ARIMO
Data Intelligence for All

DISTRIBUTED TENSORFLOW:

SCALING GOOGLE'S DEEP LEARNING LIBRARY ON SPARK

Christopher Nguyen, PhD. | Chris Smith | Ushnish De | Vu Pham | Nanda Kishore

ADATAO
DATA INTELLIGENCE FOR ALL

Nov 2015

Strata NYC

Distributed
DL on Spark
+ Tachyon

ARIMO
Data Intelligence for All

Feb 2016

Spark Summit East

Distributed
TensorFlow
on Spark

ARIMO
Data Intelligence for All

June 2016

Hadoop Summit

TensorFlow
Ensembles
Spark

Outline

1. Motivation
2. Architecture
3. Experimental Parameters
4. Results
5. Lesson Learned
6. Summary

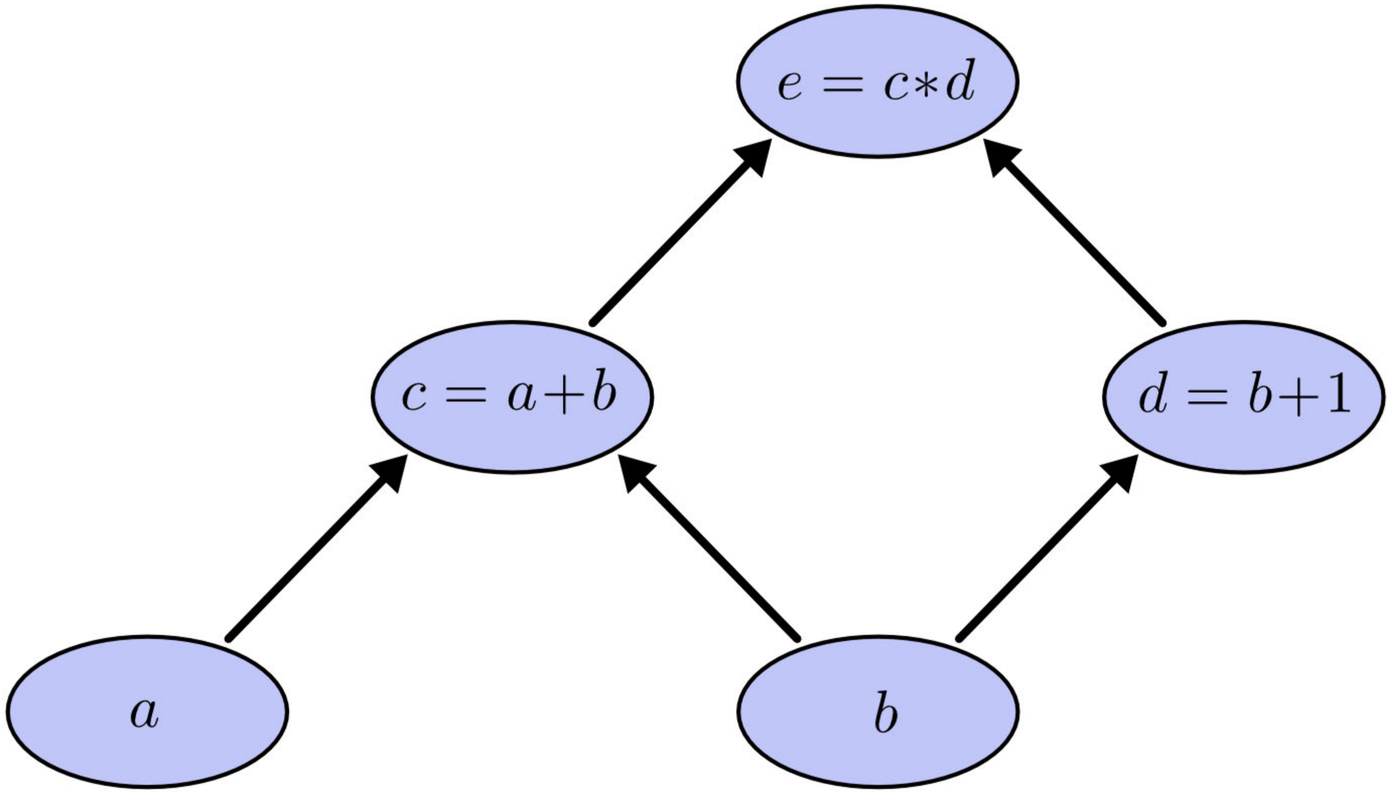
Motivation

1 Google TensorFlow

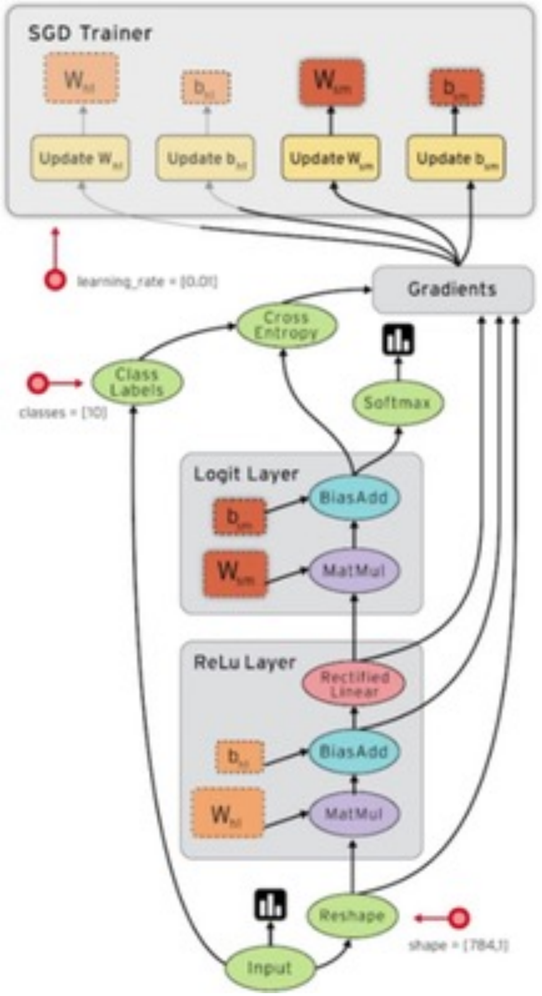
2 Spark Deployments

TensorFlow Review

Computational Graph



Graph of a Neural Network



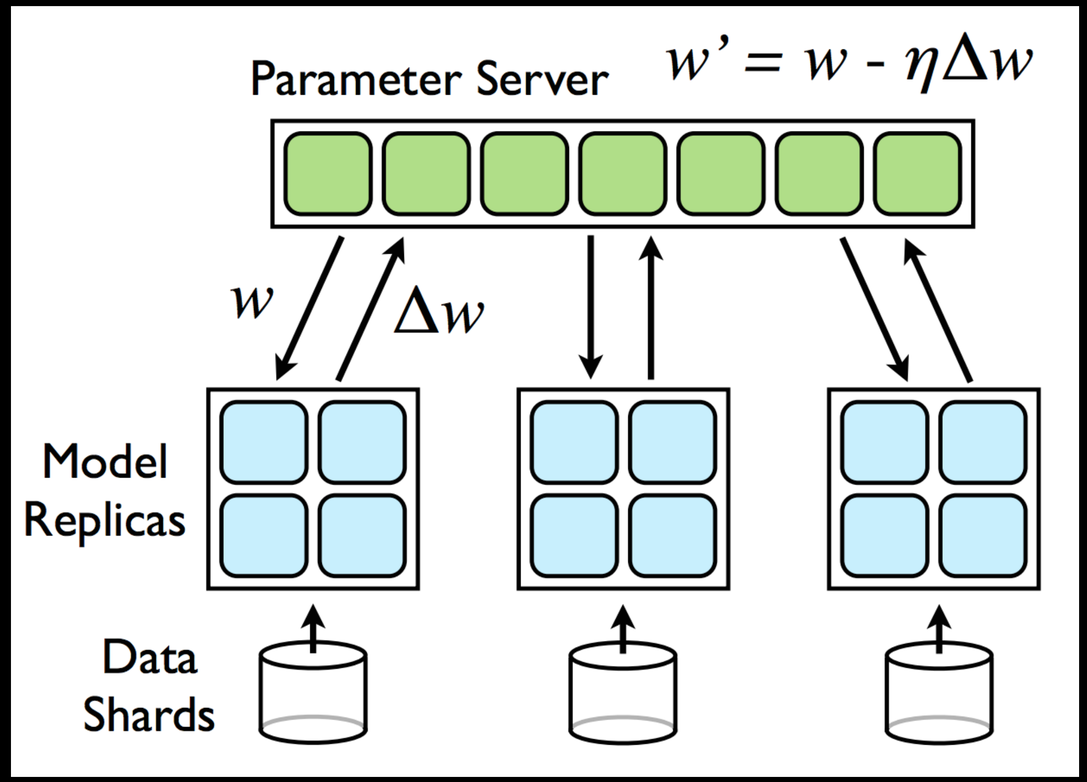
Why TensorFlow?

