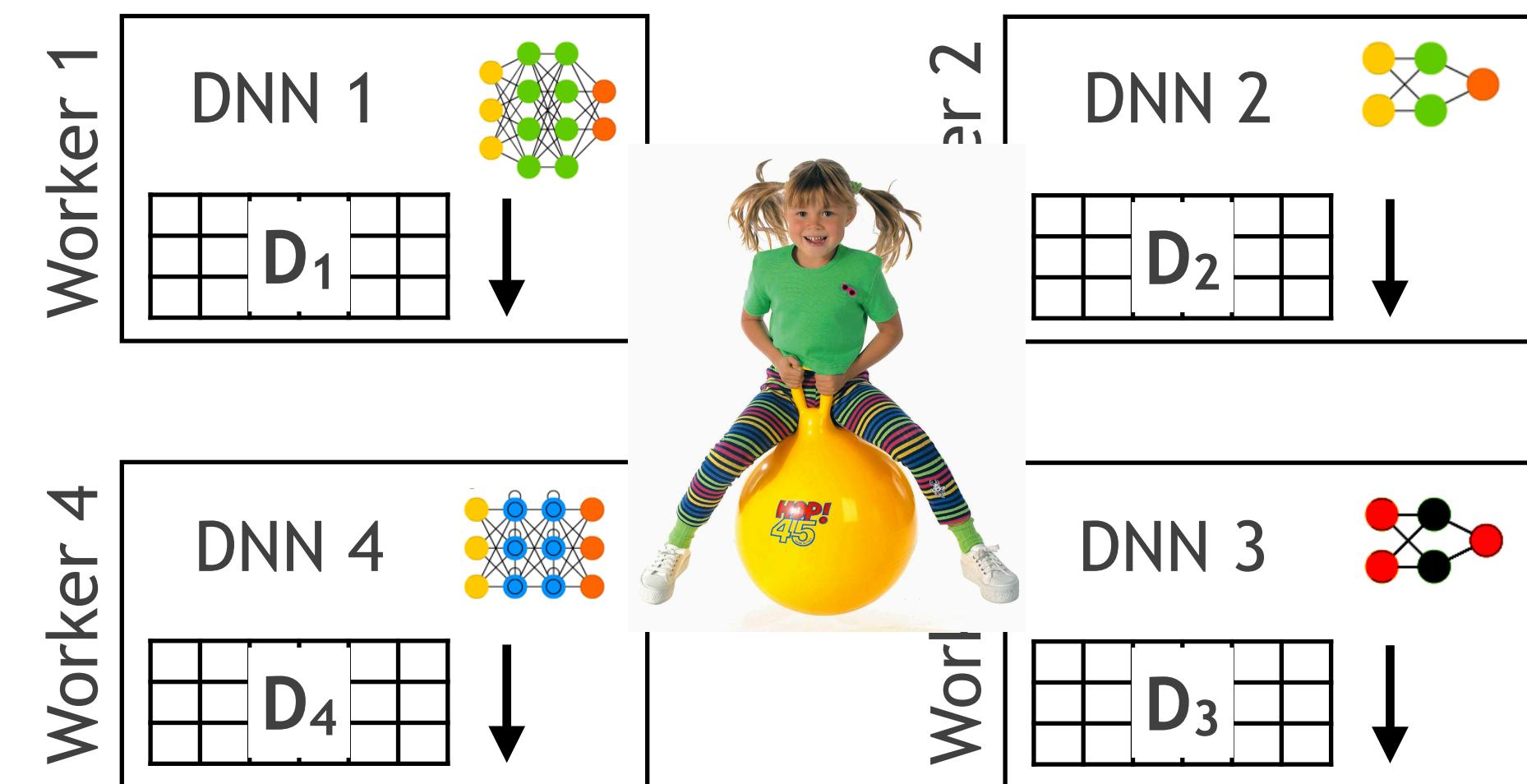
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Epoch 1.1 starts in parallel



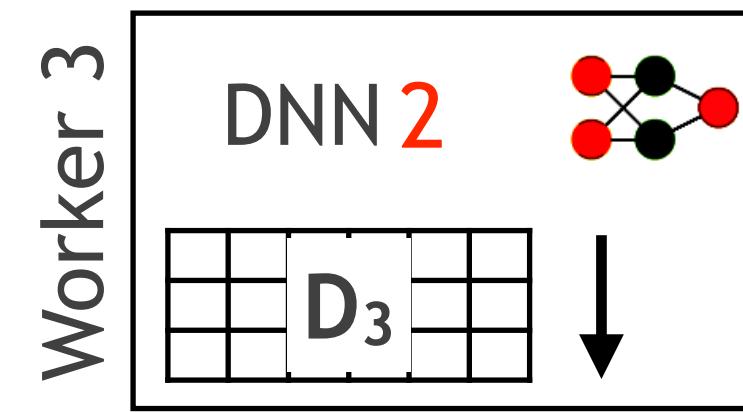
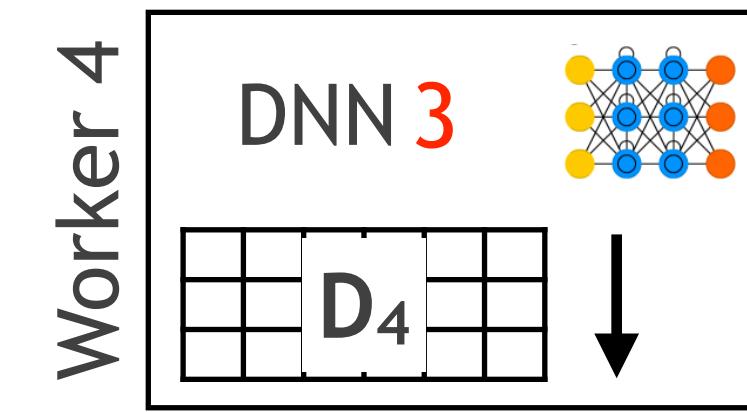
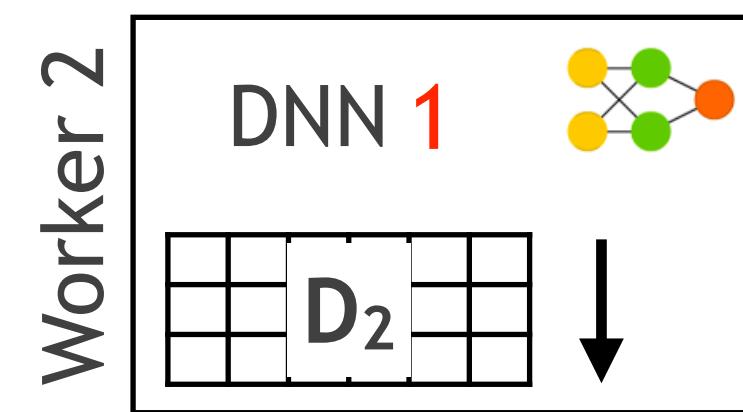
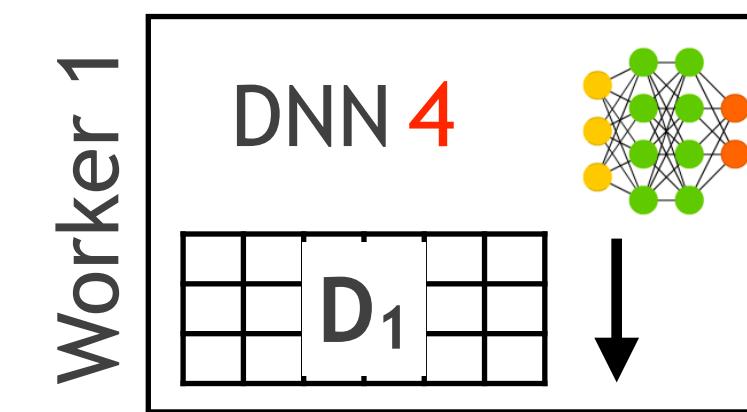
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Epoch 1.1 starts in parallel



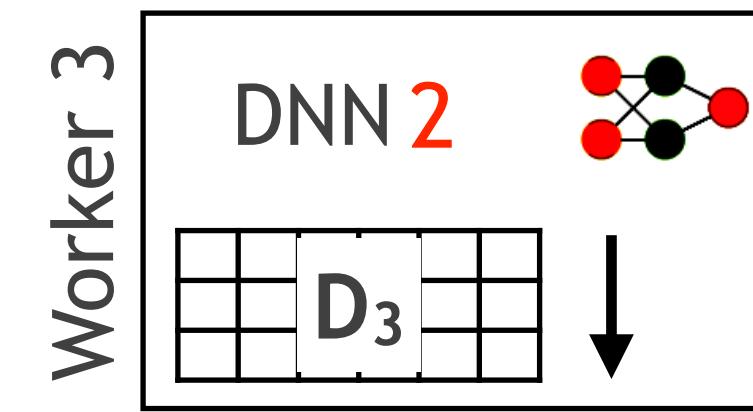
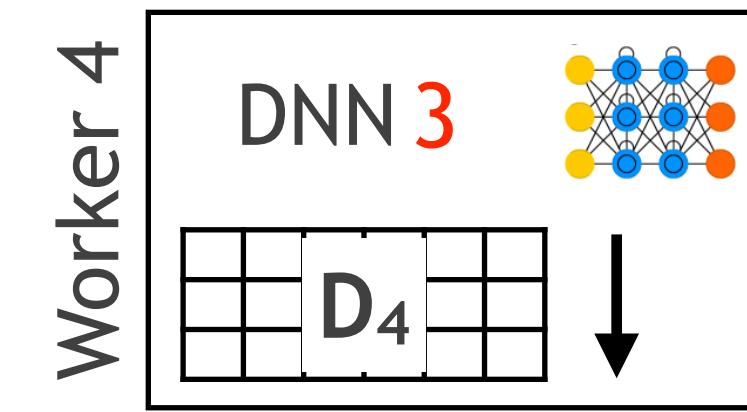
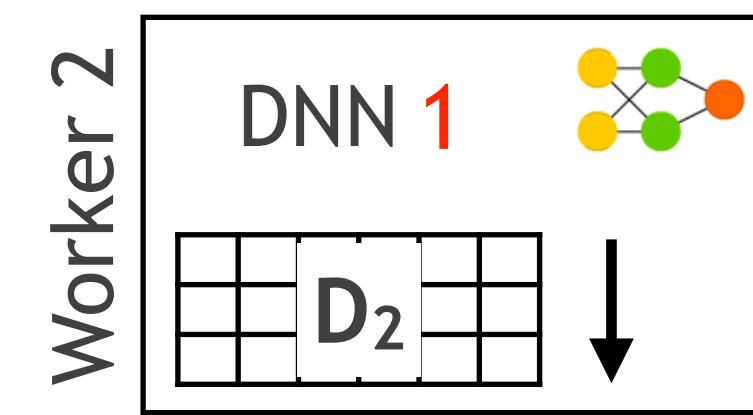
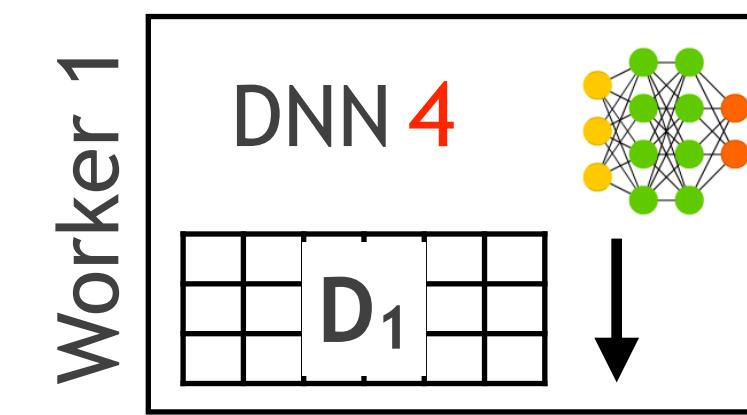
# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Epoch 1.2 starts in parallel



# Model Hopper Parallelism (MOP)

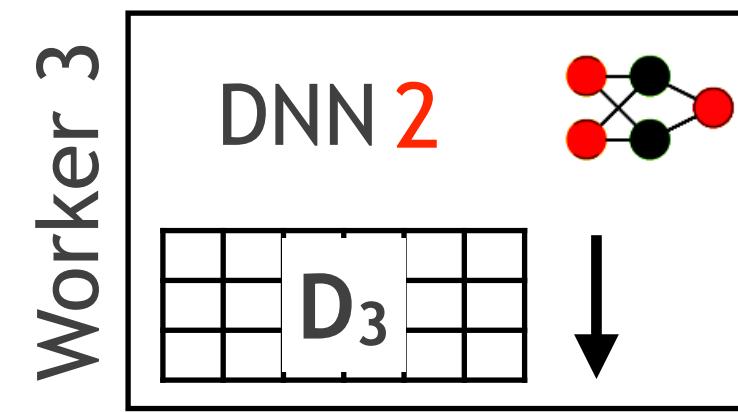
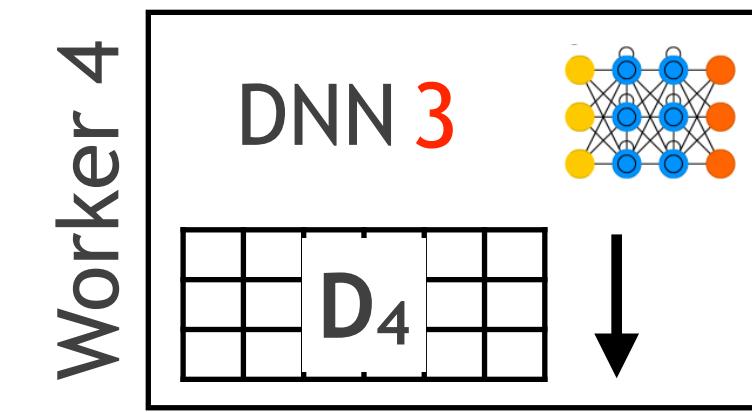
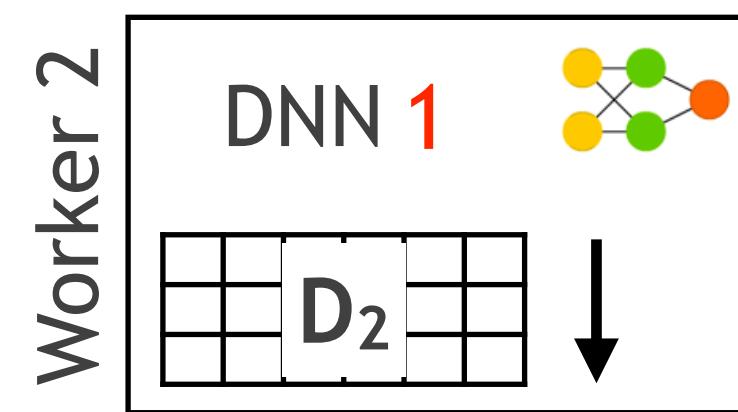
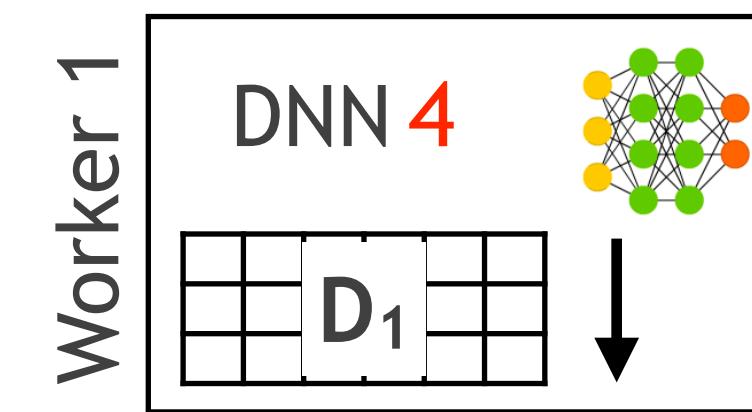
Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Each model keeps “hopping” until  
it sees all of D

Epoch 1.2 starts in parallel



# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

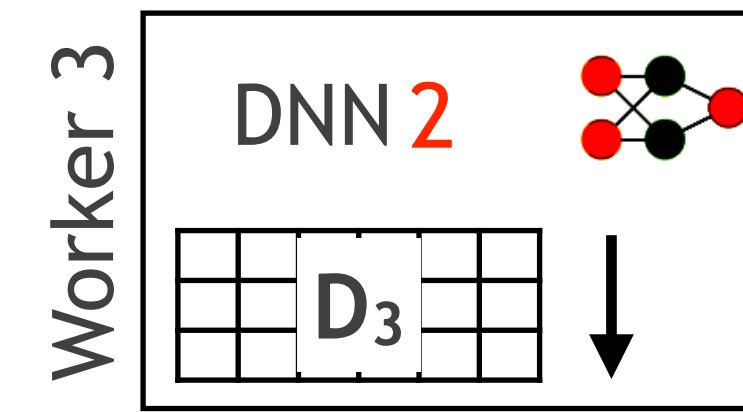
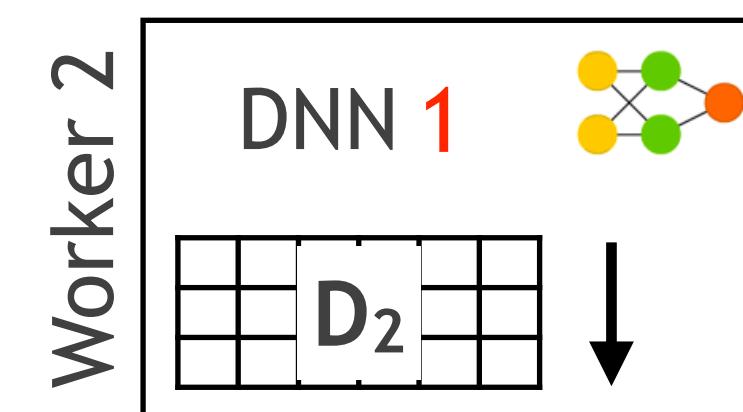
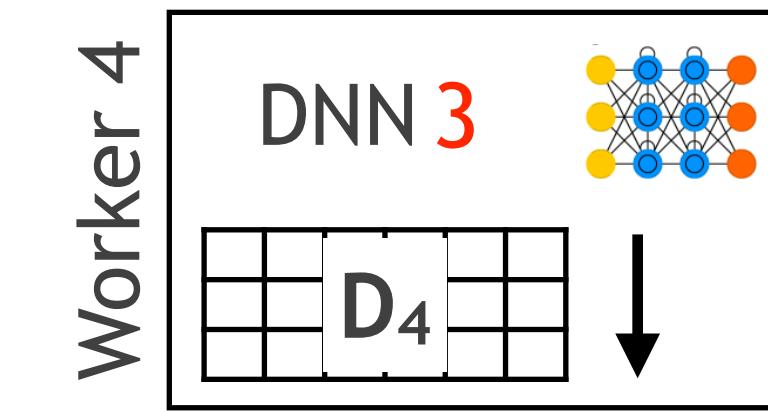
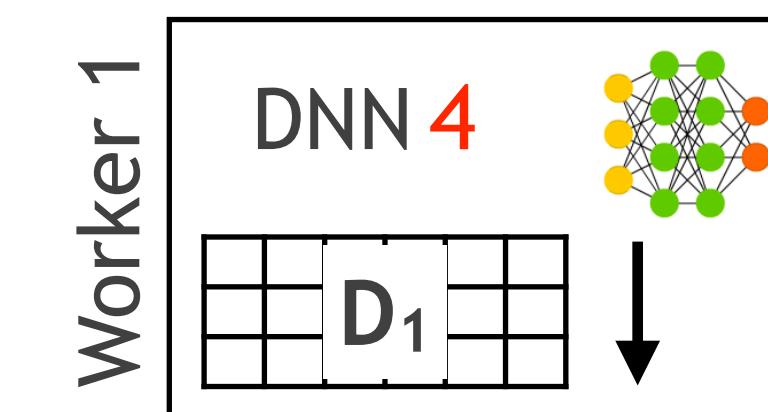
Shuffle and partition dataset  
Run  $n$  DNNs on  $n$  workers

Each model keeps “hopping” until  
it sees all of D

**Strong theoretical guarantees:**

1. *Equivalence* to sequential SGD
2. Hits lower bound on comm. cost

Epoch 1.2 starts in parallel



# Model Hopper Parallelism (MOP)

Insight from Optimization Theory:

SGD is robust to *data ordering randomness*

Technical Challenges (See paper for details):

Resource-aware scheduling of evolving model configs' hops

Support data replication, fault tolerance, and elasticity

Non-disruptive integration with existing DL systems

1. *Equivalence* to sequential SGD
2. Hits lower bound on comm. cost

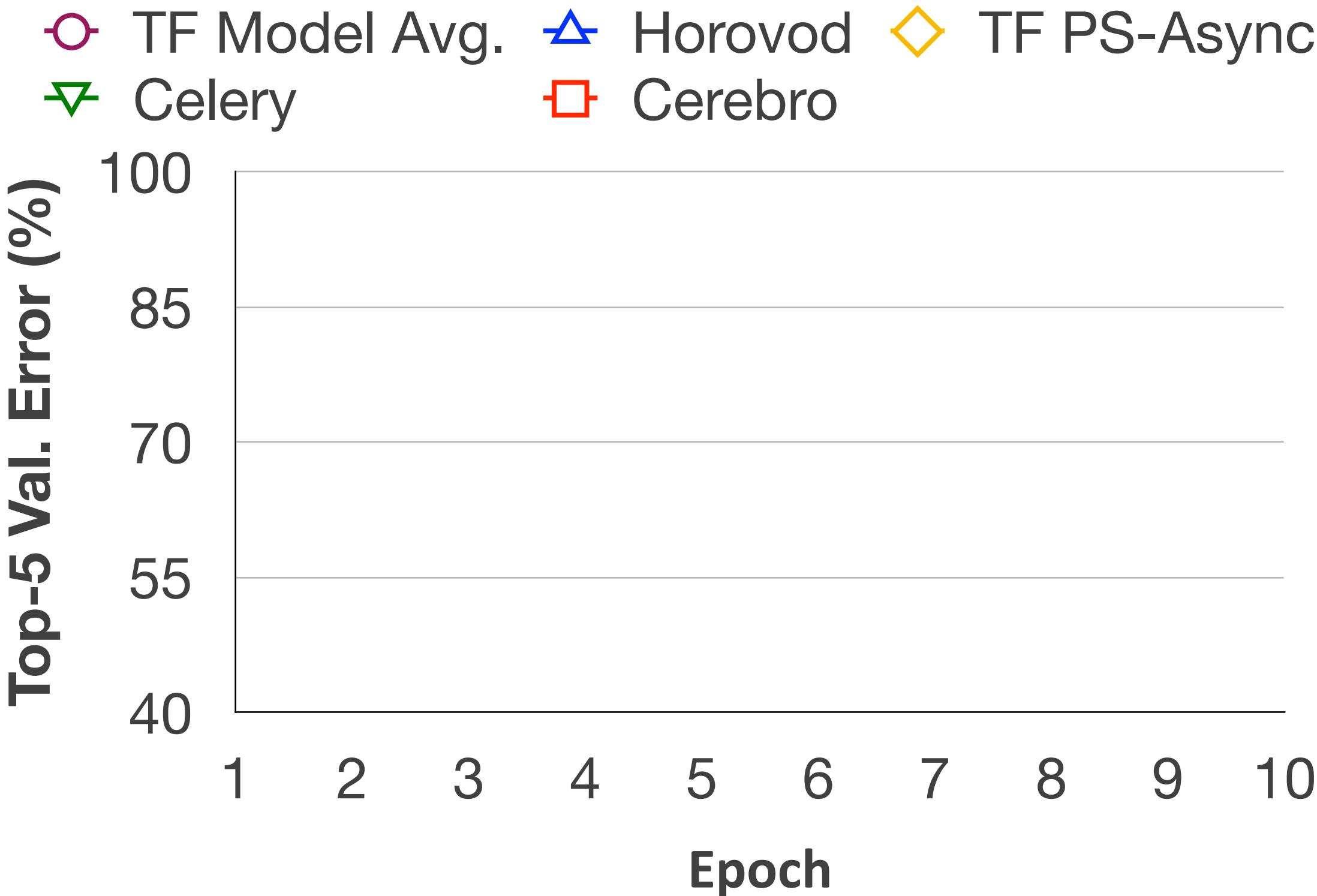


# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes

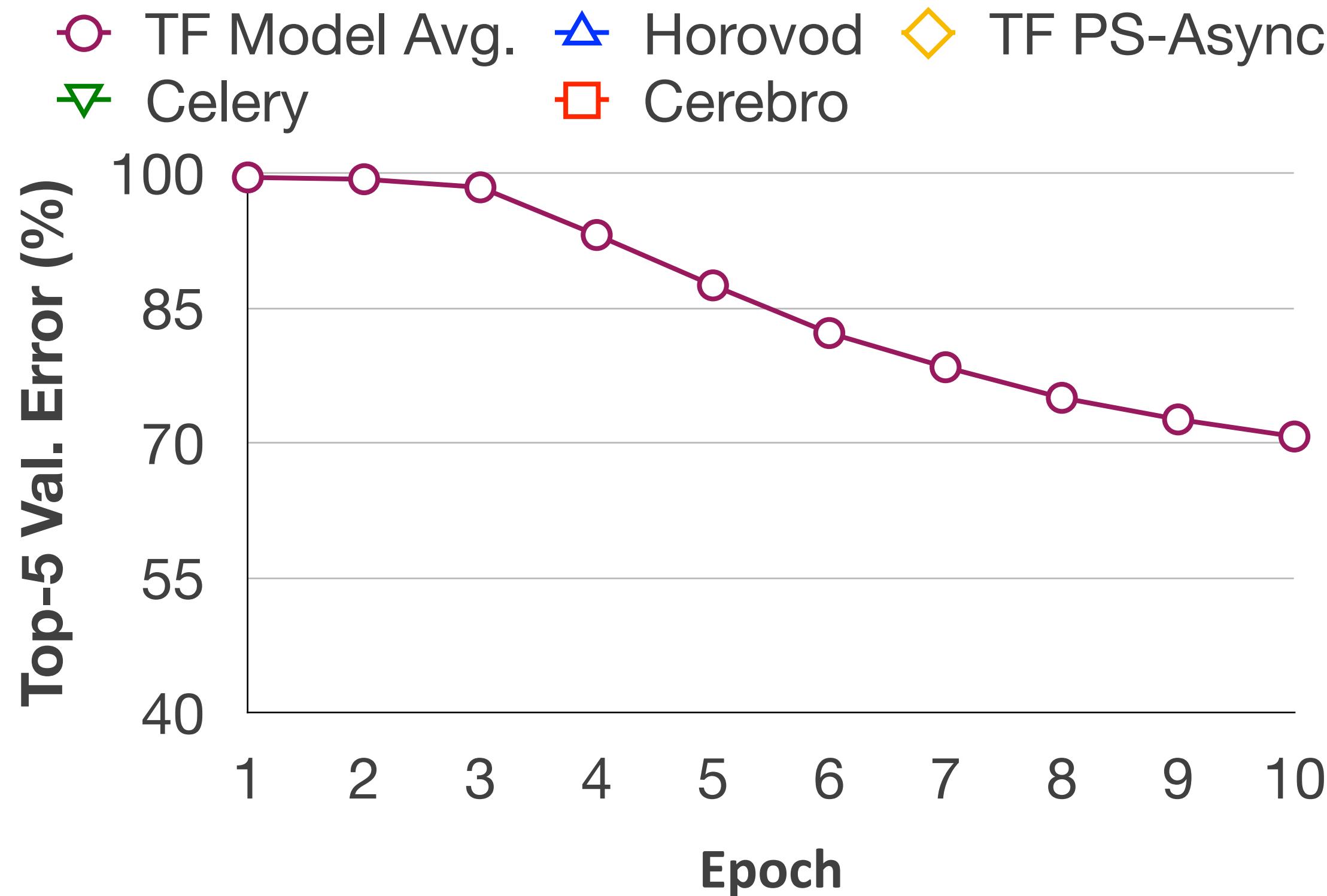
# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