Google uses TensorFlow for the following:

- Search
- Advertising
- Speech Recognition
- Photos
- Maps
- StreetView—Blurring out house numbers
- Translate
- YouTube

DistBelief

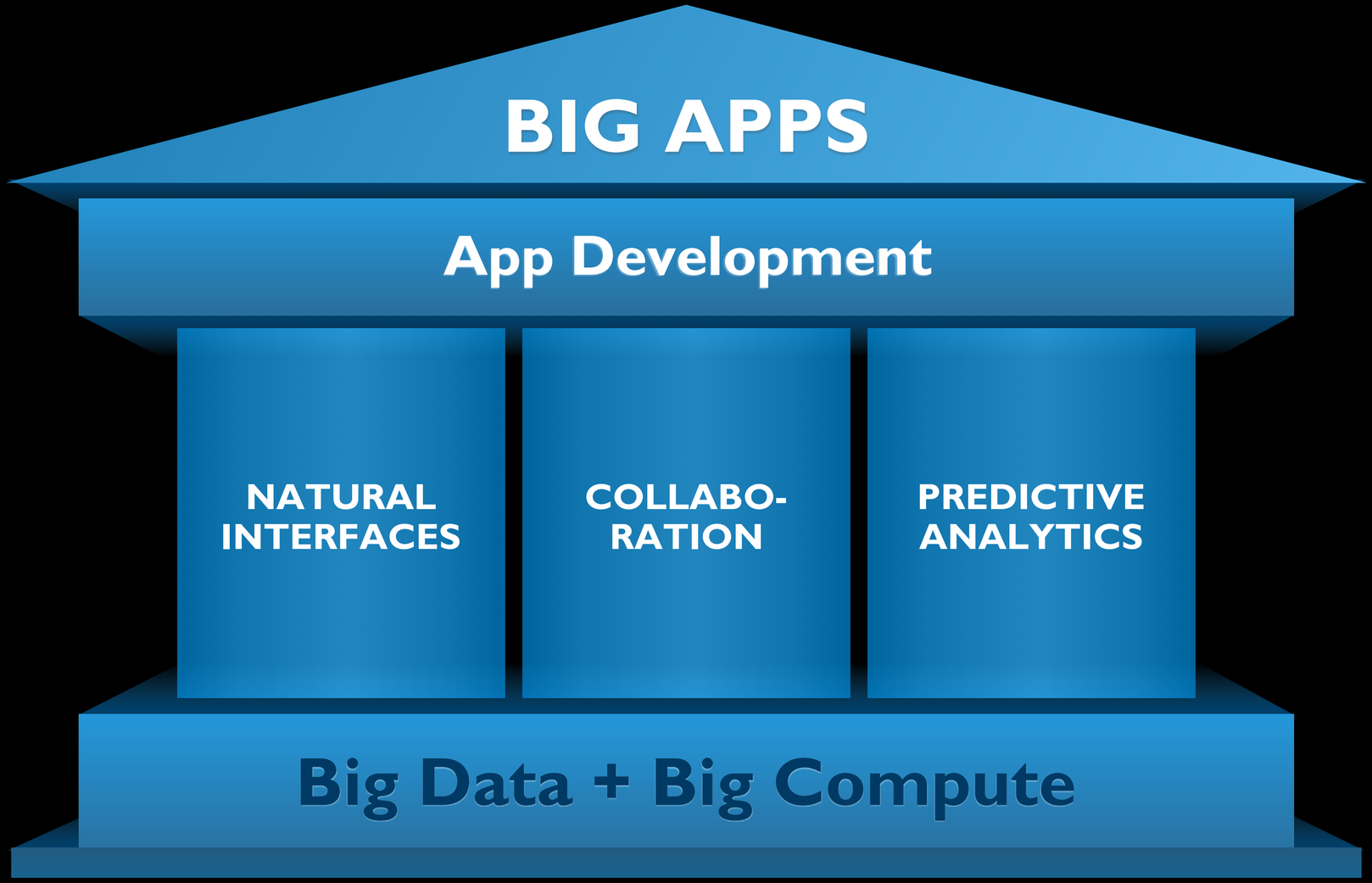
1. Compute Gradients



2. Update Params (Descent)

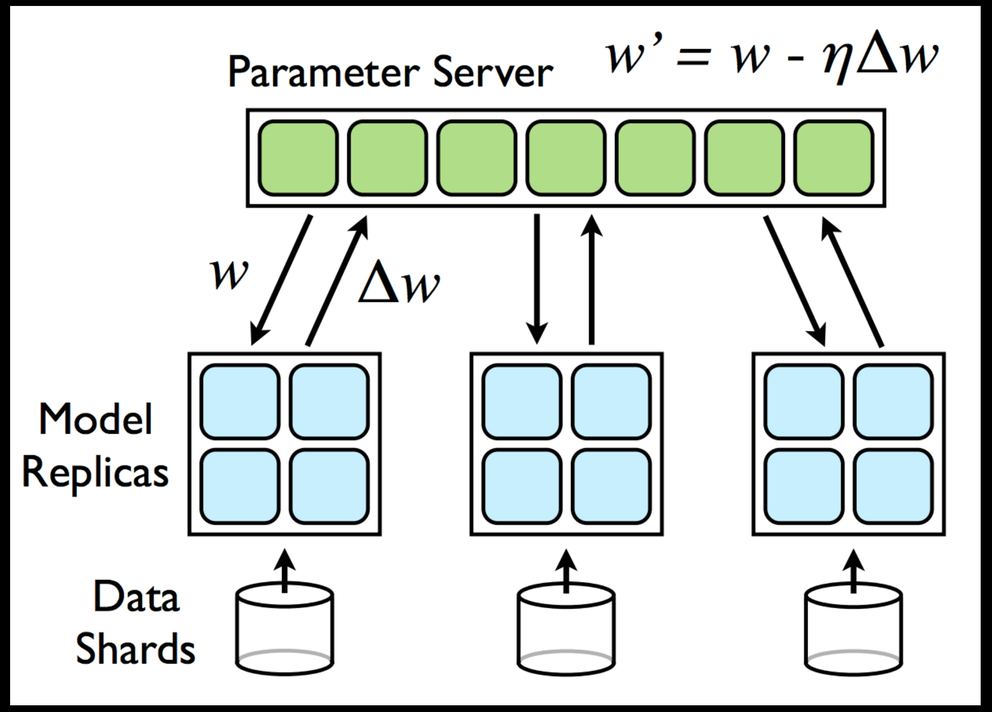
Large Scale Distributed Deep Networks, Jeff Dean et al, NIPS 2012