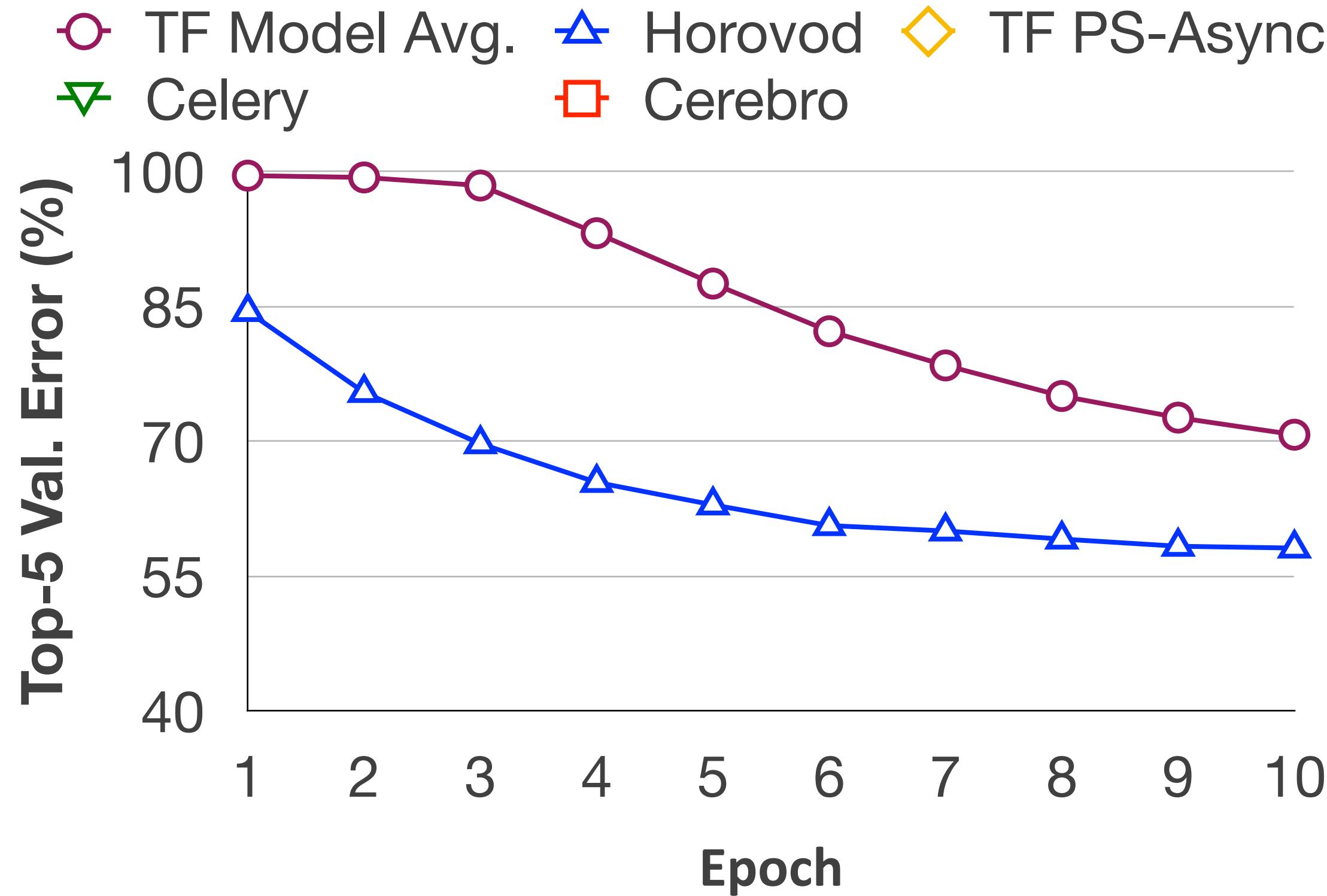
# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



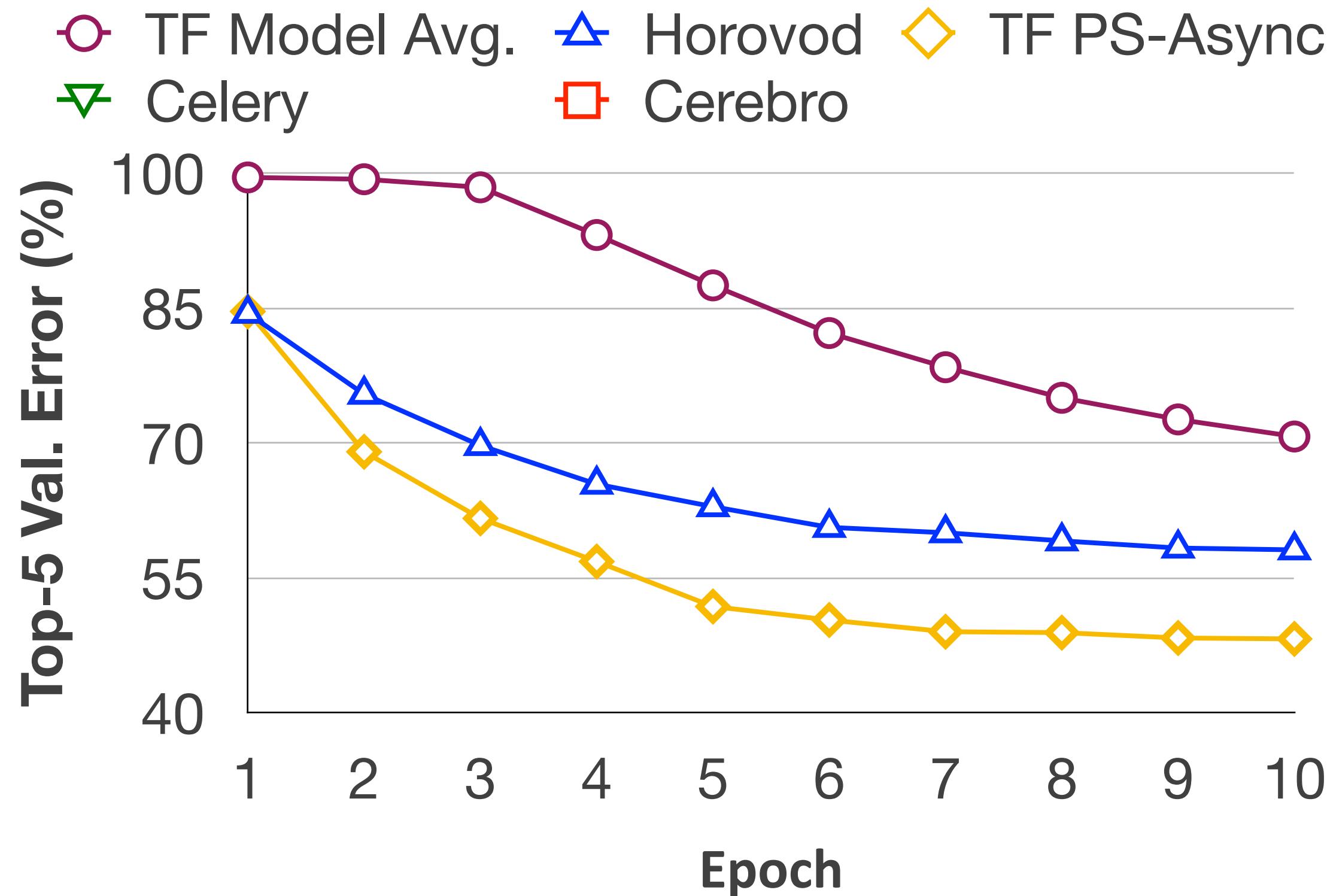
# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



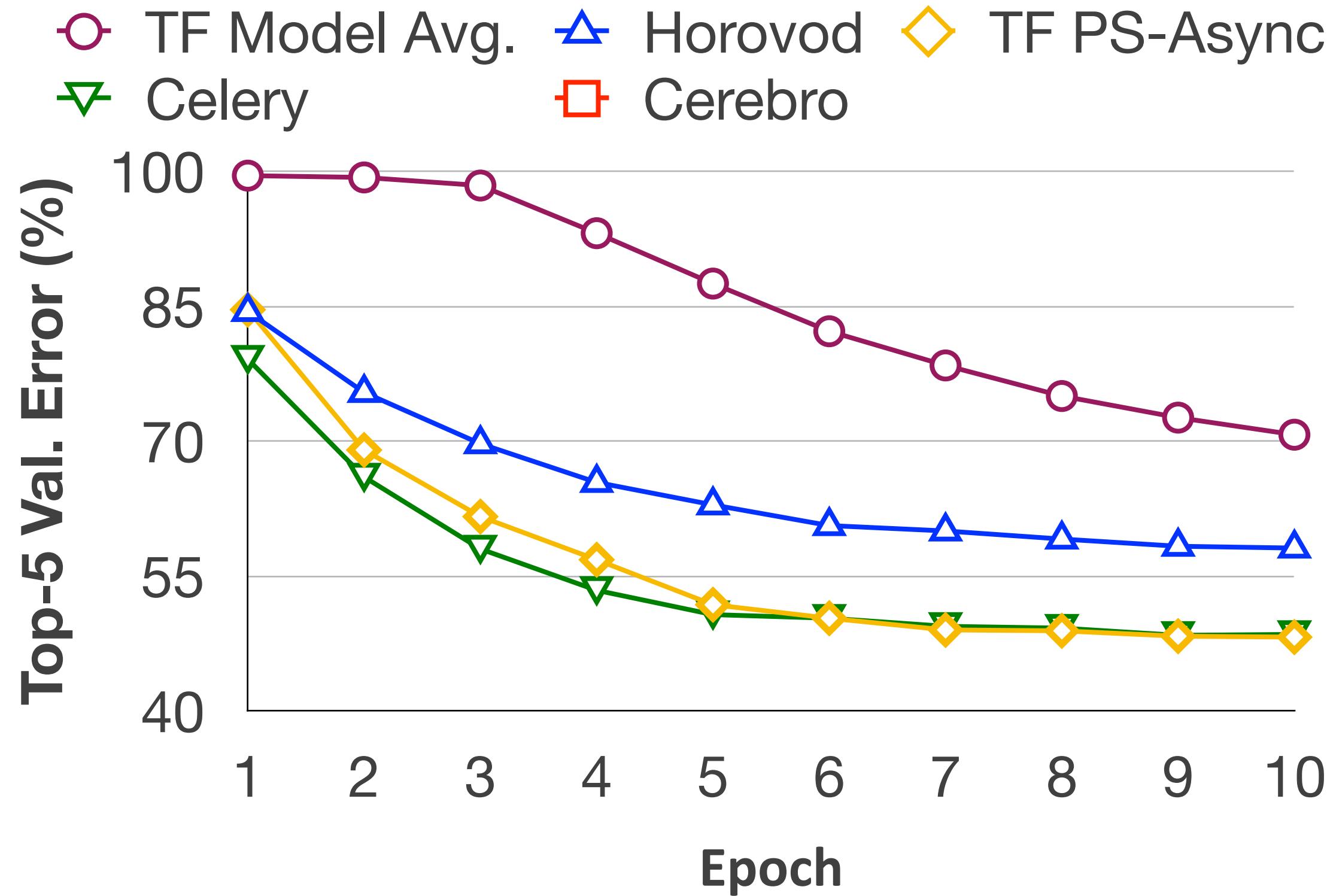
# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



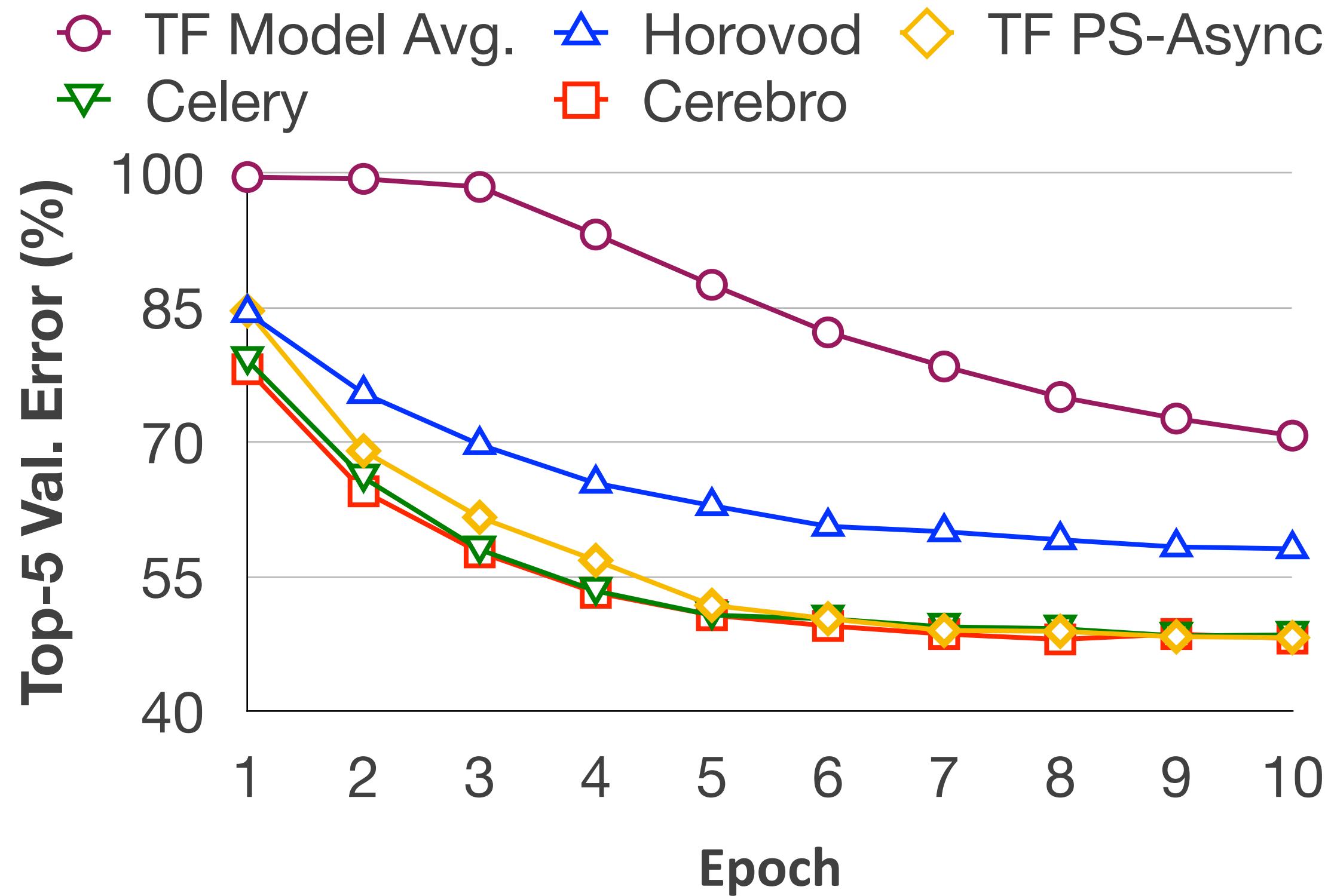
# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



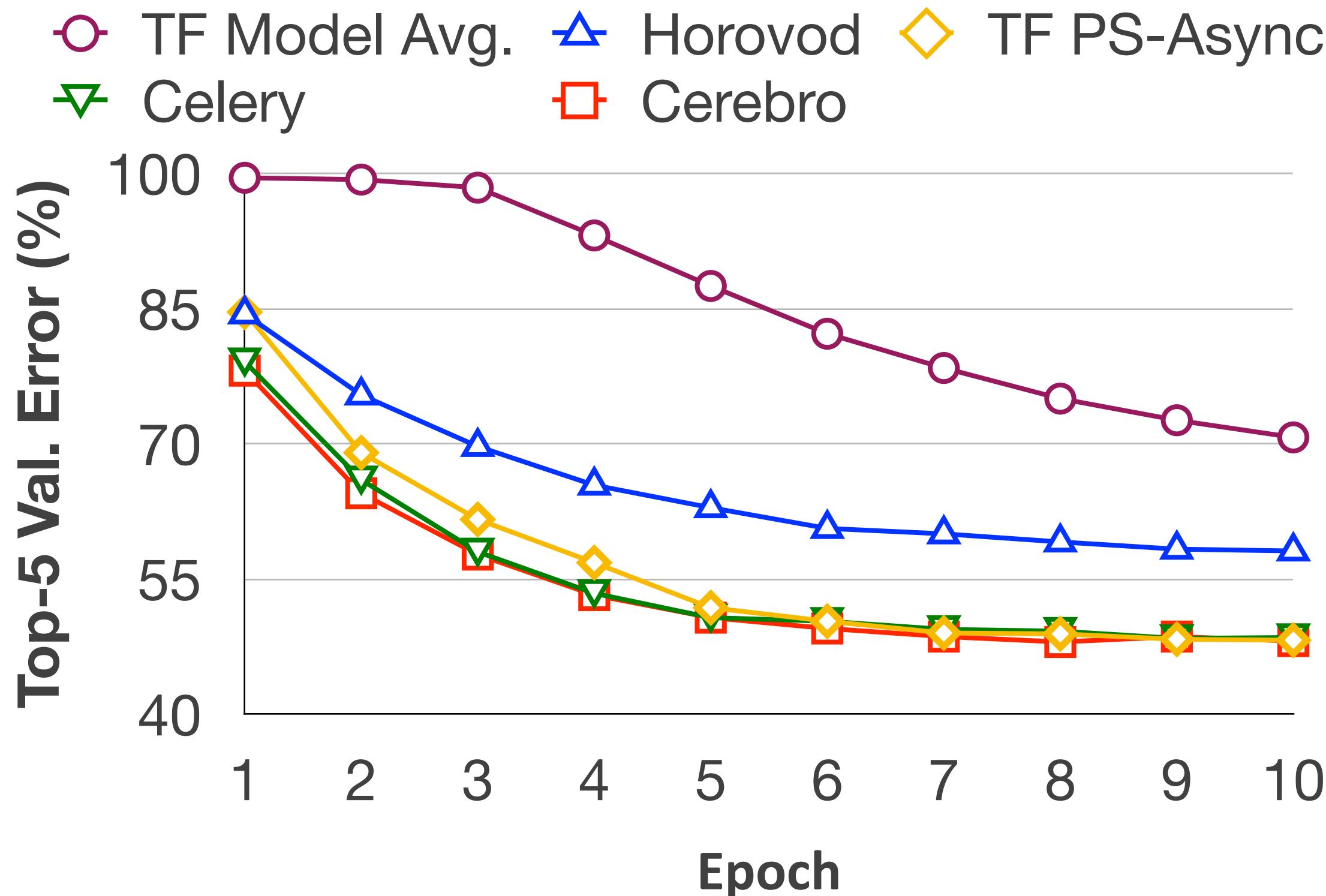
# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



# Experimental Evaluation

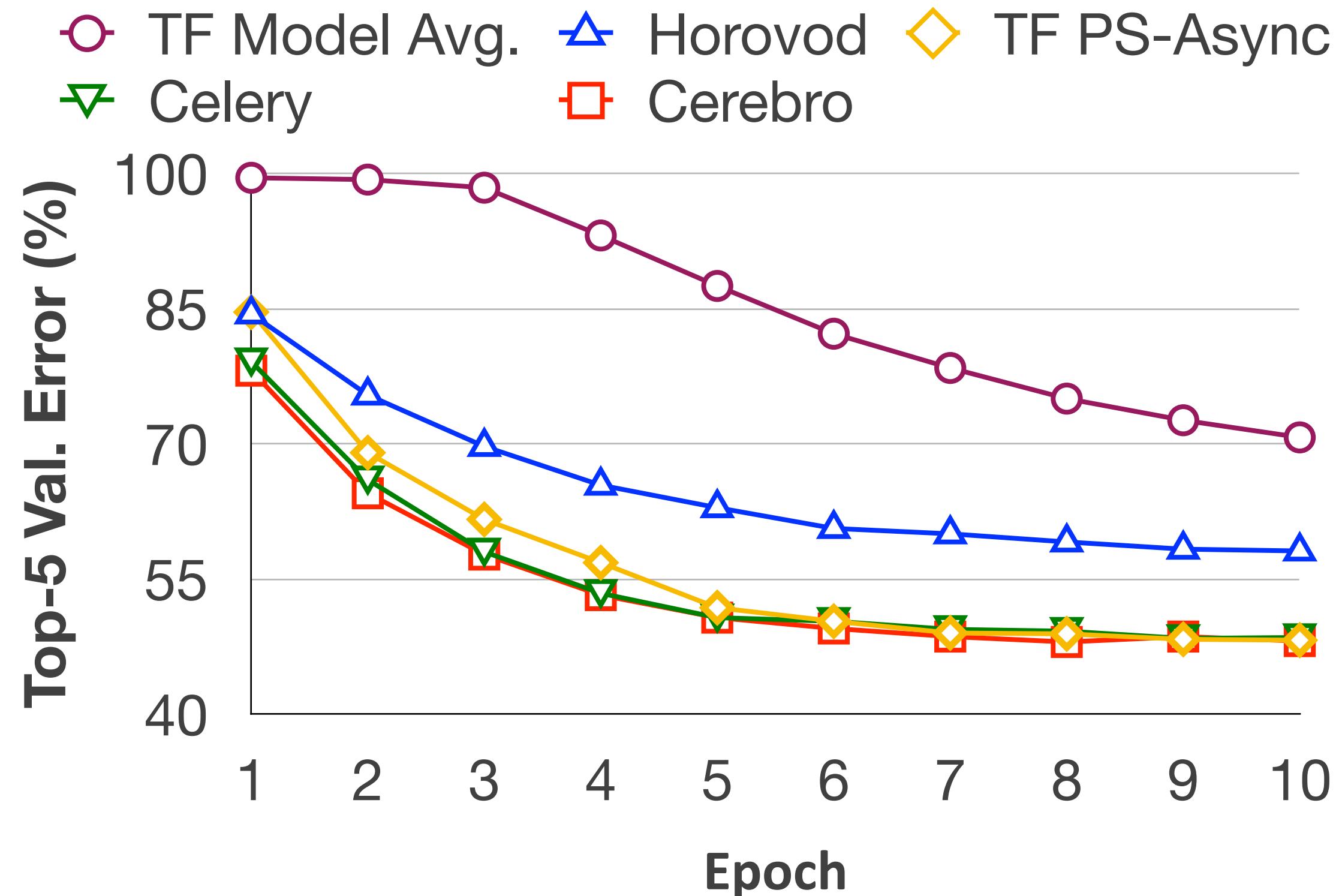
**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes



Method	Runtime (hrs)	Memory
TF PS-Async.	190.0 (!)	200 GB
Horovod	54.2	200 GB
TF Model Averaging	19.7	200 GB
Celery (Task-Parallel)	17.2-19.0*	1600 GB (!)
<b>MOP/Cerebro</b>	<b>17.7</b>	<b>200 GB</b>

# Experimental Evaluation

**Setup:** ImageNet; 16 CNN configurations; TensorFlow; 8 GPU nodes

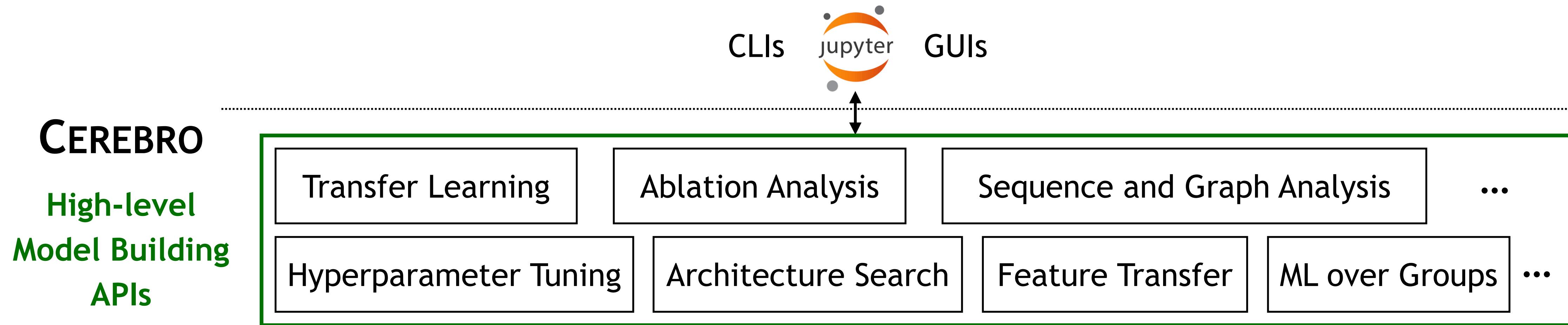


Method	Runtime (hrs)	Memory
TF PS-Async.	190.0 (!)	200 GB
Horovod	54.2	200 GB
TF Model Averaging	19.7	200 GB
Celery (Task-Parallel)	17.2-19.0*	1600 GB (!)
<b>MOP/Cerebro</b>	<b>17.7</b>	<b>200 GB</b>