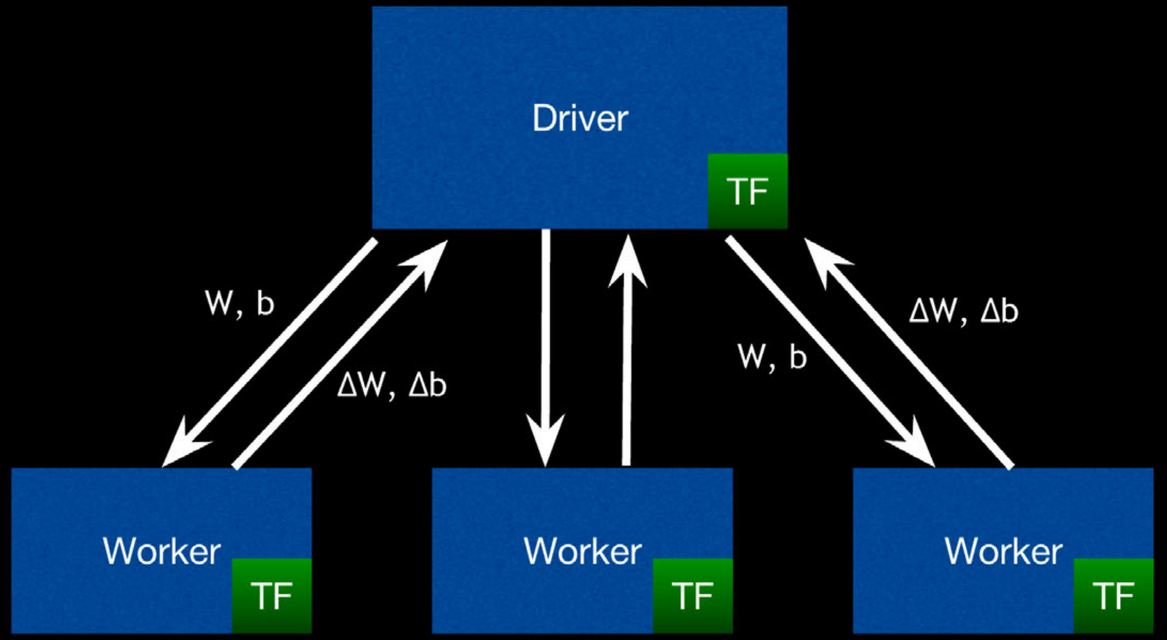
Arimo's 3 Pillars



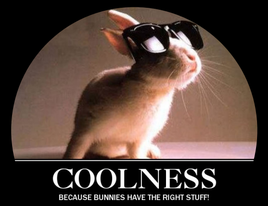
Architecture

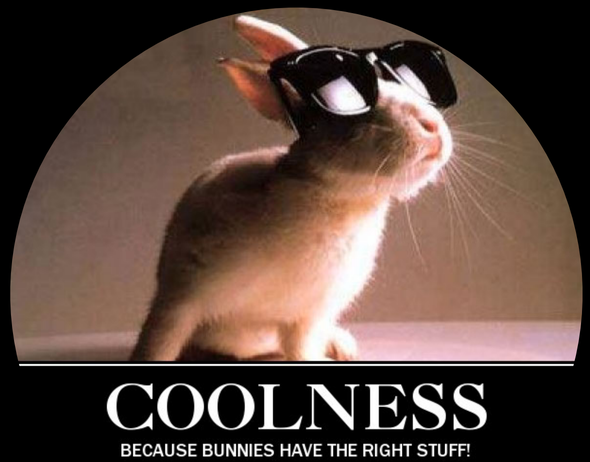
TensorSpark Architecture





Spark & Tachyon Architectural Options





Spark-Only

Model as Broadcast Variable

Tachyon-Storage

Model Stored as