Cerebro offers overall best combination of resource efficiency and accuracy

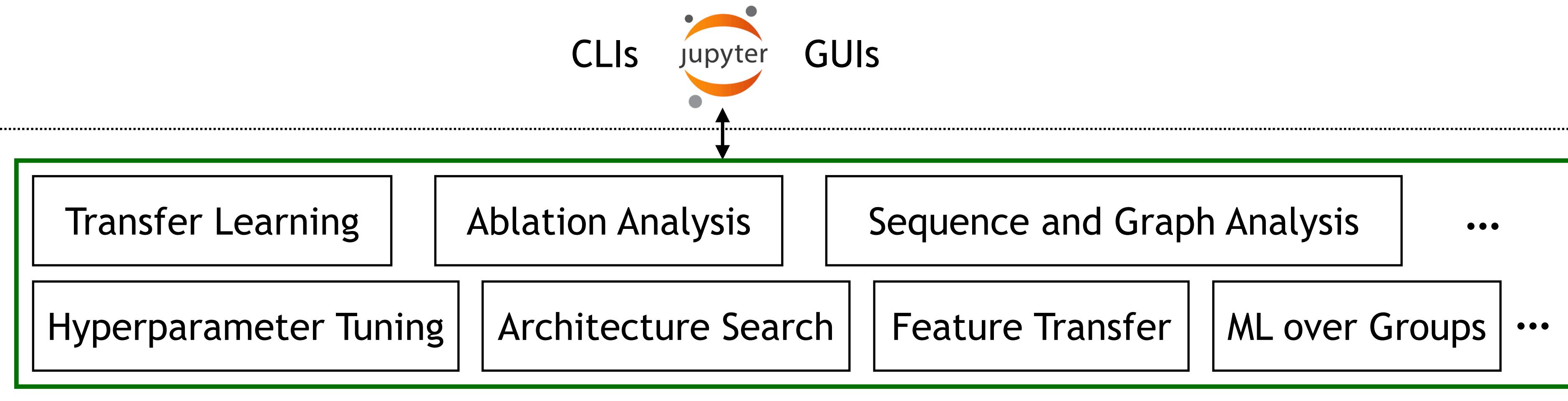
*Cerebro: Efficient and Reproducible Model Selection on Deep Learning Systems. SIGMOD DEEM'19*  
*Cerebro: A Data System for Optimized Deep Learning Model Selection. VLDB'20*

# Full Vision of the Cerebro Platform

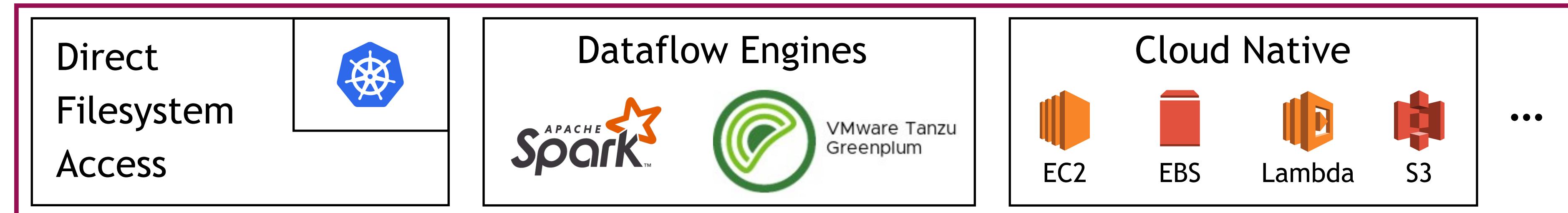


# Full Vision of the Cerebro Platform

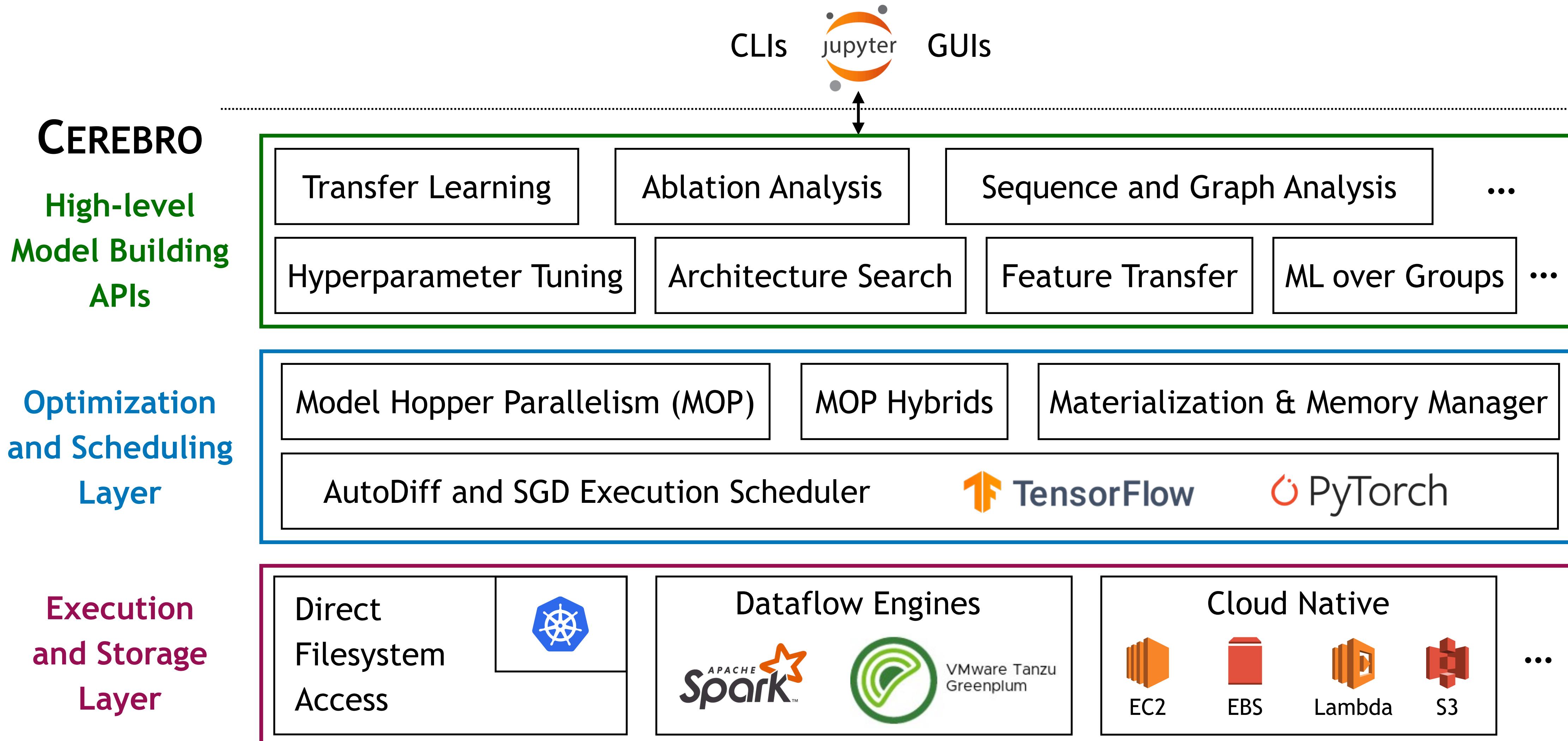
CEREBRO  
High-level  
Model Building  
APIs



Execution  
and Storage  
Layer



# Full Vision of the Cerebro Platform

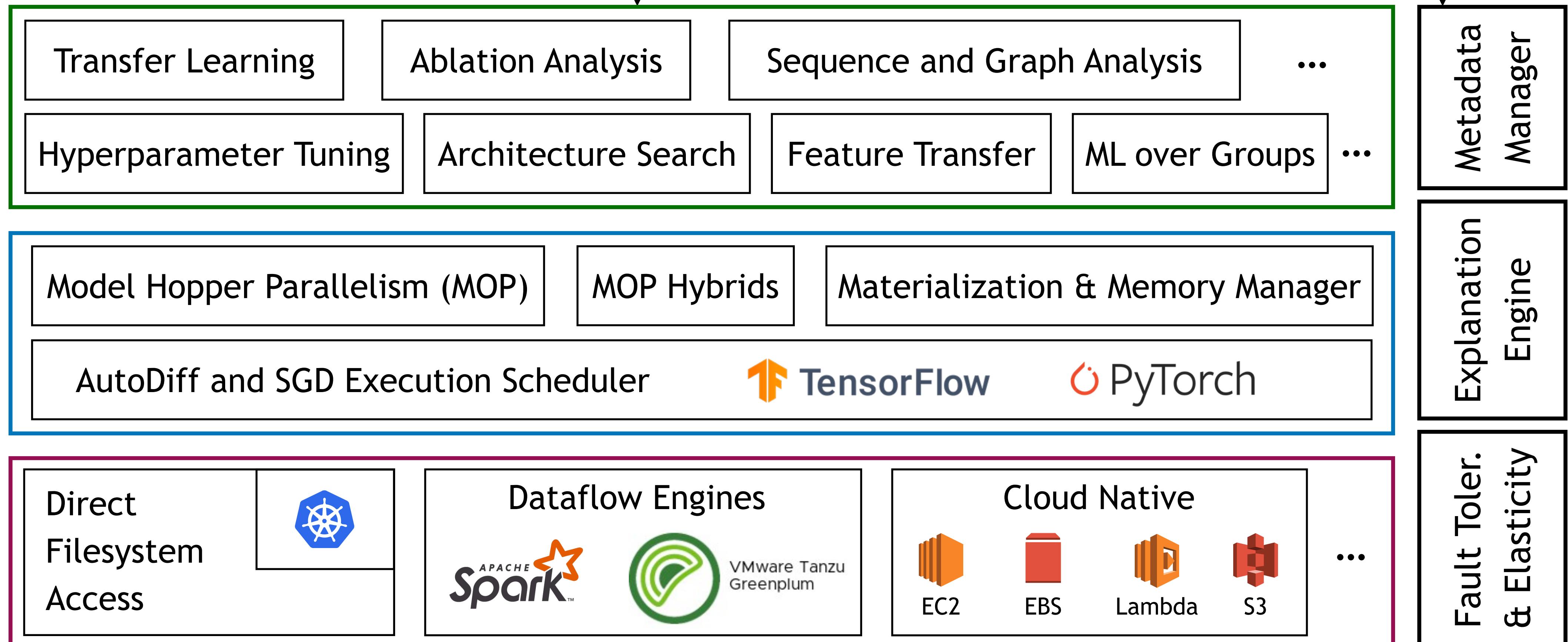
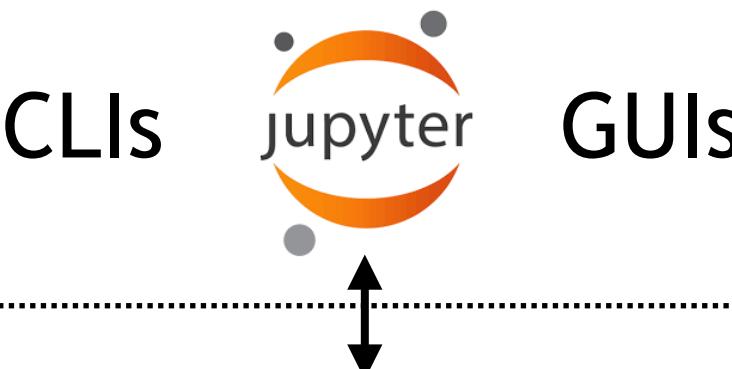


# Full Vision of the Cerebro Platform

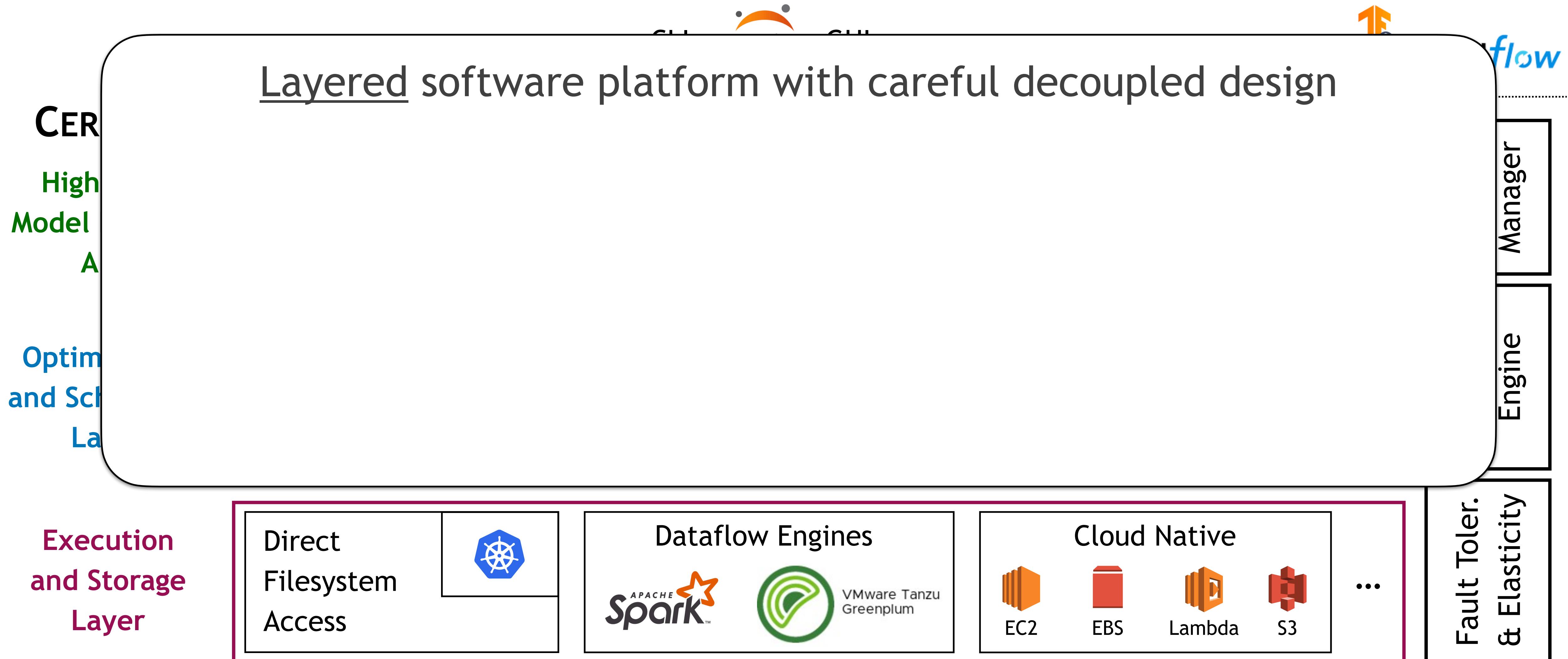
**CERE BRO**  
High-level  
Model Building  
APIs

Optimization  
and Scheduling  
Layer

Execution  
and Storage  
Layer



# Full Vision of the Cerebro Platform



Cerebro: A Layered Data Platform for Scalable Deep Learning. CIDR'21

# Full Vision of the Cerebro Platform

CER  
High Model  
A  
Optim and Sch  
La

Execution  
and Storage  
Layer

Layered software platform with careful decoupled design  
Unified platform for all kinds of data modalities, DL workloads

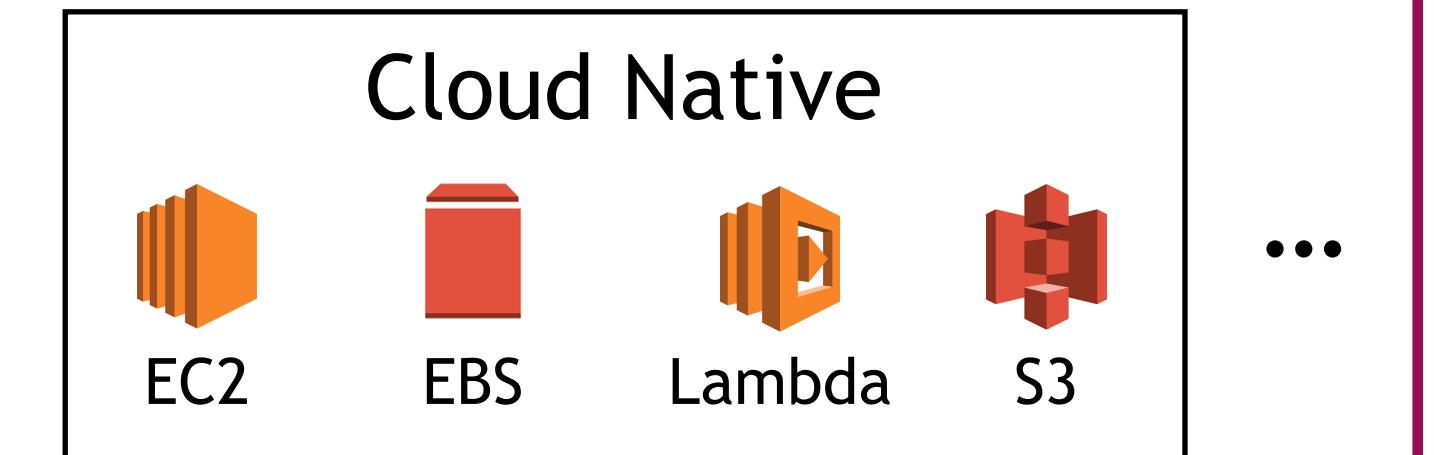
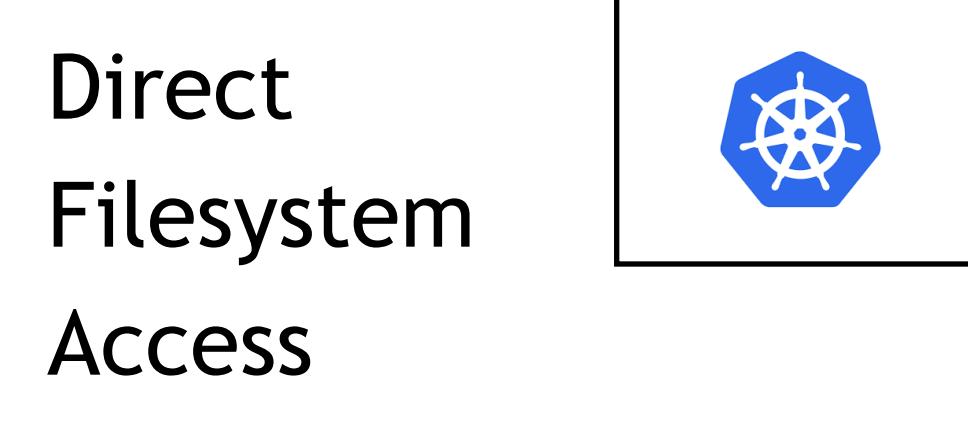


flow

Engine Manager

Engine

Fault Toler.  
& Elasticity



# Full Vision of the Cerebro Platform

CER  
High Model A  
Optim and Sch La

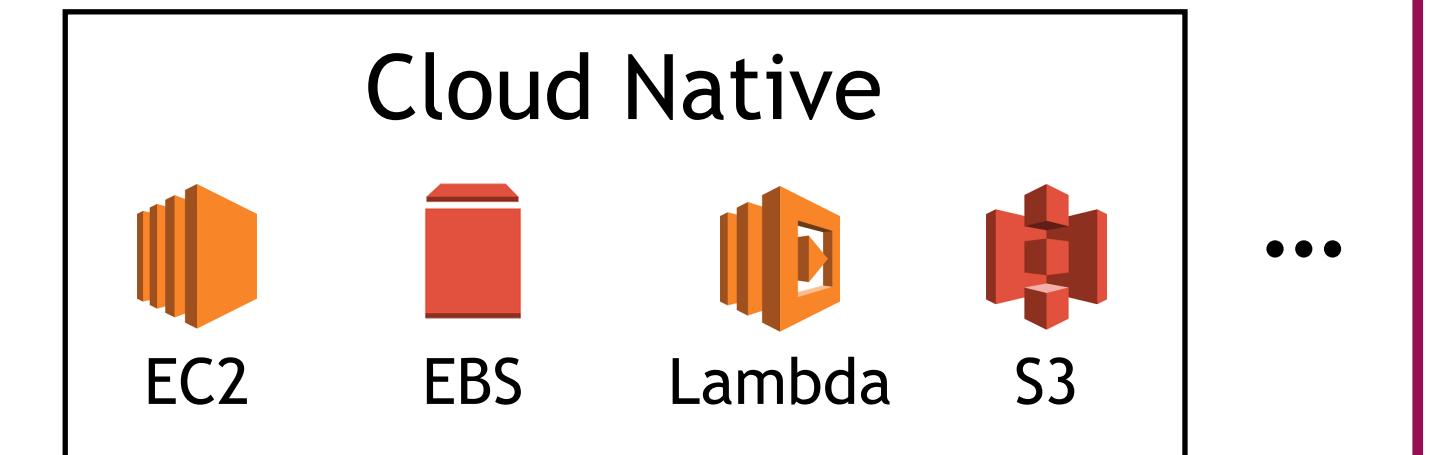
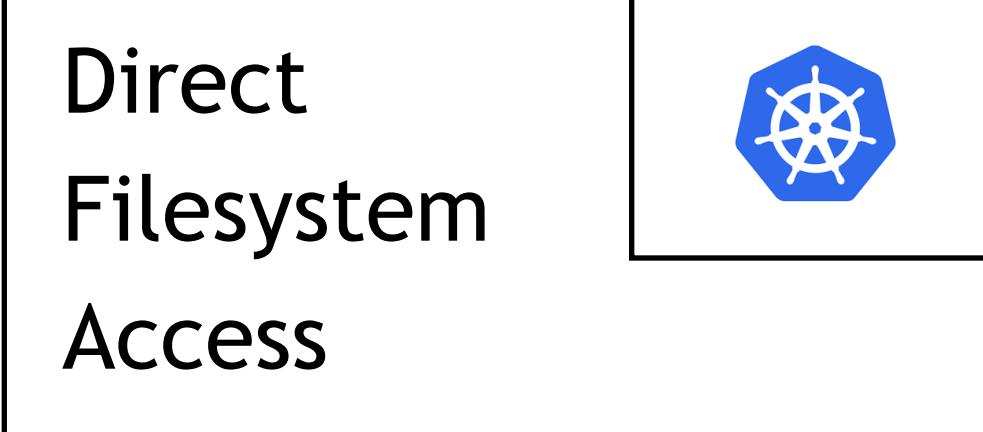
Layered software platform with careful decoupled design  
Unified platform for all kinds of data modalities, DL workloads  
**Productivity:** Focus on what of DL, not how of scaling/optimizing

flow

Manager

Engine

Execution and Storage Layer

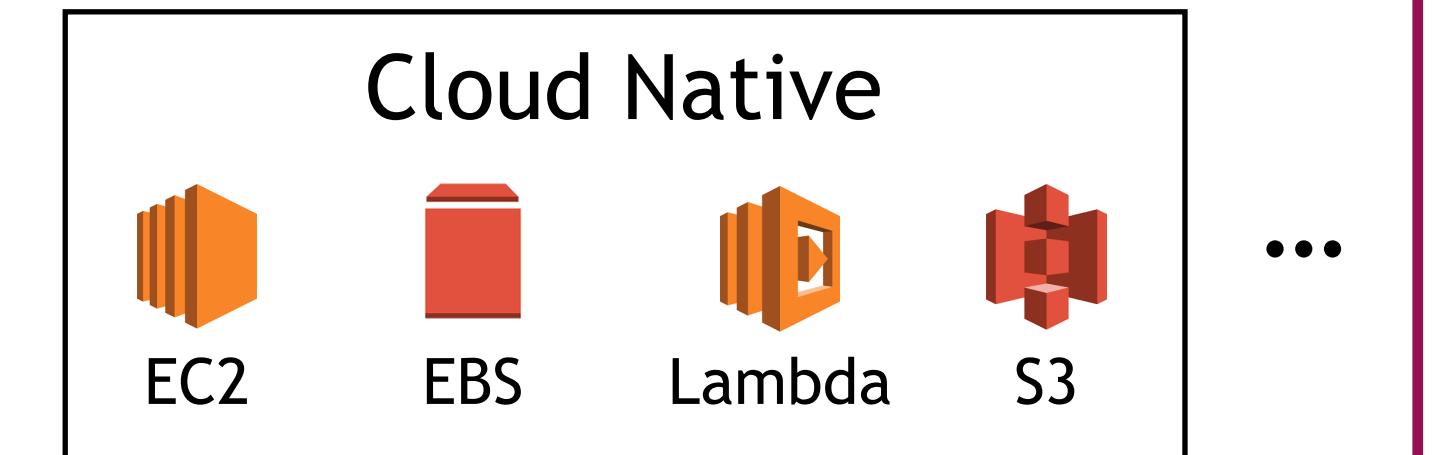
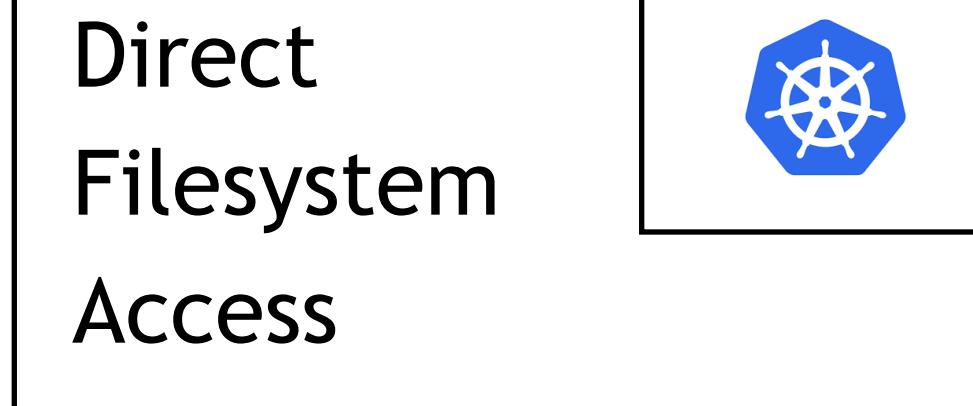


Fault Toler. & Elasticity

# Full Vision of the Cerebro Platform

CER  
High Model A  
Optim and Sch La

Execution and Storage Layer



Fault Toler. & Elasticity



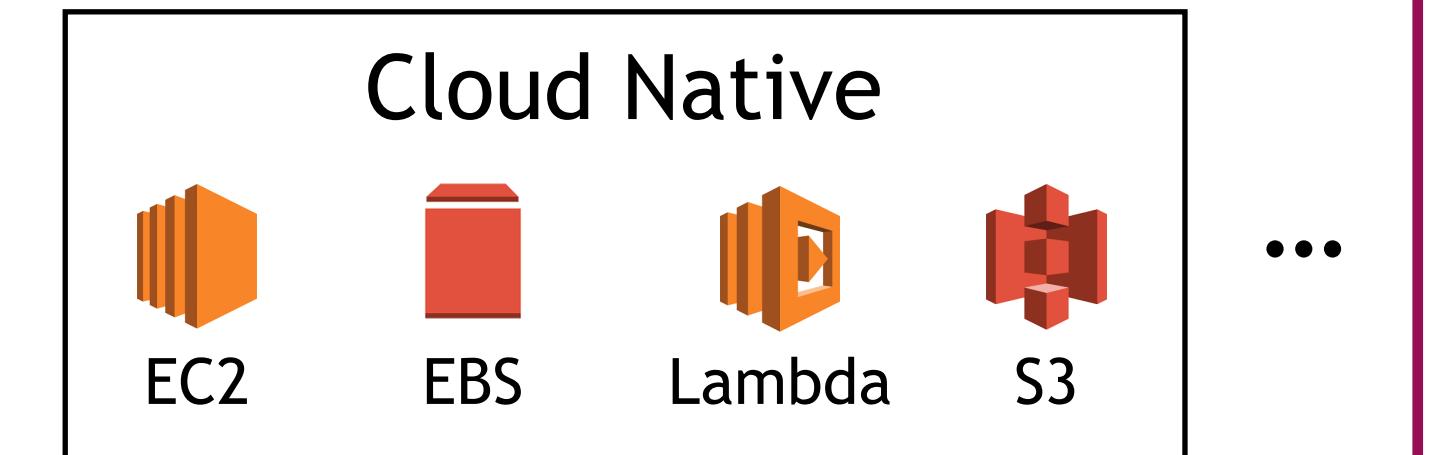
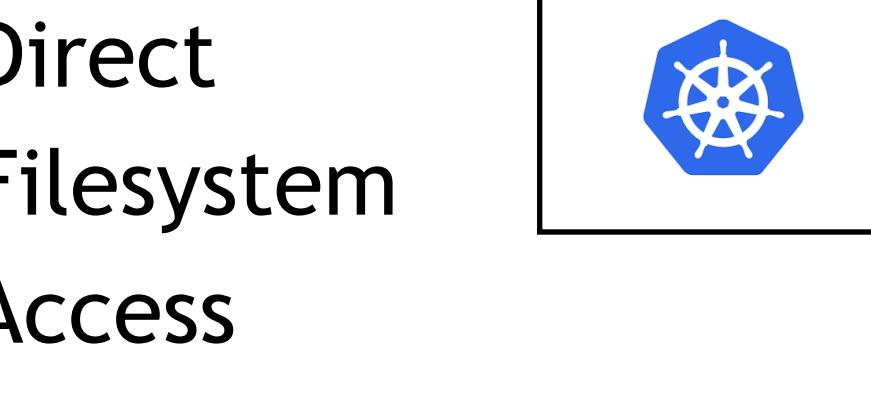
Layered software platform with careful decoupled design

Unified platform for all kinds of data modalities, DL workloads

**Productivity:** Focus on what of DL, not how of scaling/optimizing

**Scalability:** Seamlessly scale to any size of data/models/tasks

Execution and Storage Layer

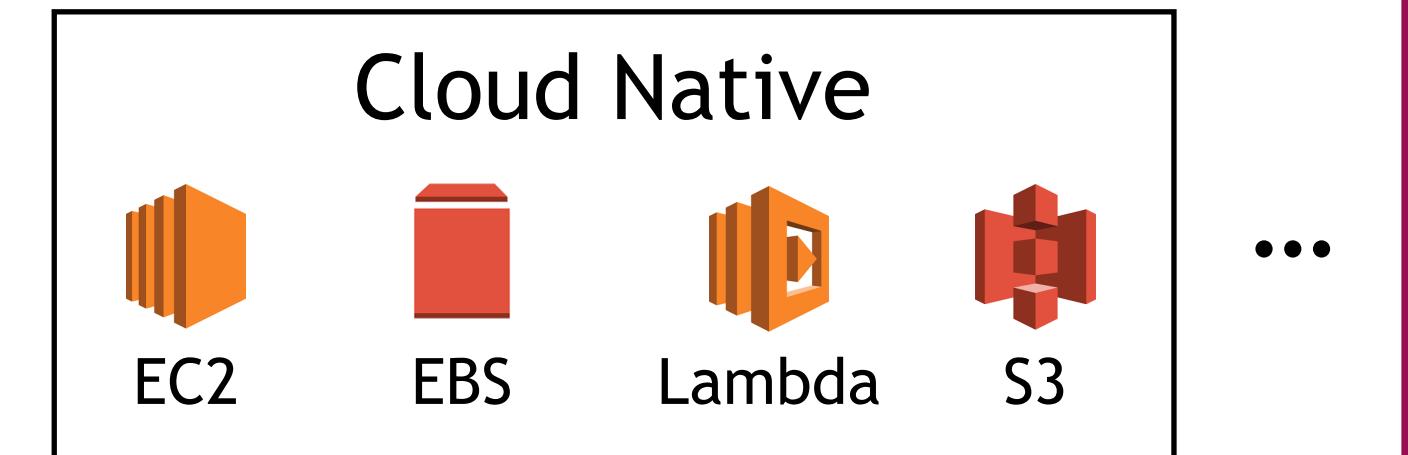
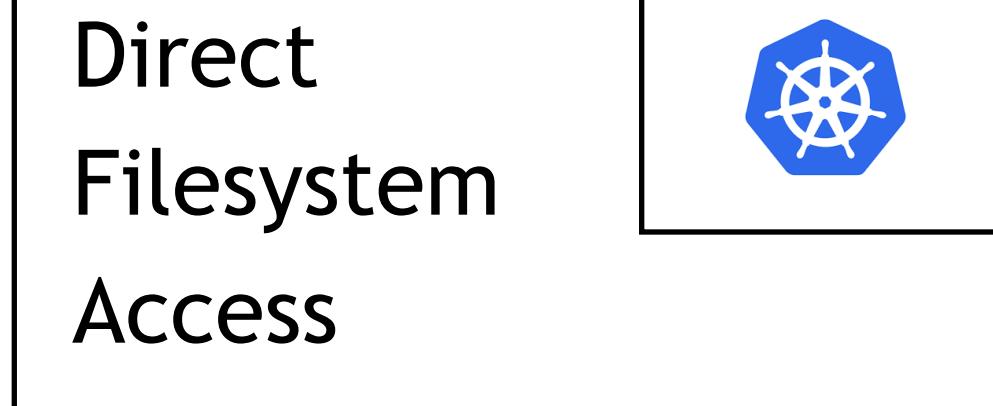


Fault Toler. & Elasticity

# Full Vision of the Cerebro Platform

CER  
High  
Model  
A  
Optim  
and Sch  
La

Execution  
and Storage  
Layer



Fault Toler.  
& Elasticity



Layered software platform with careful decoupled design

Unified platform for all kinds of data modalities, DL workloads

**Productivity:** Focus on what of DL, not how of scaling/optimizing

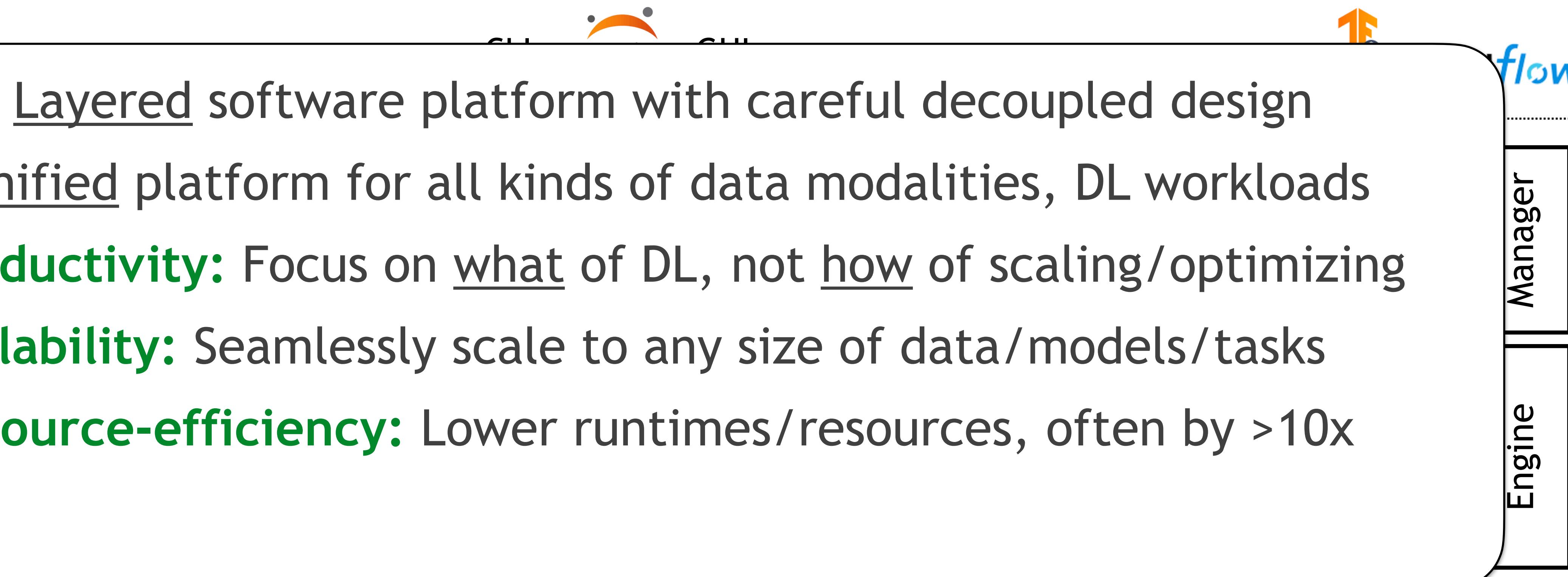
**Scalability:** Seamlessly scale to any size of data/models/tasks

**Resource-efficiency:** Lower runtimes/resources, often by >10x

flow

Manager

Engine

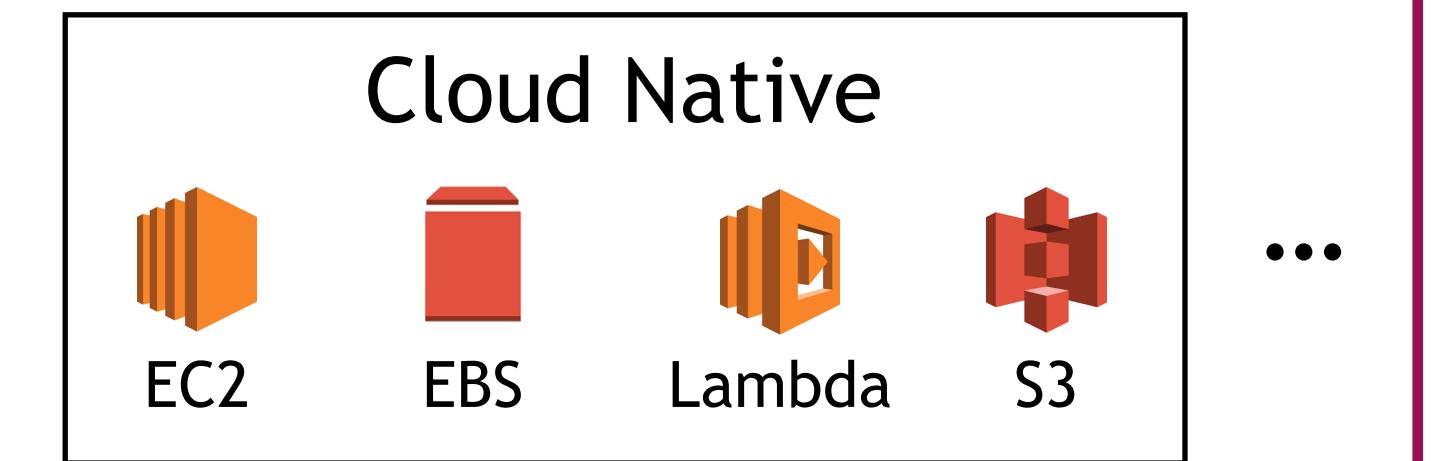
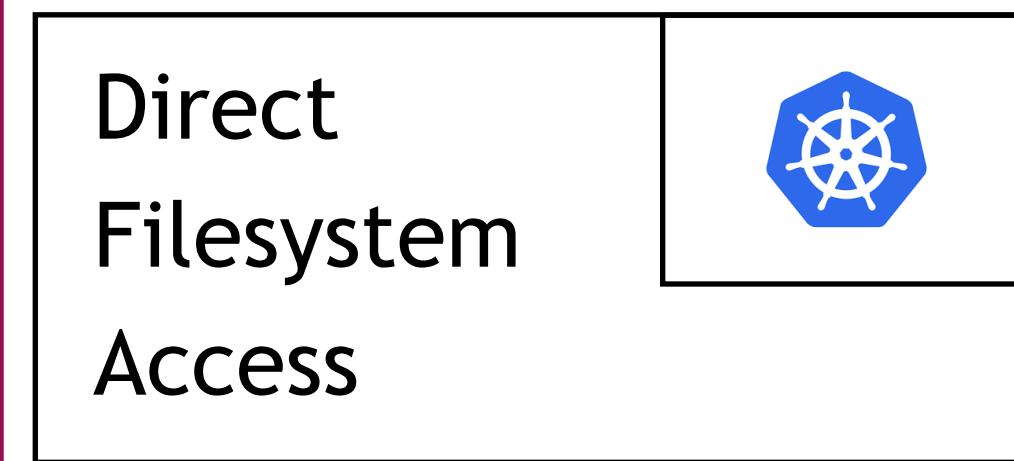


# Full Vision of the Cerebro Platform

CER  
High  
Model  
A  
Optim  
and Sch  
La

Layered software platform with careful decoupled design  
Unified platform for all kinds of data modalities, DL workloads  
**Productivity:** Focus on what of DL, not how of scaling/optimizing  
**Scalability:** Seamlessly scale to any size of data/models/tasks  
**Resource-efficiency:** Lower runtimes/resources, often by >10x  
**Portability:** Any popular storage/execution backend; any DL tool

Execution  
and Storage  
Layer



Fault Toler.  
& Elasticity

# Cerebro: Early Impact and Trajectory

Used on TBs by UCSD Public Health scientists; Physics & Materials Eng. next  
Shipped with Apache MADlib by Pivotal; VMware collaboration  
Cerebro+Spark released; all open sourced under Apache License v2.0  
Coming soon: More workloads + optimizations + domain science use cases

# Cerebro: Early Impact and Trajectory

Used on TBs by UCSD Public Health scientists; Physics & Materials Eng. next  
Shipped with Apache MADlib by Pivotal; VMware collaboration  
Cerebro+Spark released; all open sourced under Apache License v2.0  
Coming soon: More workloads + optimizations + domain science use cases



# Cerebro: Open Source Release and Demo

## Table of Contents

[What is Cerebro?](#)

[Install](#)

[Getting Started](#)

[Cerebro API](#)

[Acknowledgement](#)

## Quick search

Go

## Cerebro Documentation

Cerebro is a data system for optimized deep learning model selection. It uses a novel parallel execution strategy called **Model Hopper Parallelism (MOP)** to execute end-to-end deep learning model selection workloads in a more resource-efficient manner.

## Installation

The best way to install the Cerebro is via pip.

```
pip install -U cerebro-dl
```

Alternatively, you can git clone and run the provided Makefile script to install the master branch

```
git clone https://github.com/ADALabUCSD/cerebro-system.git && cd cerebro-system && make install
```

### Note:

You MUST be running on **Python >= 3.6** with **Tensorflow >= 2.2** and **Apache Spark >= 2.4**

## Table of Contents

Links on project webpage: <https://adalabucsd.github.io/cerebro.html>

# Cerebro: Open Source Release and Demo

## Table of Contents

### Table of Contents

#### What is Cerebro

[What is Cerebro?](#)

#### Install

[Install](#)

#### Getting Started

[Getting Started](#)

- Executing a Model Selection Workload

- Visualizing the Model Selection Process

- Training on Existing Petastorm Datasets

- End-to-End Example

[Cerebro API](#)

[Acknowledgement](#)

### Quick

#### Quick search

Go

## Getting Started

### Executing a Model Selection Workload

Cerebro allows you to perform model selection of your deep neural network directly on an existing Spark DataFrame, leveraging Spark's ability to scale across multiple workers:

```
from cerebro.backend import SparkBackend
from cerebro.keras import SparkEstimator

# data storage for intermediate data and model artifacts.
from cerebro.storage import LocalStore, HDFSStore

# Model selection/AutoML methods.
from cerebro.tune import GridSearch, RandomSearch, TPESearch

# Utility functions for specifying the search space.
from cerebro.tune import hp_choice, hp_uniform, hp_quniform, hp_loguniform

import tensorflow as tf
from pyspark.sql import SparkSession

spark = SparkSession \
    .builder \
    .appName("Cerebro Example") \
    .getOrCreate()

...
backend = SparkBackend(spark_context=spark.sparkContext, num_workers=2)
store = LocalStore(prefix_path='/user/username/experiments')

# Initialize input DataFrames.
# You can download sample dataset from https://apache.googlesource.com/spark/
df = spark.read.format("libsvm").load("sample_libsvm_data.txt").repartition(2)
train_df, test_df = df.randomSplit([0.8, 0.2])
```

Links on project webpage: <https://adalabucsd.github.io/cerebro.html>

# Cerebro: Open Source Release and Demo

**(A) Intermittent Human-in-the-Loop Cerebro**

Upload a model script file OR choose a popular model

Drag and Drop or Select a File

Select a popular model:

ResNet-50

**Select this model**

---

**(B) Setup experiment**

Name of experiment (string type)  
Description (optional)  
Name of features (string type, comma separated)  
Name of label (string type)  
Maximum training epochs (integer type)  
Data store prefix path (string type)  
Estimator function name (<module\_name>:<function\_name>)  
Hyperparameters search strategy (please select)

ImageNet
description:(string)
image
label
10
hdfs://storage_server/imagenet
ImageNet_model_gen_script:est
GridSearch

Hyperparameter grid table:

Name	Hyperparameter Type	Choices	Min Value	Max Value	Quantum	Data Type
learning_rate	quantum log uniform (base 10)		-6	-4	2	float
batch_size	categorical	32, 256				integer
l2_reg	quantum log uniform (base 10)		-6	-4	2	float
model	categorical	ResNet50, VGG16				string

Add New Parameter(One row)



**(D) Models**

Input model name and operate on the

Model Name

Resume, Delete, Stop, Clone

Model Name	Experiment Name	Parent
filter data...		/
Michael	ResNet-Example	/
Christopher	ResNet-Example	/

**(E) Clone Model**

Base Model Name: Michael

Warm Start on Base Model     Start From Scratch

Name	Hyperparameter Type	Choices	Min Value	Max Value	Quantum	Data Type
Learning Rate	log uniform	None	-3	-1	0.5	float
Batch Size	quantum uniform	None	64	64	0	integer
Drop Out	uniform	None	0.1	0.1	None	float

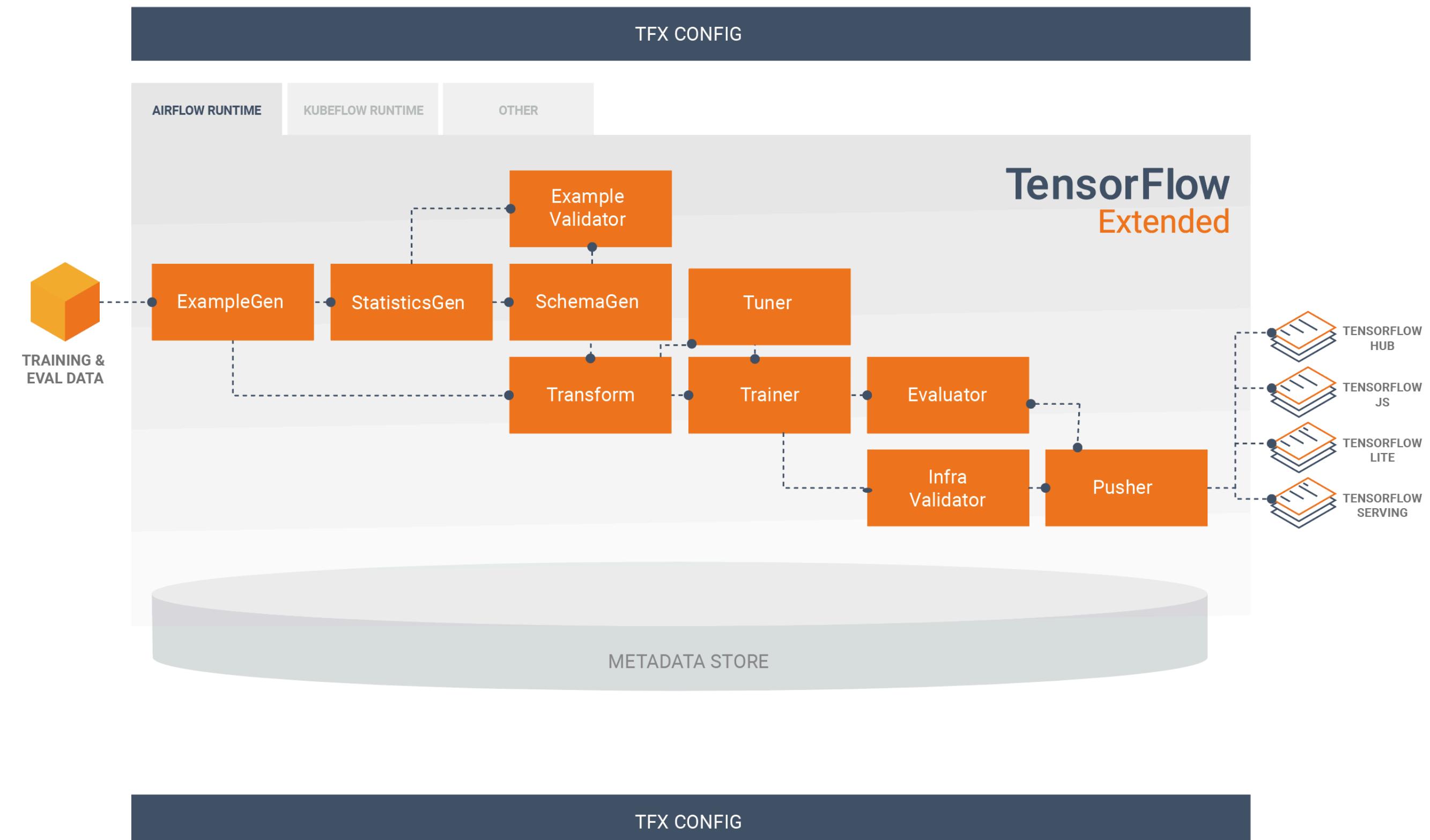
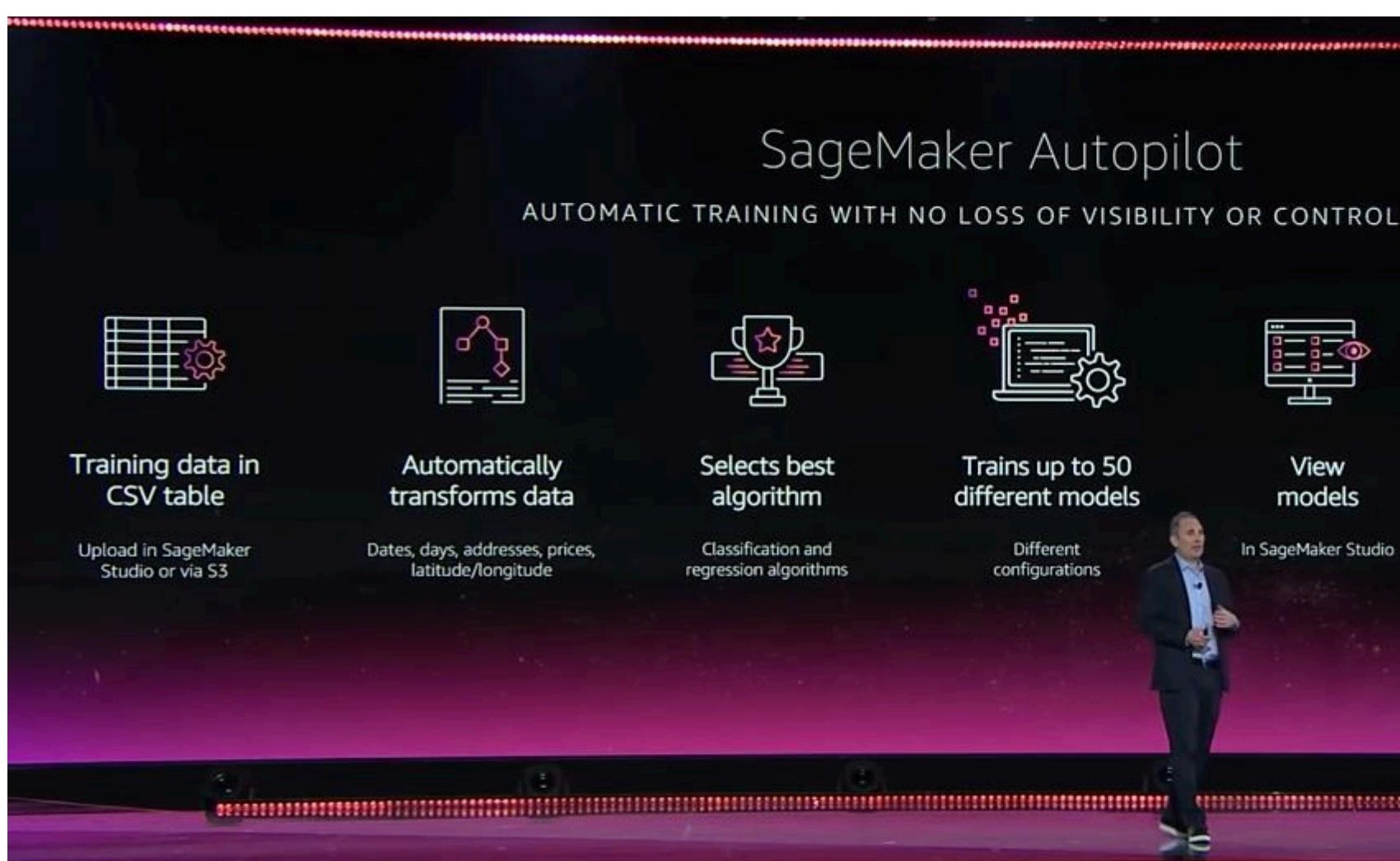
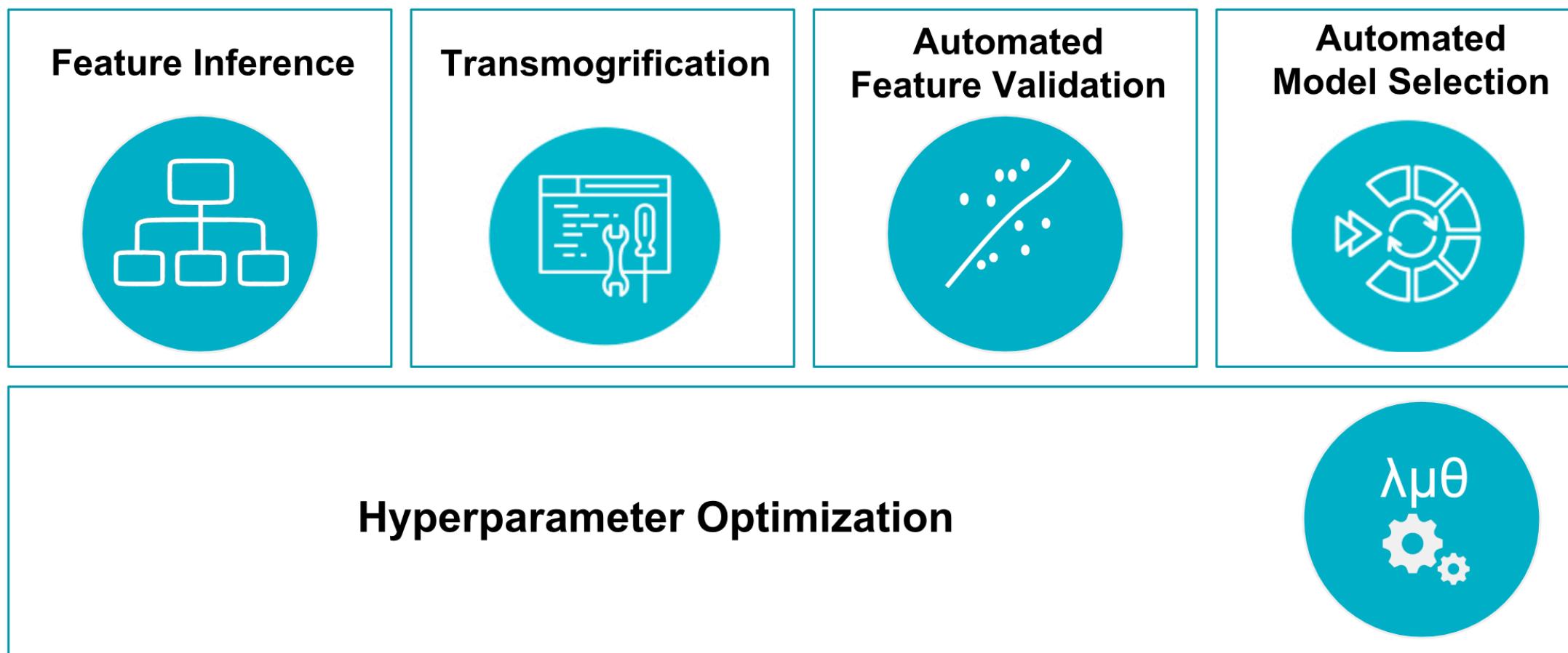
Links on project webpage: <https://adalabucsd.github.io/cerebro.html>

# Outline

- | The New DBfication of ML/AI
- | Two Examples from My Research
  - | Example 1: Scalable DL Systems
  - | Example 2: Auto Data Prep for ML
- | Accelerating the DBfication of ML/AI

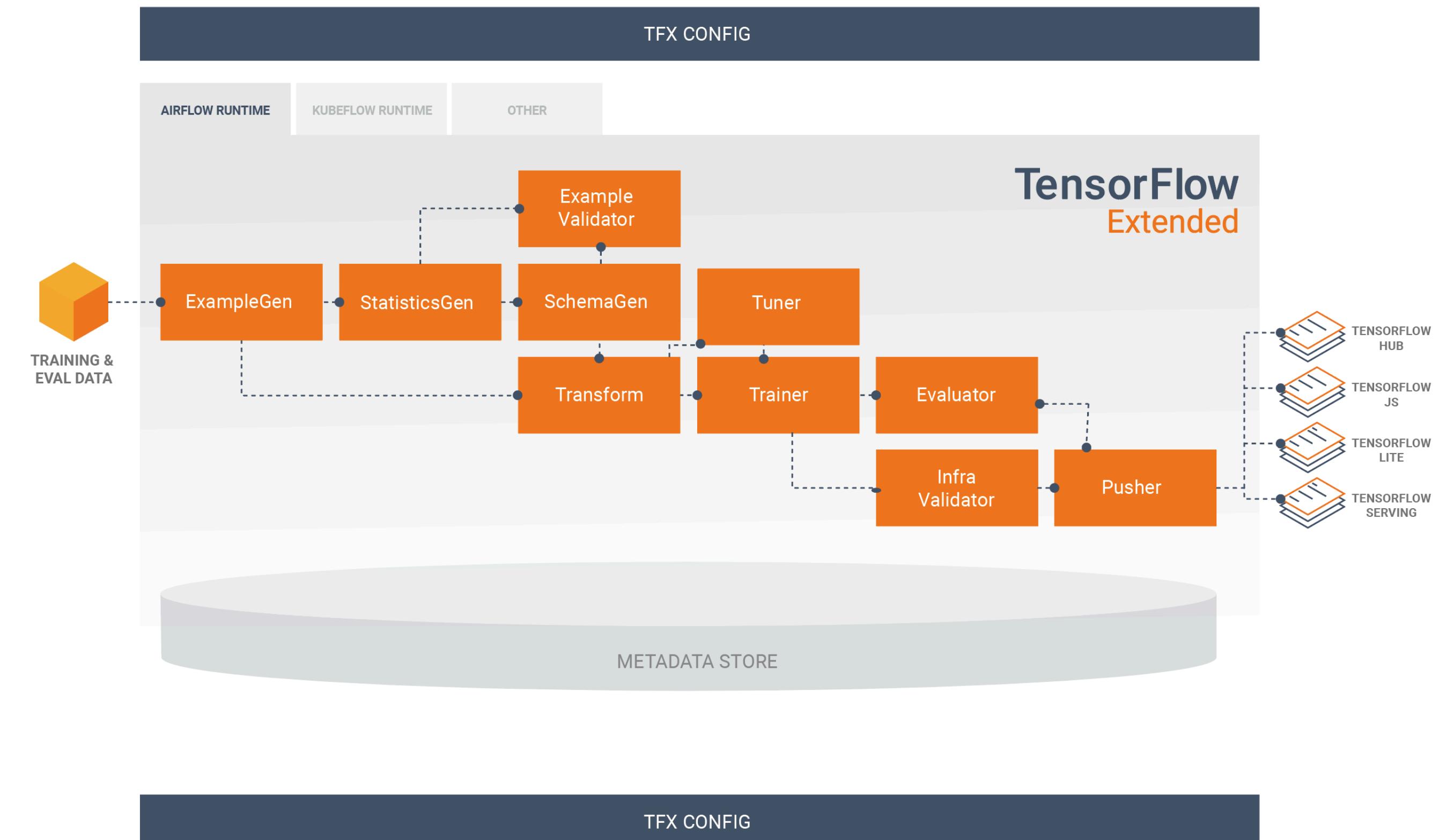
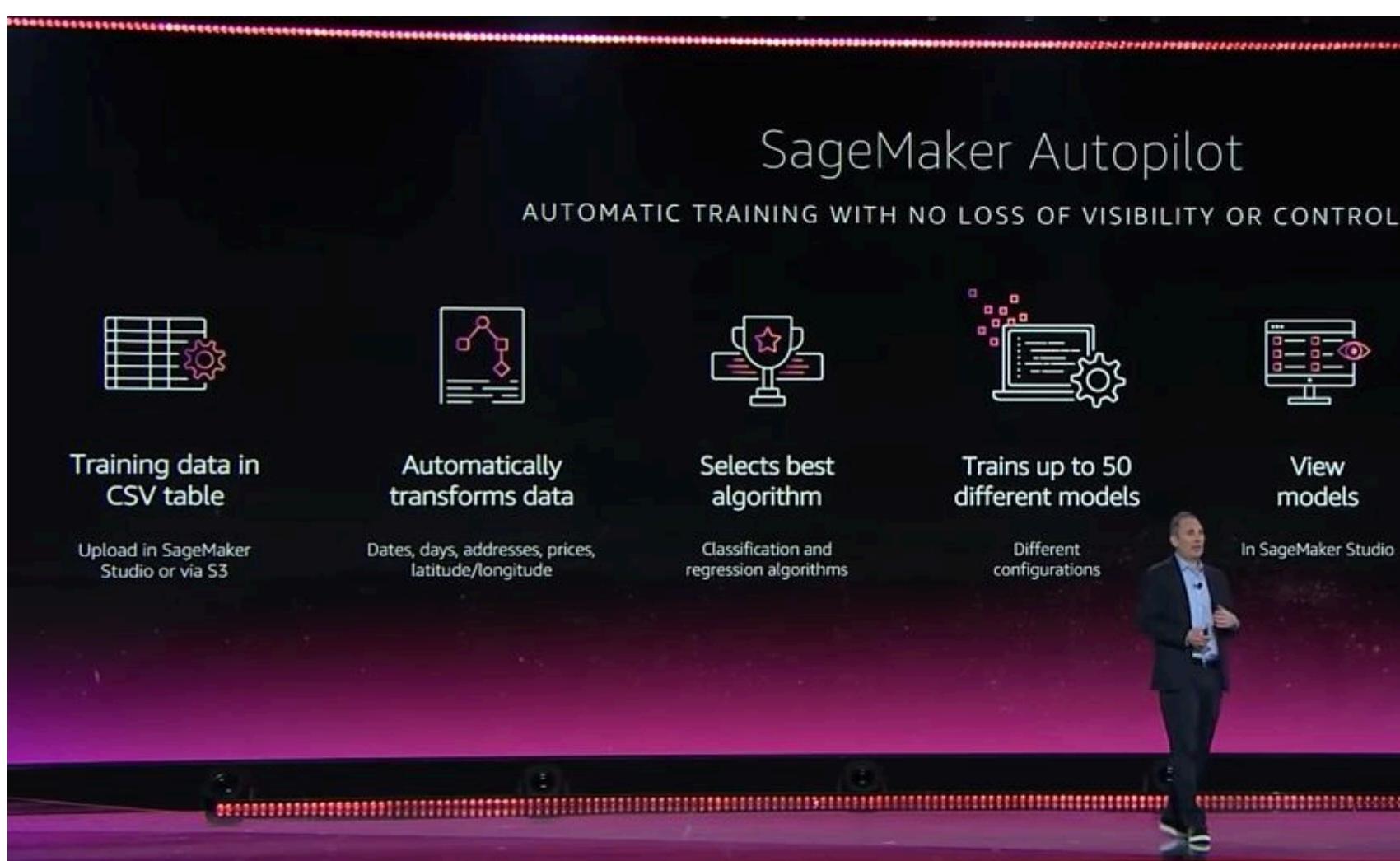
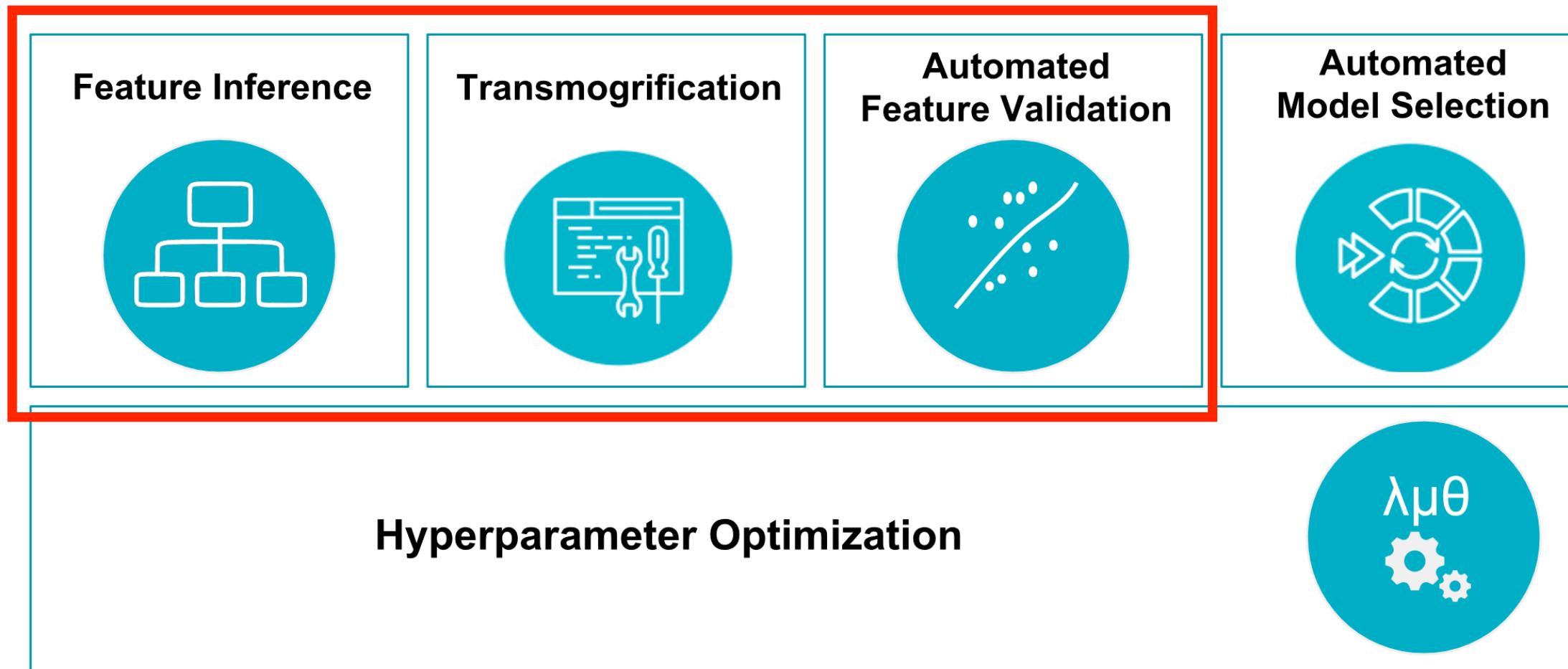
# Example 2: Auto Data Prep for ML

## TransmogrifAI



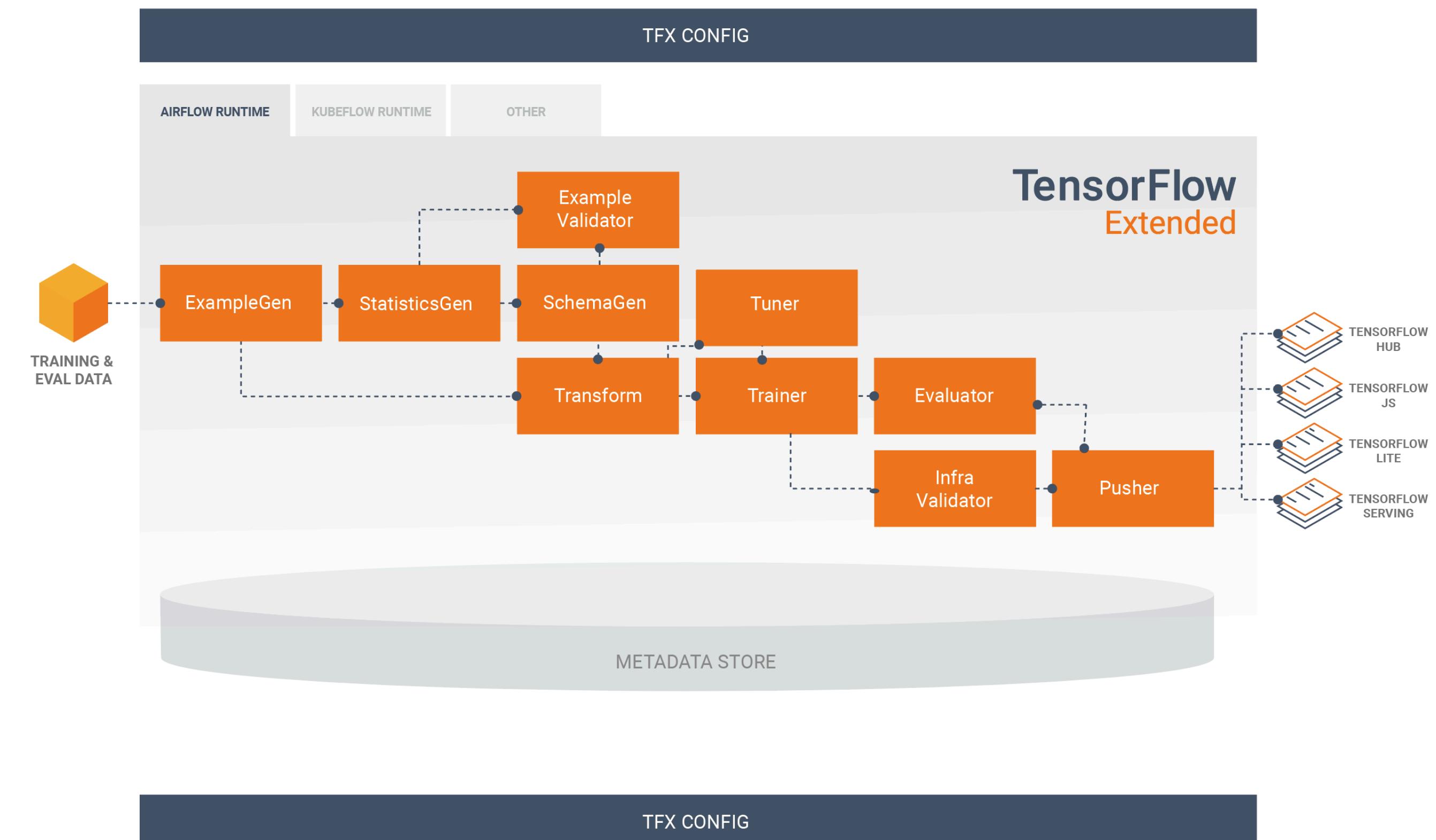
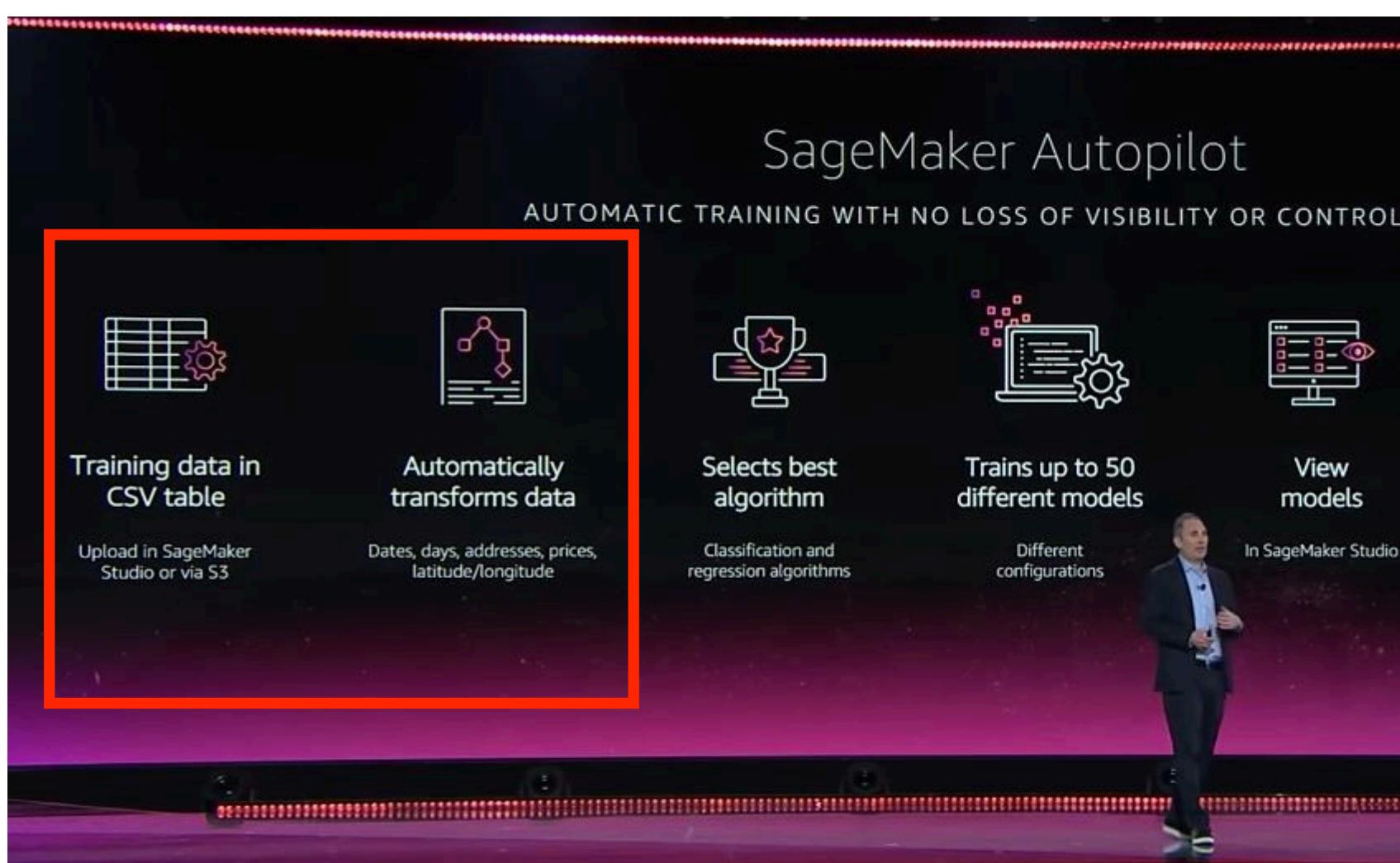
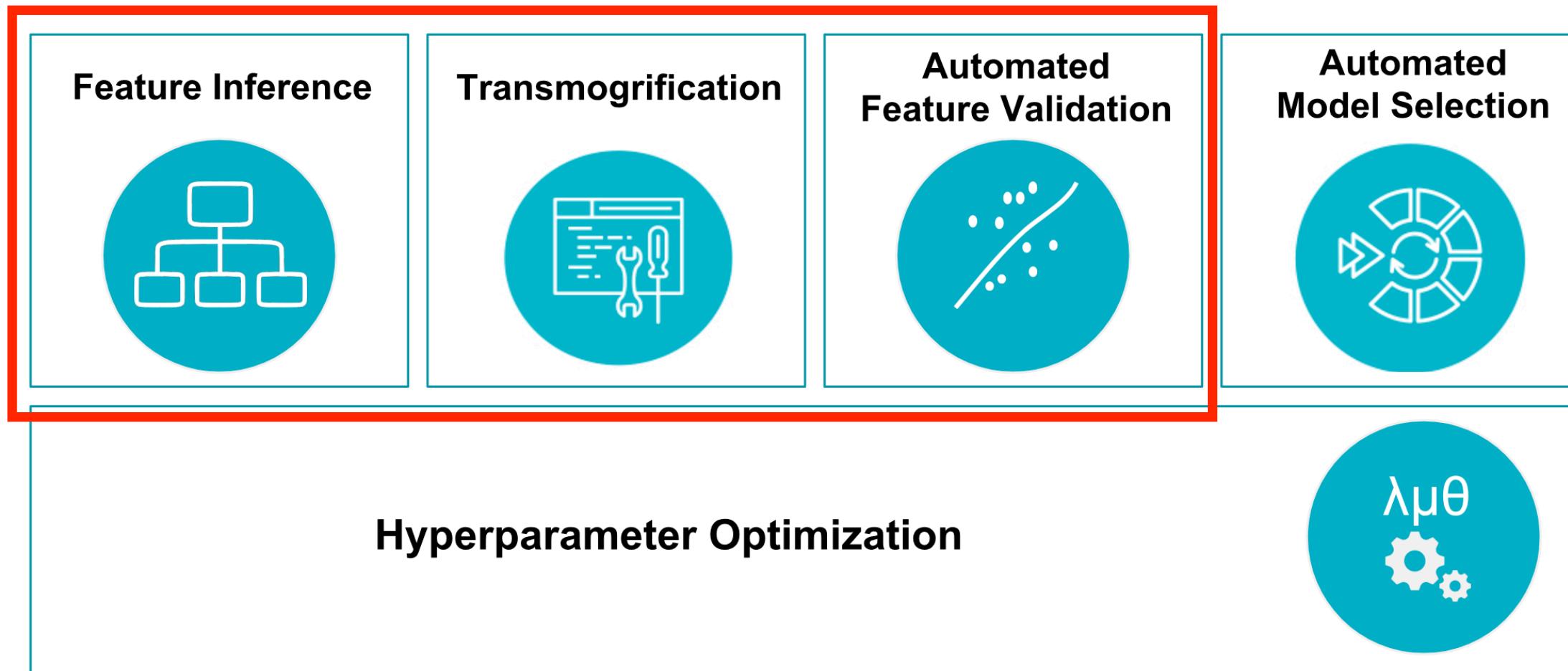
# Example 2: Auto Data Prep for ML

## TransmogrifAI



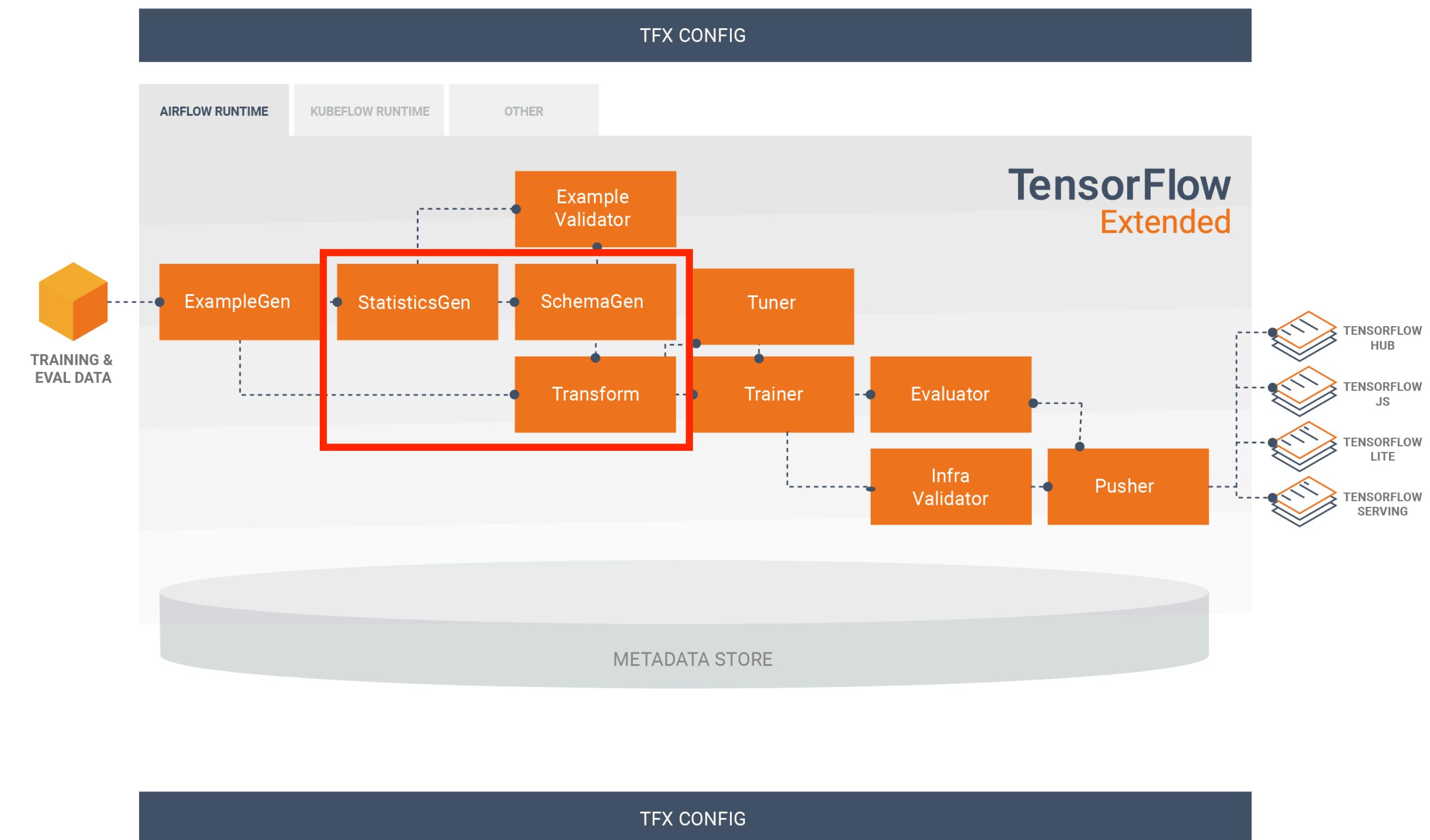
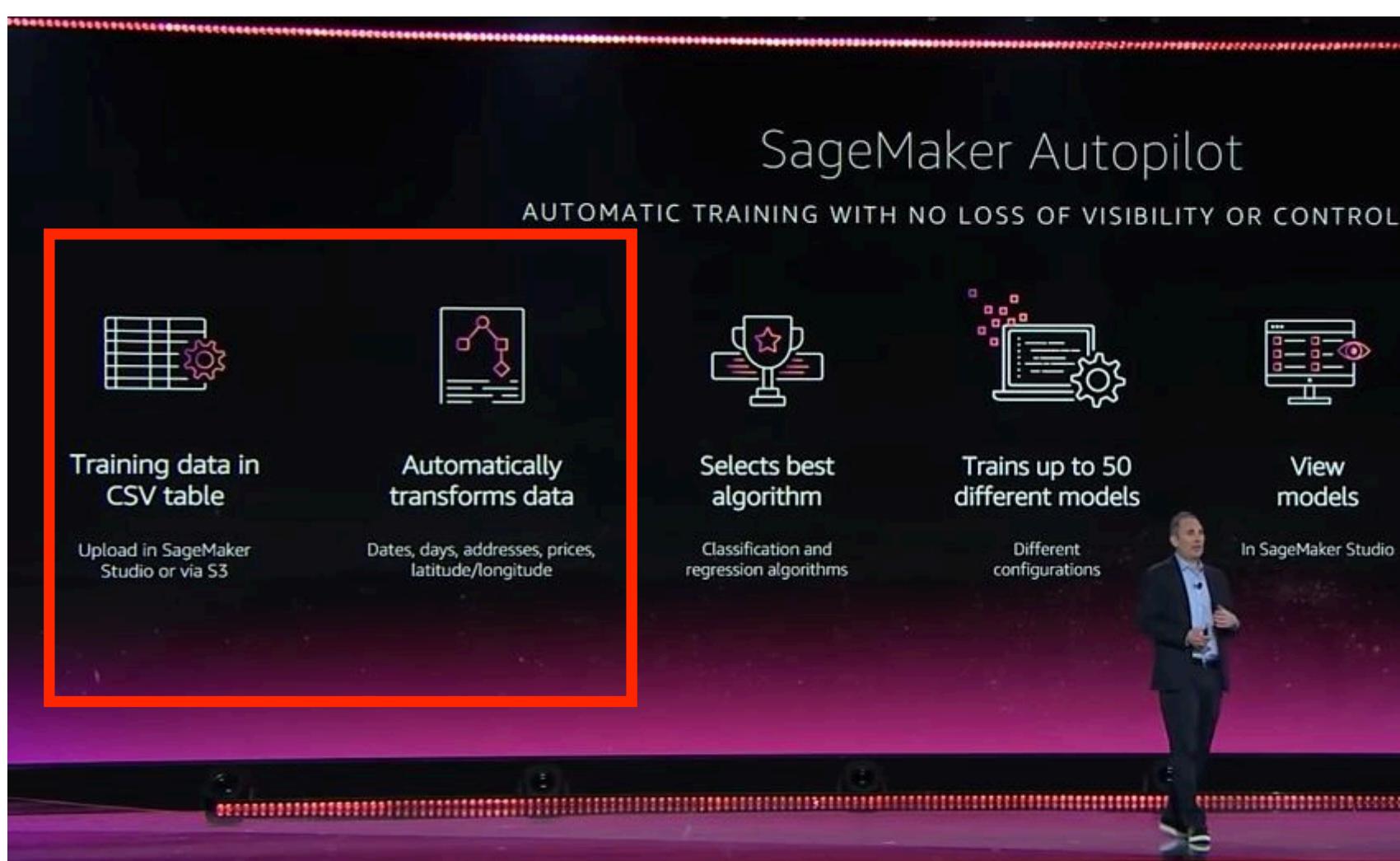
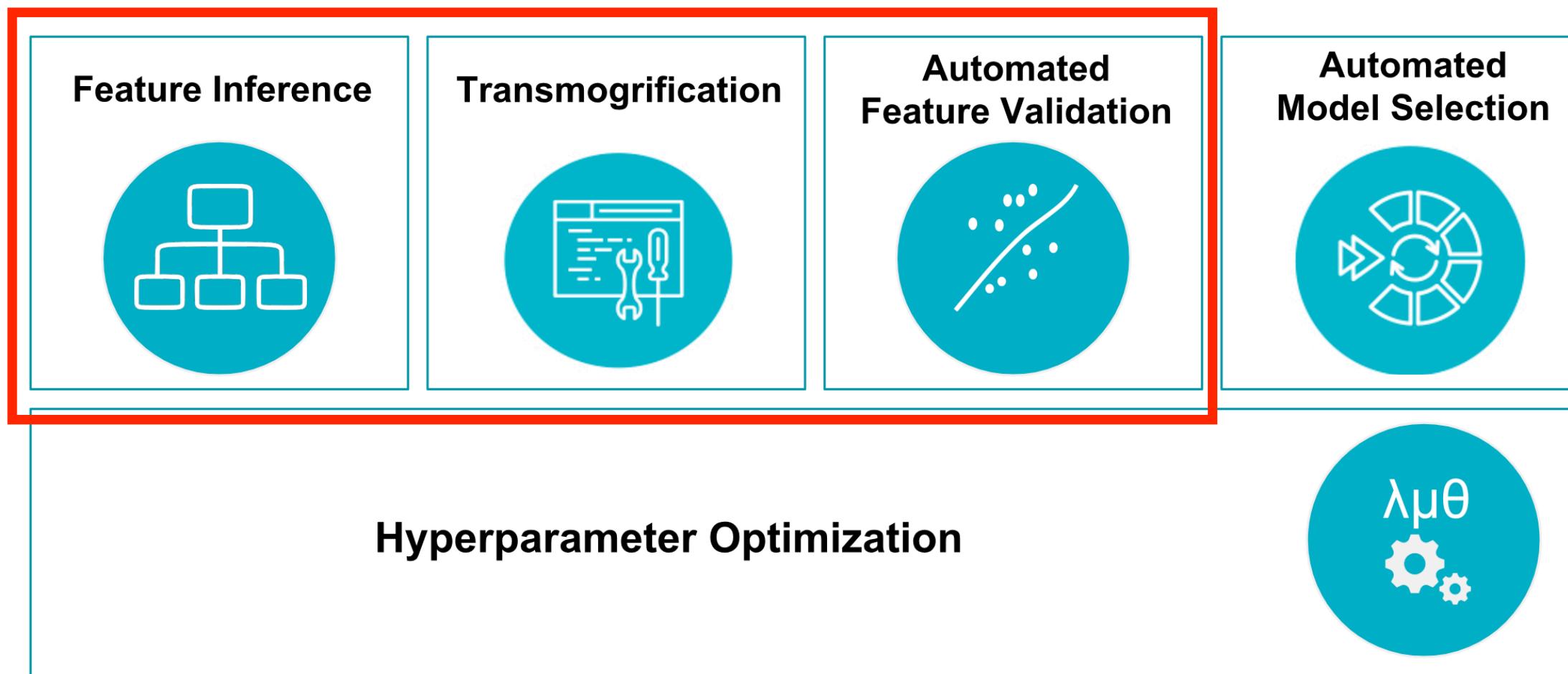
# Example 2: Auto Data Prep for ML

## TransmogrifAI



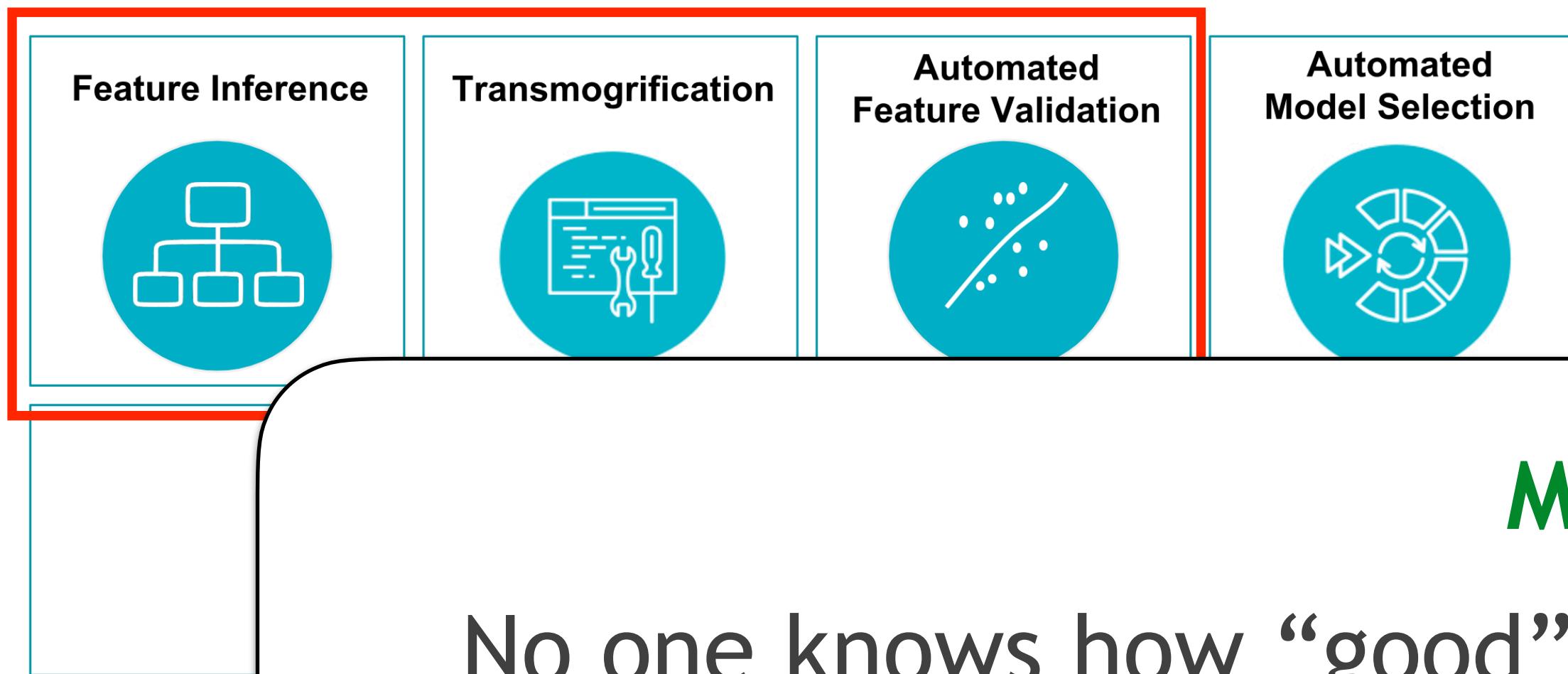
# Example 2: Auto Data Prep for ML

## TransmogrifAI



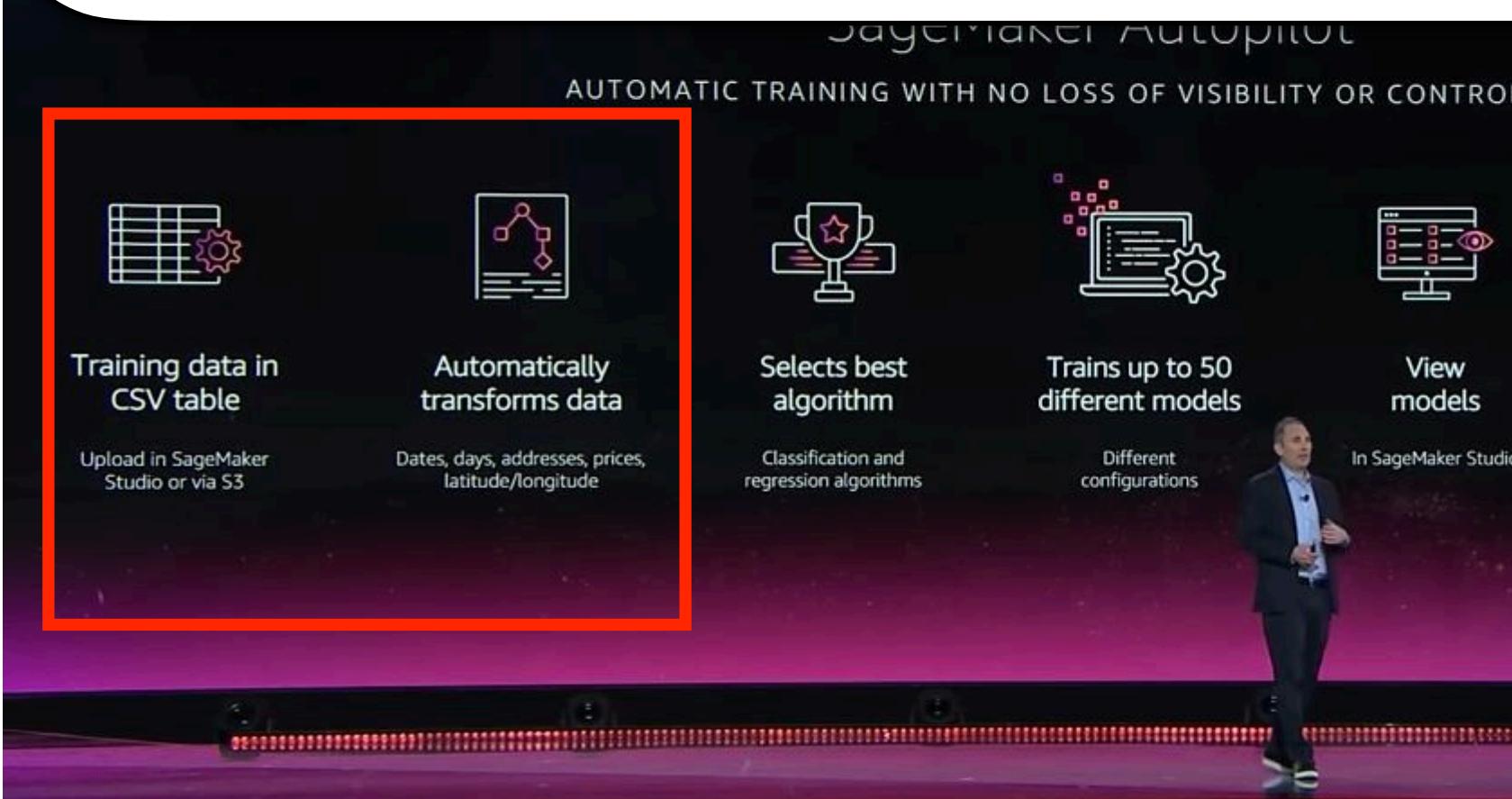
# Example 2: Auto Data Prep for ML

TransmogrifAI



**Major Issue:**

No one knows how “good” objectively such auto data prep is!



# Project SortingHat

Benchmarks for  
ML Data Prep =

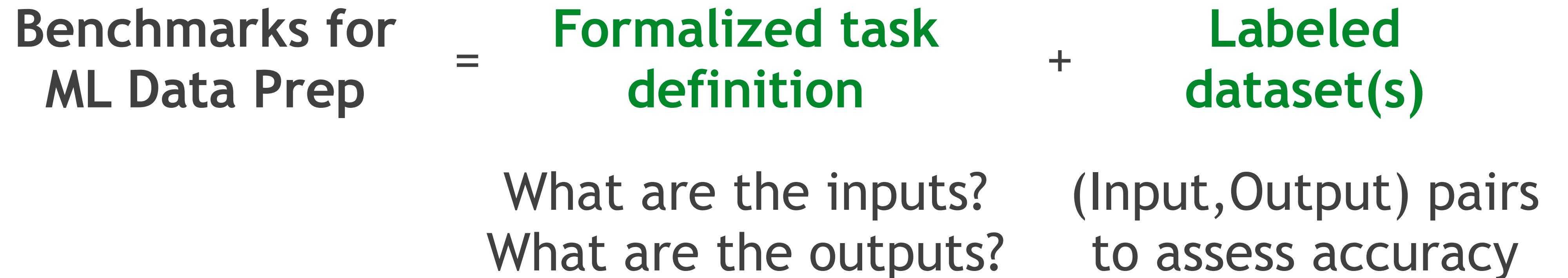
*The ML Data Prep Zoo: Towards Semi-Automatic Data Preparation for ML. SIGMOD DEEM'19*  
*Towards Benchmarking Feature Type Inference for AutoML Platforms. SIGMOD'21*

# Project SortingHat

Benchmarks for  
ML Data Prep = Formalized task  
definition

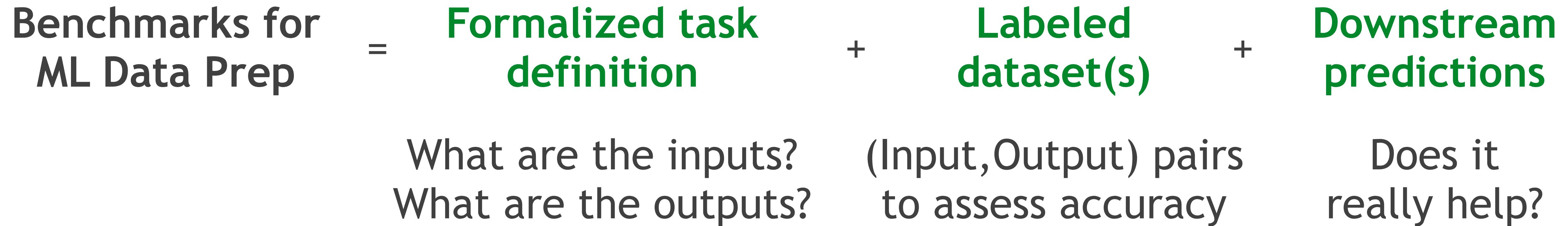
What are the inputs?  
What are the outputs?

# Project SortingHat



*The ML Data Prep Zoo: Towards Semi-Automatic Data Preparation for ML. SIGMOD DEEM'19  
Towards Benchmarking Feature Type Inference for AutoML Platforms. SIGMOD'21*

# Project SortingHat



*The ML Data Prep Zoo: Towards Semi-Automatic Data Preparation for ML. SIGMOD DEEM'19  
Towards Benchmarking Feature Type Inference for AutoML Platforms. SIGMOD'21*

# Project SortingHat

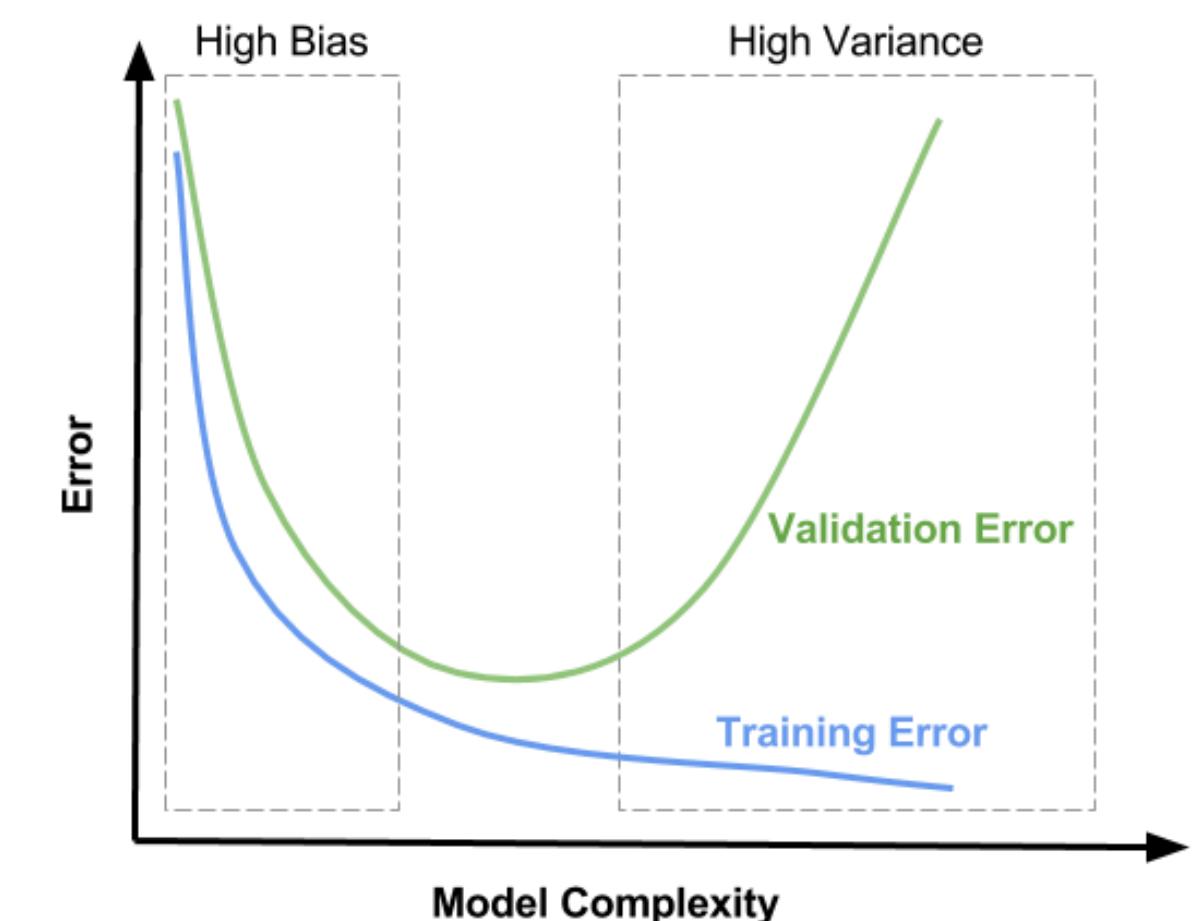
Benchmarks for ML Data Prep = **Formalized task definition** + **Labeled dataset(s)** + **Downstream predictions**

What are the inputs?  
What are the outputs?

(Input,Output) pairs  
to assess accuracy

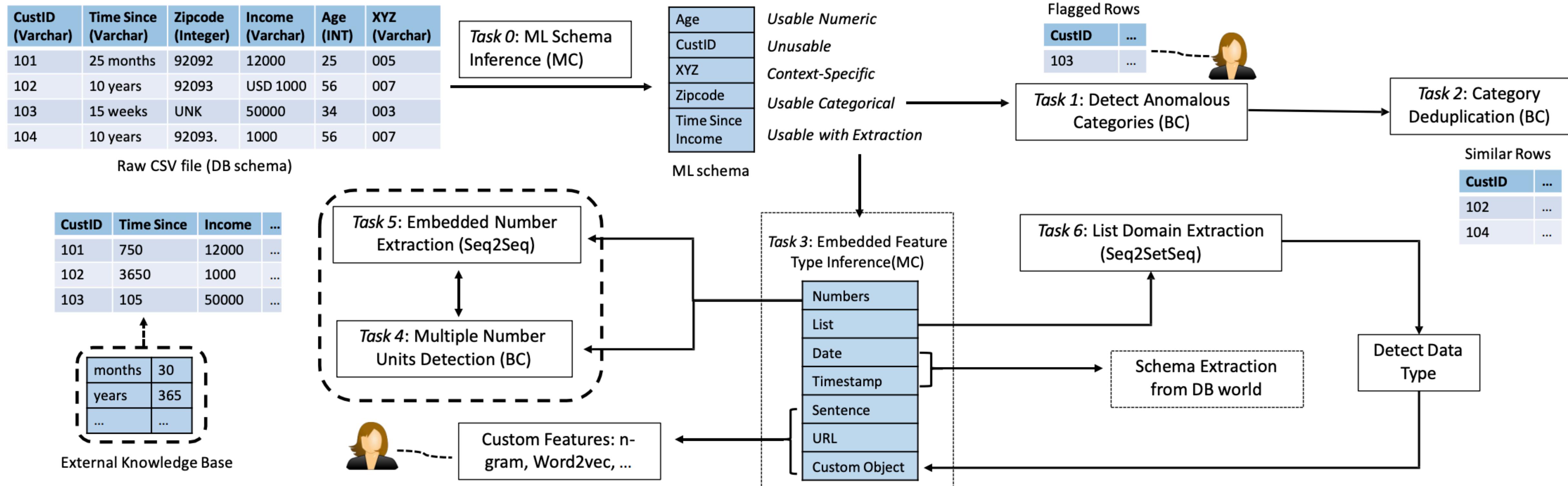
Does it  
really help?

How exactly does data prep affect ML accuracy?  
Are “failsafes” possible for errors in auto data prep?



# Project SortingHat

Current major tasks covered in our ML Data Prep Zoo benchmarks:



The ML Data Prep Zoo: Towards Semi-Automatic Data Preparation for ML. **SIGMOD DEEM'19**  
Towards Benchmarking Feature Type Inference for AutoML Platforms. **SIGMOD'21**

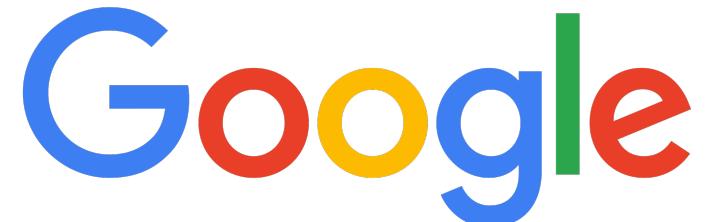
# SortingHat: Early Impact and Trajectory

Integrated with TFDV in TFX in collaboration with Google

Being tested on Google's internal AutoML benchmarks

Our model used in NSF-funded “AI Maker” model + dataset search tool

Coming soon: AWS DBMS and ML/AI products; OpenML; catalog search



# SortingHat: Open Source Benchmark Repo

Screenshot of the GitHub repository page for `pvn25/ML-Data-Prep-Zoo`.

The repository is public and has the following details:

- Code**: master branch, 1 branch, 1 tag.
- Issues**: 8
- Pull requests**: 0
- Actions**: 0
- Projects**: 0
- Wiki**: 0
- Security**: 0
- Insights**: 0

**Commits** (35 total):

File	Type	Date
MLFeatureTypeInference	updates	4 months ago
.gitattributes	updates	11 months ago
LICENSE	updates	11 months ago
README.md	updates	6 months ago

**README.md** content:

## ML Data Prep Zoo

A zoo of labelled datasets and ML models for data prep tasks. Please refer to our [paper](#) for more details.

**Tasks:**

- Task 1 (t1): ML Feature Type Inference (Multi-class classification)
- Task 2 (t2): Category Deduplication (Binary classification)
- Task 3 (t3): Embedded Number Extraction (Sequence-to-sequence learning)
- Task 4 (t4): Detect Anomalous Categories (Binary classification)
- Task 5 (t5): Multiple Number Units Detection (Binary classification)
- Task 6 (t6): List Domain Extraction (Sequence-to-set-of-sequence learning)

**About**: No description, website, or topics provided.

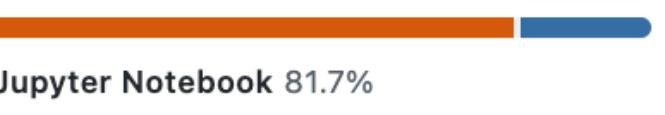
**Code**: Go to file, Code dropdown.

**Readme**, **Apache-2.0 License**.

**Releases**: 1 tags.

**Packages**: No packages published.

**Languages**: Jupyter Notebook 81.7%, Python 18.3%.



Links on project webpage: <https://adalabucsd.github.io/sortinghat.html>

# SortingHat: Open Source Benchmark Repo

The screenshot shows a GitHub repository page for 'ML-Data-Pool'. The repository path is 'pvn25 / ML-Data-Pool / ML-Data-Prep-Zoo / MLFeatureTypeInference / Benchmark-Labeled-Data /'. The 'Code' tab is selected, showing a commit by 'Vraj Shah' from May 13, 2018, with 8 issues. The commit message is 'Vraj Shah and Vraj Shah updates'. The commit details show updates to 'Metadata', 'README.md', 'data\_test.csv', and 'data\_train.csv'. The 'README.md' file contains the following content:

```
## Benchmark Labeled Data

- `data_train` and `data_test` contains the base featurization split of the raw CSV files
- `y_act` column in the files denote the ground truth label. The coding of the labels is given as follows:

  Numeric : 0
  Categorical: 1
  Datetime:2
  Sentence:3
  URL: 4
  Numbers: 5
  List: 6
  Not-Generalizable: 7
  Custom Object (or Context-Specific): 8

- `Metadata/` contains the record id and the source details of the raw CSV files.
- `Our-Base-Featurization-Split/` contains the additional features extracted from the base featurized files for the ML models
- The raw data files that we used to create the base featurized files is available here for download.
```

Links on project webpage: <https://adalabucsd.github.io/sortinghat.html>

# SortingHat: Open Source Benchmark Repo

Code Issues 8

Vraj Shah and Vraj Shah updates 30b7557 on May 13 History

Metadata updates 11 months ago

..

Dictionary updates 11 months ago

CNN updates 6 months ago

LogReg.pkl updates 6 months ago

README.md updates 11 months ago

RandomForest.pkl updates 6 months ago

SVM.pkl updates 6 months ago

Not-Generalizable: 7  
Custom Object (or Context-Specific): 8

- Metadata/ contains the record id and the source details of the raw CSV files.
- Our-Base-Featurization-Split/ contains the additional features extracted from the base featurized files for the ML models
- The raw data files that we used to create the base featurized files is available here for [download](#).

Links on project webpage: <https://adalabucsd.github.io/sortinghat.html>

# SortingHat: Open Source Benchmark Repo

The screenshot shows a GitHub repository interface for the 'Downstream-Benchmark' project. The repository has 8 issues and is last updated 11 months ago. The main content area displays an 'AutoML Benchmark' table comparing various machine learning models across different feature types.

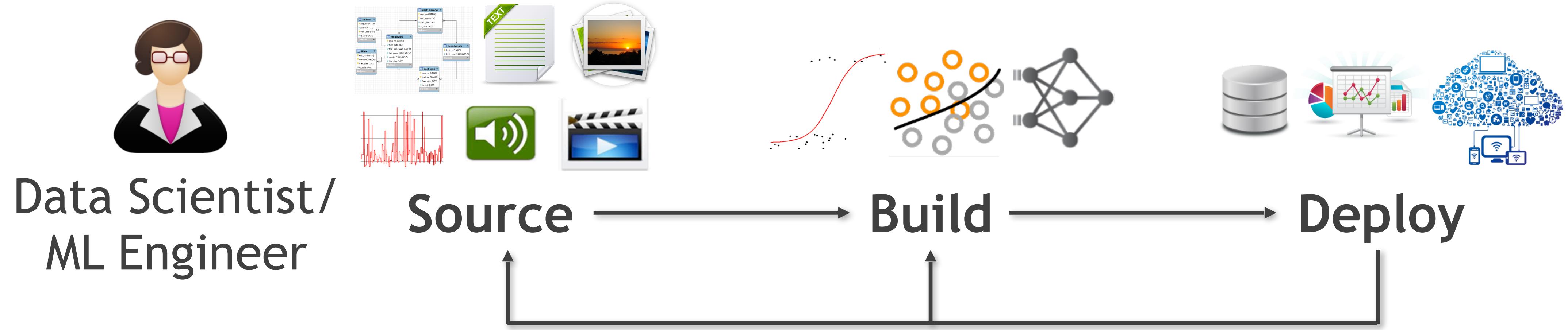
**AutoML Benchmark**

The following table presents the binarized class-specific accuracy, precision, and recall of different approaches on our **benchmark labeled held-out test dataset**.

Feature Type	Metric	TFDV	Pandas	TransmogrifAI	AutoGluon	Log Reg	CNN	Rand Forest
Numeric	Precision	0.64	0.615	0.605	0.648	0.91	0.892	0.936
	Recall	1	1	1	1	0.931	0.977	0.987
	Accuracy	0.799	0.777	0.767	0.807	0.943	0.95	0.971
Categorical	Precision	0.414	-	-	0.703	0.826	0.875	0.91
	Recall	0.643			0.534	0.891	0.888	0.954
	Accuracy	0.708			0.841	0.931	0.945	0.968
Datetime	Precision	0.972	0.956	1	1	0.985	0.957	0.986
	Recall	0.489	0.915	0.454	0.887	0.957	0.957	0.972
	Accuracy	0.963	0.991	0.961	0.992	0.996	0.994	0.997
Sentence	Precision	0.475	-	-	0.512	0.882	0.871	0.899
	Recall	0.511			0.913	0.728	0.804	0.87
	Accuracy	0.951			0.956	0.983	0.985	0.989
Not-Generalizable	Precision	-	-	-	0.45	0.709	0.714	0.946

Links on project webpage: <https://adalabucsd.github.io/sortinghat.html>

# The New DBfication of ML/AI



Metadata Management for ML  
Data Prep/Cleaning for ML  
Multimodal ML Query Models  
Data Search, Labeling, etc.

...

Scalable Data Systems for ML  
Query Optimization for ML  
Cloud and Streaming Infra.  
Provenance and Debugging

...

Benchmark Frameworks and Data  
Fairness, Transparency, Privacy, etc.



*Leaving these problems open => Huge waste of time/  
effort/money/energy/etc. by users of ML/AI!*

*Leaving these problems open => Huge waste of time/  
effort/money/energy/etc. by users of ML/AI!*

*It is high time for DB, ML/AI, and more areas to join  
forces to accelerate the DBfication of ML/AI!*

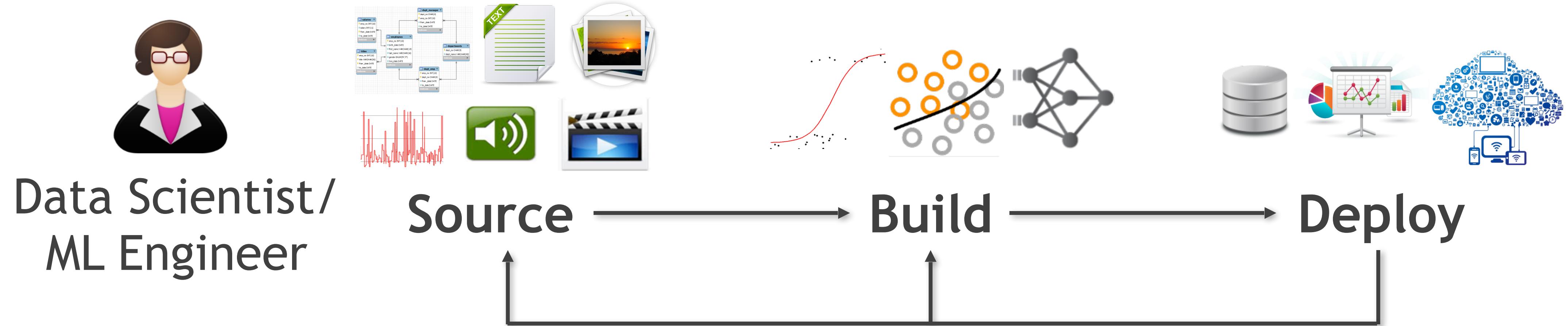
# Outline

■ The New DBfication of ML/AI

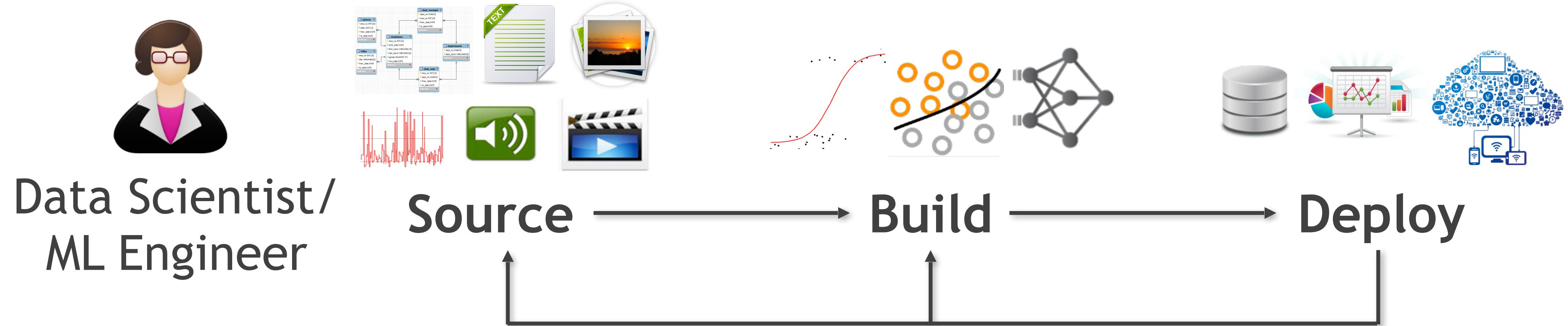
■ Two Examples from My Research

■ Accelerating the DBfication of ML/AI

# DBfication of ML/AI is NOT just a DB Concern

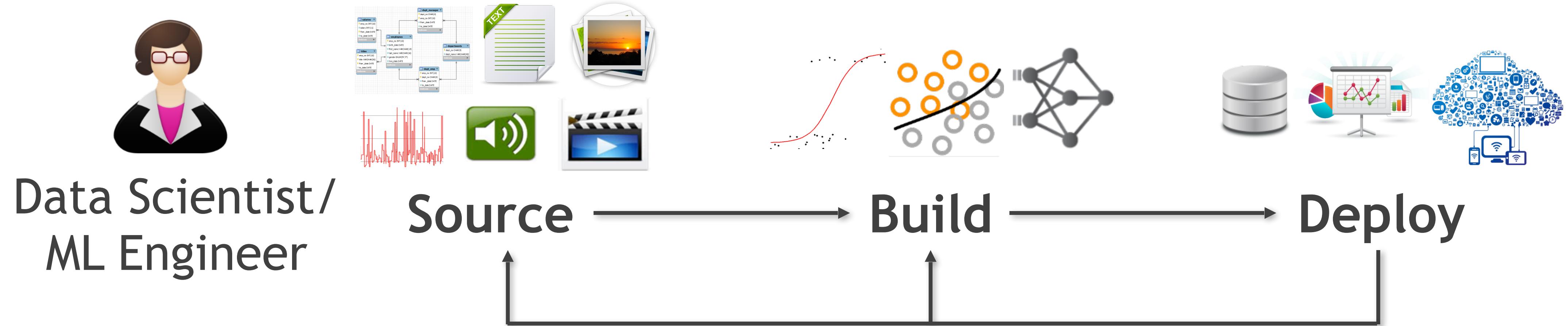


# DBfication of ML/AI is NOT just a DB Concern



1a. Applications of ML, IR, and data mining in this arena:

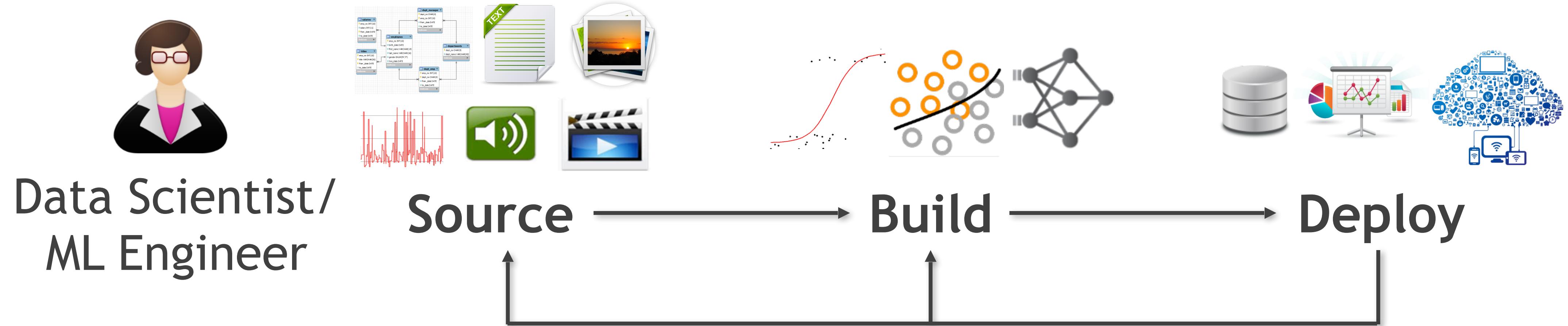
# DBfication of ML/AI is NOT just a DB Concern



1a. Applications of ML, IR, and data mining in this arena:

Programmatic and learned label engineering

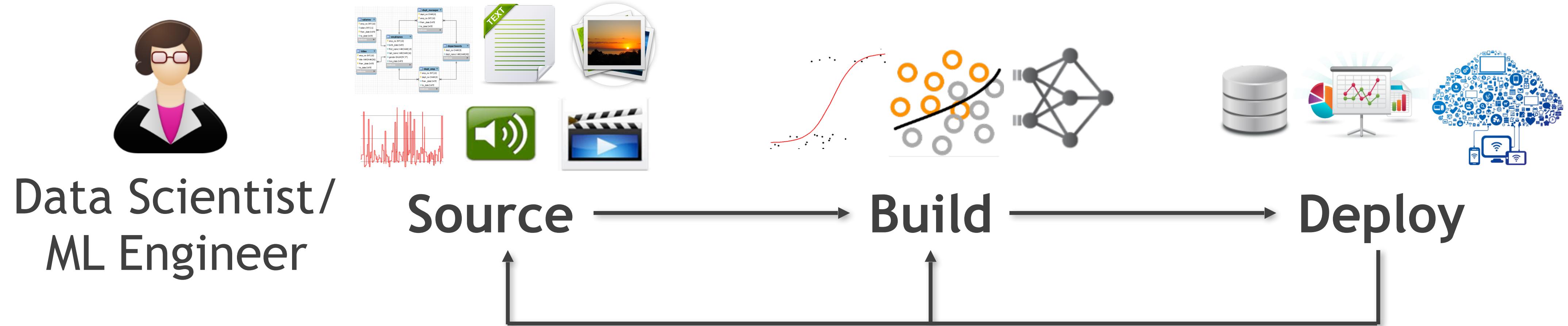
# DBfication of ML/AI is NOT just a DB Concern



## 1a. Applications of ML, IR, and data mining in this arena:

Programmatic and learned label engineering  
Dataset discovery and search systems

# DBfication of ML/AI is NOT just a DB Concern



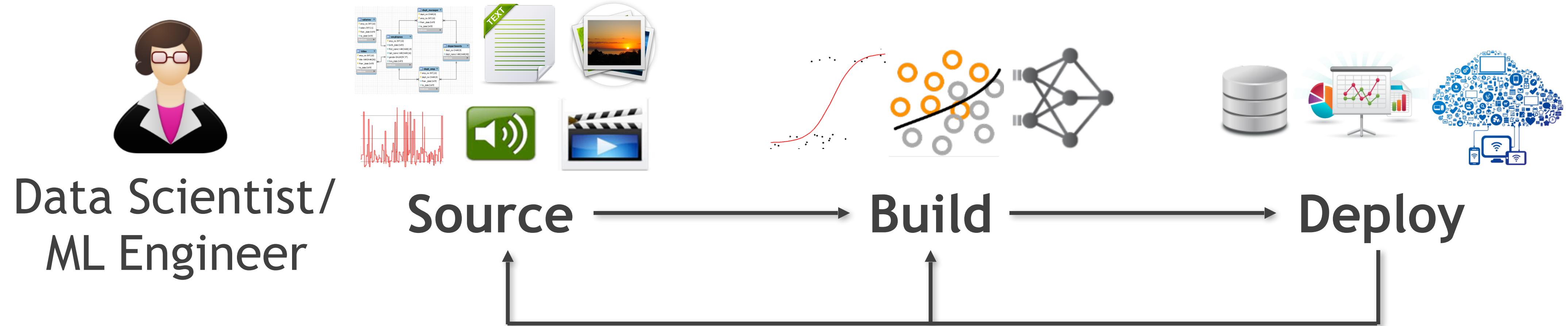
## 1a. Applications of ML, IR, and data mining in this arena:

Programmatic and learned label engineering

Dataset discovery and search systems

Recommendation systems for Data Science pipelines

# DBfication of ML/AI is NOT just a DB Concern



## 1a. Applications of ML, IR, and data mining in this arena:

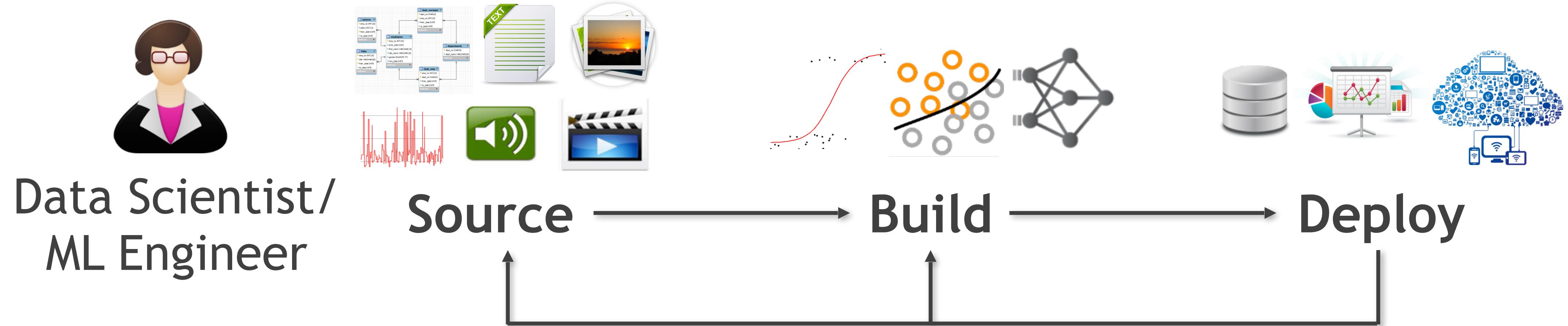
Programmatic and learned label engineering

Dataset discovery and search systems

Recommendation systems for Data Science pipelines

Automatic prediction monitoring and actionable recourse

# DBfication of ML/AI is NOT just a DB Concern



## 1a. Applications of ML, IR, and data mining in this arena:

Programmatic and learned label engineering

Dataset discovery and search systems

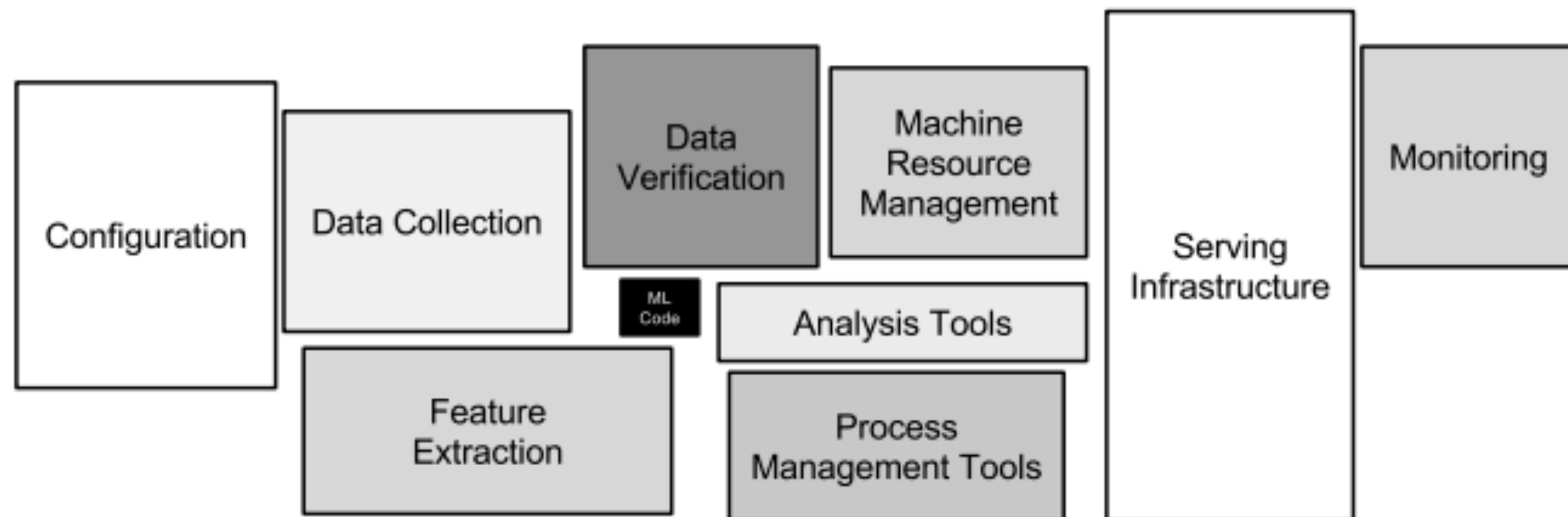
Recommendation systems for Data Science pipelines

Automatic prediction monitoring and actionable recourse

Hybrid human-in-the-loop AutoML platforms

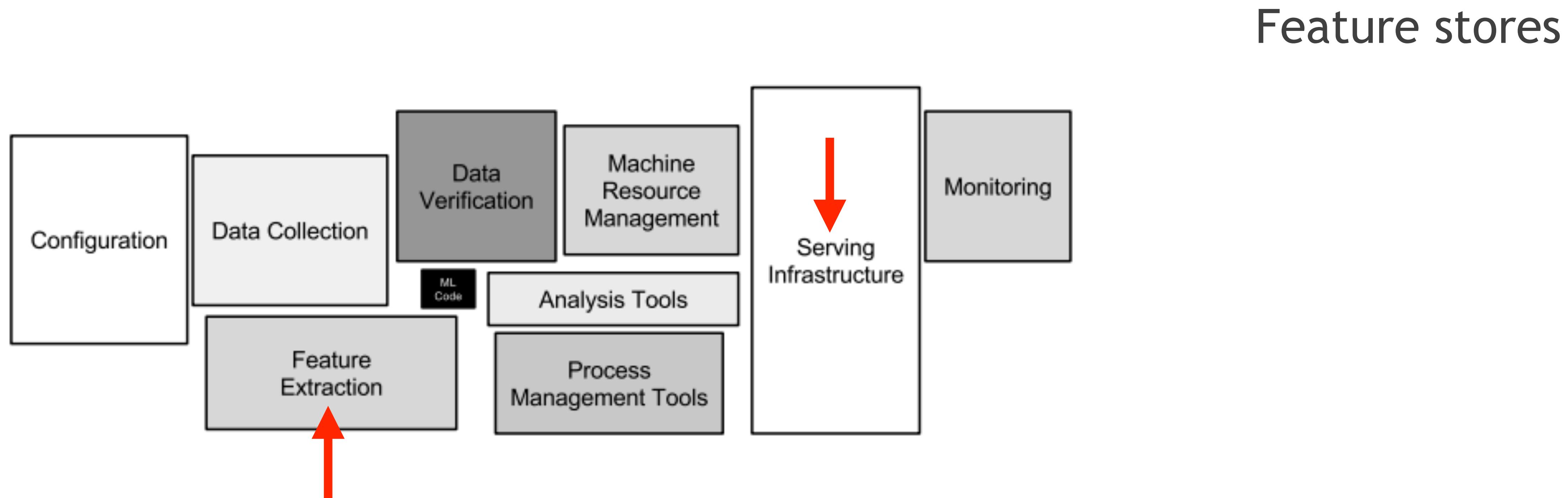
# DBfication of ML/AI is NOT just a DB Concern

## 1b. More intersection with systems, PL/software engineering, etc:



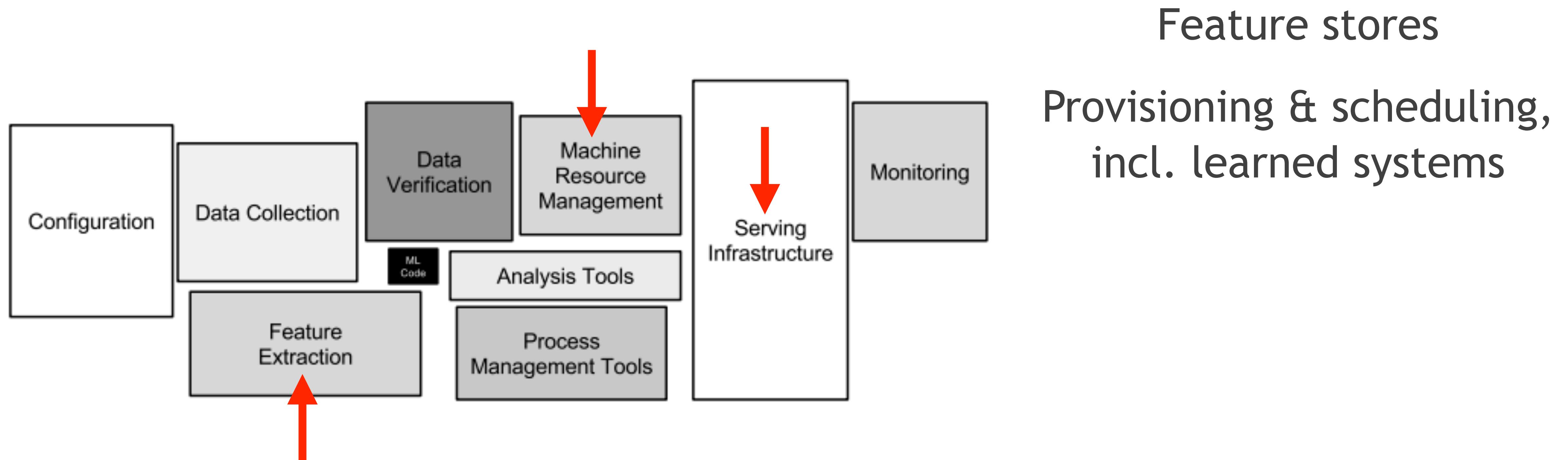
# DBfication of ML/AI is NOT just a DB Concern

## 1b. More intersection with systems, PL/software engineering, etc:



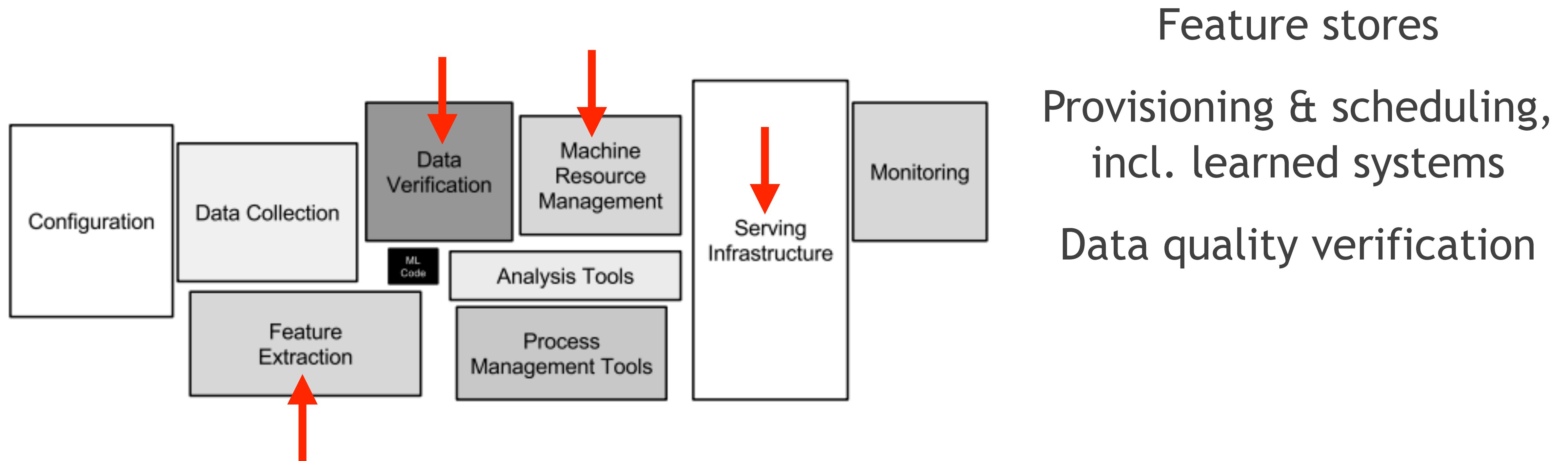
# DBfication of ML/AI is NOT just a DB Concern

## 1b. More intersection with systems, PL/software engineering, etc:



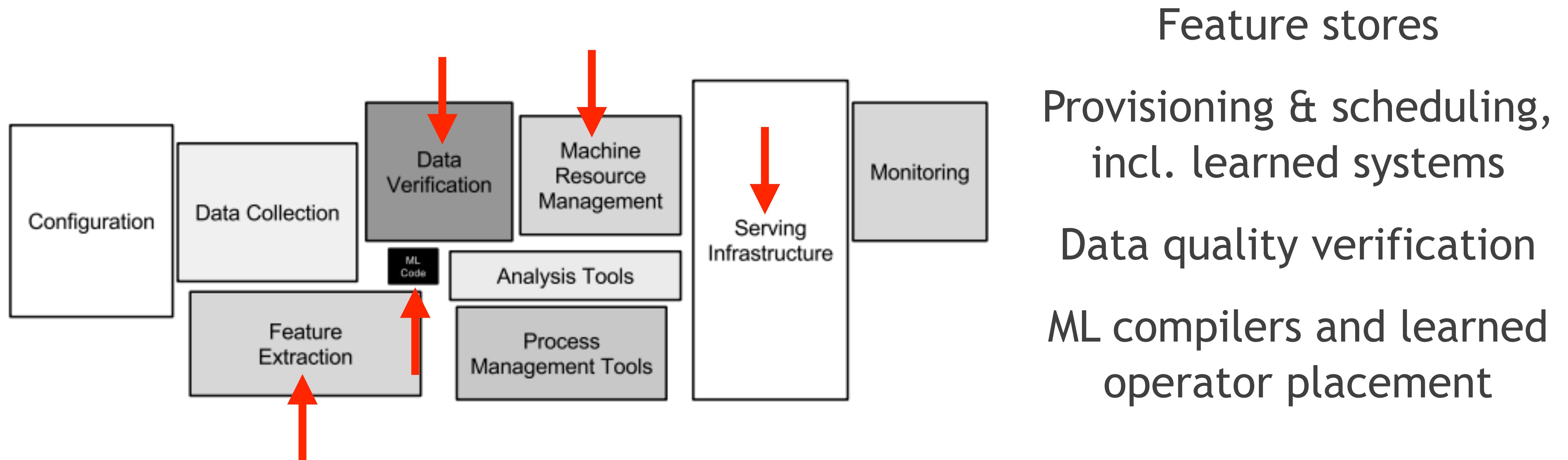
# DBfication of ML/AI is NOT just a DB Concern

## 1b. More intersection with systems, PL/software engineering, etc:



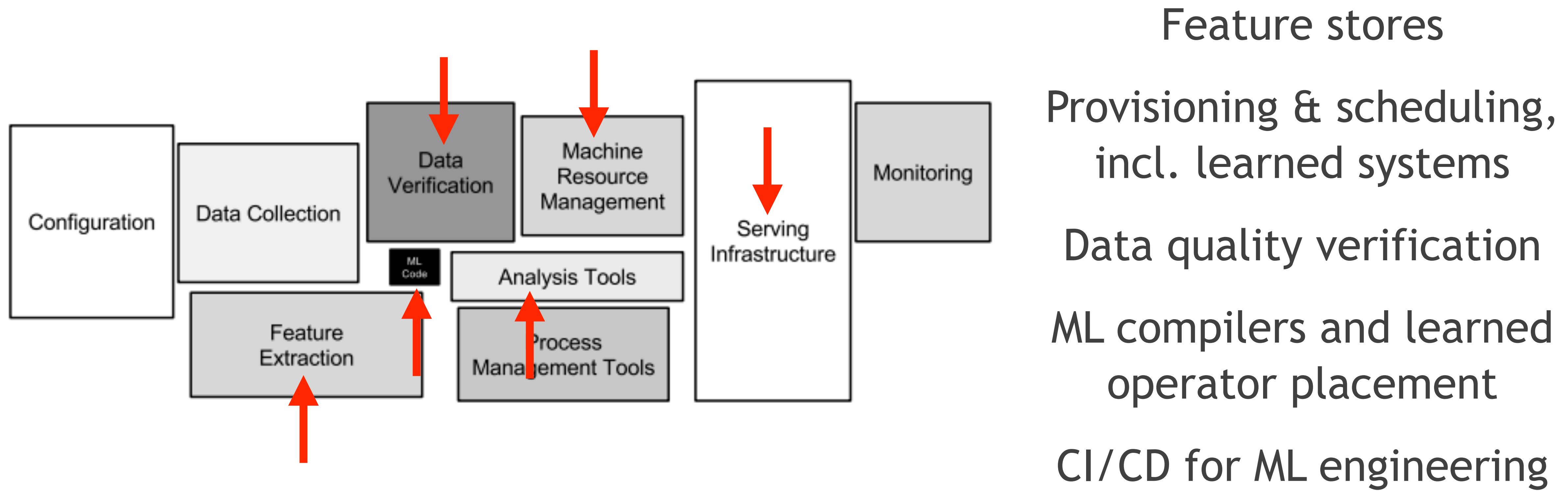
# DBfication of ML/AI is NOT just a DB Concern

## 1b. More intersection with systems, PL/software engineering, etc:



# DBfication of ML/AI is NOT just a DB Concern

## 1b. More intersection with systems, PL/software engineering, etc:



...

# Accelerating the DBfication of ML/AI

2. Check out my “DB for ML” grad course and research book? :)

*CSE 234/291: Data Systems for Machine Learning.* UC San Diego. 2020.

*Data Management in Machine Learning Systems.* Morgan & Claypool Publishers. 2019.

# Accelerating the DBfication of ML/AI

2. Check out my “DB for ML” grad course and research book? :)

*CSE 234/291: Data Systems for Machine Learning.* UC San Diego. 2020.

*Data Management in Machine Learning Systems.* Morgan & Claypool Publishers. 2019.

3. Check out SIGMOD DEEM and HILDA Workshops. Check out MLSys.

# Accelerating the DBfication of ML/AI

## 2. Check out my “DB for ML” grad course and research book? :)

*CSE 234/291: Data Systems for Machine Learning.* UC San Diego. 2020.

*Data Management in Machine Learning Systems.* Morgan & Claypool Publishers. 2019.

## 3. Check out SIGMOD DEEM and HILDA Workshops. Check out MLSys.

## 4. Check out recent “DB for ML” tutorials and papers.

*Data Management in Machine Learning: Challenges, Techniques, and Systems.* SIGMOD 2017.

*Data Management Challenges in Production Machine Learning.* SIGMOD 2017.

*Data Collection and Quality Challenges for Deep Learning.* VLDB 2020.

# Accelerating the DBfication of ML/AI

# Accelerating the DBfication of ML/AI

5. Check out topical panel discussions on “DB for ML” stuff.

# Accelerating the DBfication of ML/AI

## 5. Check out topical panel discussions on “DB for ML” stuff.



Photos from the workshop. L to R: (1) The DEEM audience. (2) The Panelists: Matei, Joaquin, Jens, Joey, and Manasi (Martin not pictured). (3) Advocatus Diaboli.

SIGMOD DEEM'18

# Accelerating the DBfication of ML/AI

## 5. Check out topical panel discussions on “DB for ML” stuff.

The screenshot shows a blog post from the ACM SIGMOD Blog. The title is "ML/AI SYSTEMS AND APPLICATIONS: IS THE SIGMOD/Vldb COMMUNITY LOSING RELEVANCE?". The author is Arun Kumar, and the date is AUGUST 21, 2018. The post includes a small profile picture of Arun Kumar and a link to the full article. Below the post is a photograph of a workshop audience in a conference room, with a panel of speakers seated at a table in front of a whiteboard.

Photos from the workshop. L to R: (1) The DEEM audience. (2) The Panelists: Matei, Joaquin, Jens, Joey, and Manasi (Martin not pictured). (3) Advocatus Diaboli.

SIGMOD DEEM'18



### Hot Button Panel Discussion

*Automation of Data Prep, ML, and Data Science:  
New Cure or Snake Oil?*



### Panel of Experts:



Felix Naumann  
University of Potsdam &  
Hasso Plattner Institute



Ihab Ilyas  
Inductiv, Tamr, &  
University of Waterloo



Joe Hellerstein  
Trifacta &  
UC Berkeley



Luna Dong  
Amazon



Sarah Catanzaro  
Amplify Partners

### Panel Chair(s):

Arun Kumar  
UC San Diego



&  
Dr. A. Diaboli  
At large

SIGMOD'21

# Accelerating the DBfication of ML/AI

## 6. MOST IMPORTANT: Build REAL stuff and help transfer research to practice via partnerships/collaboration with ML/AI users.

Data scientists, Data analysts, ML engineers, MLOps engineers, etc.

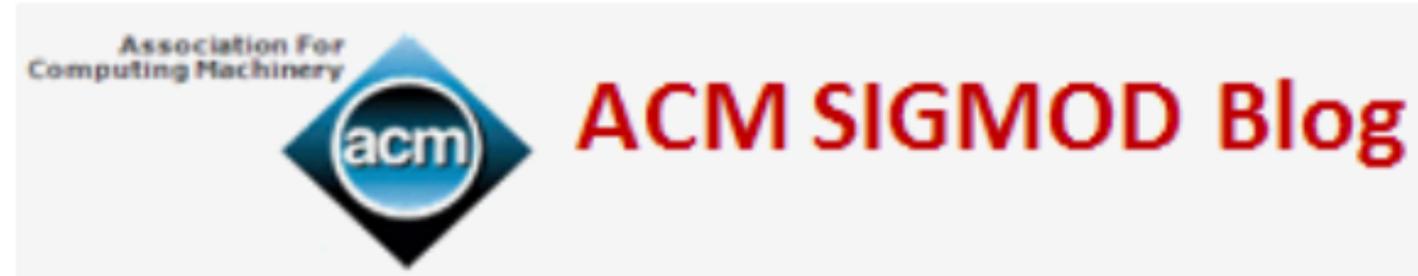
Domain sciences, enterprises, Web companies, cloud vendors, policy, etc.

Create open source artifacts, both software and datasets

Attend both research and industry venues: Spark+AI Summit, FOSDEM, etc.

...

# New Publication Avenues



Alon Halevy, Arun  
Kumar and  
Nesime Tatbul

FEBRUARY 10,  
2020

## SCALABLE DATA SCIENCE: A NEW RESEARCH TRACK CATEGORY AT PVLDB VOL 14 / VLDB

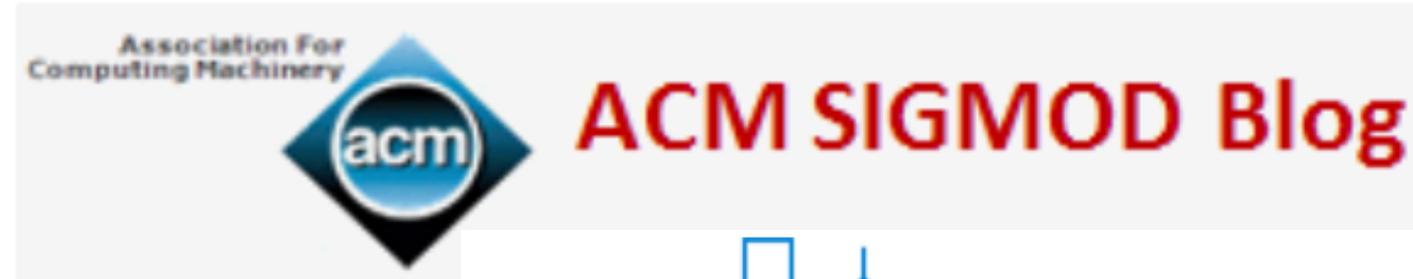
2021

≡ Uncategorized

This post introduces and explains the newly created category of “Scalable Data Science” within the Research Track of PVLDB. This category comes into effect for volume 14, i.e., submissions starting April 1, 2020, which will be evaluated by the Review Board of PVLDB vol 14 for presentation at VLDB 2021.

The Growth of Data Science

# New Publication Avenues



Alon Halev  
Kumar  
Nesime  
FEBRUAR  
2020

## SIGMOD 2022 CALL FOR RESEARCH PAPERS

- **Data Science Track**

We invite the submission of original research in data science targeting the entire data life cycle of real applications. This data life cycle encompasses databases/data management/data systems/data engineering often leveraging statistical, Machine Learning and Artificial Intelligence methods and using massive and heterogeneous collections of potentially messy datasets. Data science papers study phenomena at scales and granularities never before possible. Such papers are expected to focus on data-intensive components of data science pipelines; and solve problems in areas of interest to the

ARCH  
LDB

ata  
t for  
y the

# New Publication Avenues

The screenshot shows the ACM SIGMOD Blog website for the Fifth Conference on Machine Learning and Systems. The header features the ACM logo and the text "ACM SIGMOD Blog". Below the header, a sidebar on the left lists navigation links: "Year (2022) ▾", "Help ▾", "My Registrations", "Profile ▾", "Contact Us", "Sponsor Info", "Conflicts of Interest", "Code of Conduct", and "Proceedings". A large "MLSys" button is at the bottom of the sidebar. The main content area displays the conference details: "Santa Clara Convention Center" and "Mon Apr 11th through Thu the 14th, 2022". It includes sections for "Registration" (with "Pricing" and "Register starting Feb 06 01 PM PST"), "Schedule", and "Video Library 2021". A "Tweet" button is also present. The "Conference Overview" section describes the conference's focus on the intersection of machine learning and systems, mentioning topics like efficient model training, distributed learning algorithms, privacy, security, testing, debugging, monitoring, fairness, interpretability, explainability, data preparation, feature selection, and feature extraction.

phenomena at scales and granularities never before possible. Such papers are expected to focus on data-intensive components of data science pipelines; and solve problems in areas of interest to the

# New Publication Avenues

The screenshot shows the ACM SIGMOD Blog website. At the top left is the ACM logo and the text "Association For Computing Machinery". The main title "ACM SIGMOD Blog" is in red. Below it, the text "Fifth Conference on Machine Learning and Systems" is displayed. On the left sidebar, there are links for "Year (2022) ▾", "Help ▾", and "My Registrations". The main content area shows the location as "Santa Clara Convention Center" and the dates as "Mon Apr 11th through Thu the 14th, 2022". Below this, there are two buttons: "Registration" and "Conference Overview". At the bottom of the page, there is a navigation bar with links for "HOME", "PROGRAM", "ATTENDING", "CALLS", "KDD CUP", "SPONSORS", and "ORGANIZERS". To the left of the main content, there is a logo for "KDD2021" featuring a blue geometric pattern.

## Call for Applied Data Science Track Papers

MLSys

phenomena at scales and granularities never before possible. Such papers are expected to focus on data-intensive components of data science pipelines; and solve problems in areas of interest to the

- applications
- Data preparation, feature selection, and feature extraction



*The DBfication of ML/AI is a grand challenge for all  
of computing to help democratize ML/AI.*

*The DBfication of ML/AI is a grand challenge for all  
of computing to help democratize ML/AI.*

*Intersection of DB, ML/AI, systems, HCI, etc.; work  
with ML/AI users in industry and domain sciences.*

# My Terrific Advisees



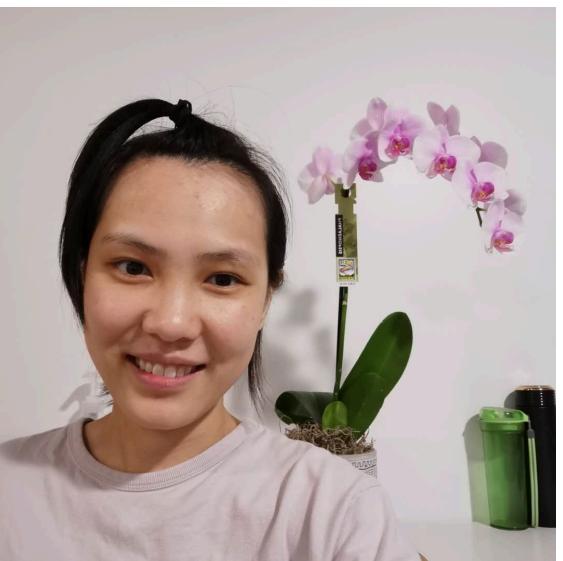
Supun Nakandala  
PhD



Tara Mirmira  
PhD



Vraj Shah  
PhD & MS



Xiuwen Zheng  
PhD & MS



Yuhao Zhang  
PhD & MS



Kabir Nagrecha  
BS & PhD

Liangde Li  
Kyle Luoma

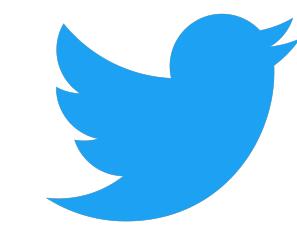
**Alumni:**  
Side Li  
Advitya Gemawat  
Shaoqing Yi  
David Justo  
Kevin Yang

<https://ADALabUCSD.github.io>

arunkk@eng.ucsd.edu



[github.com/ADALabUCSD](https://github.com/ADALabUCSD)



@TweetAtAKK

ACKS:

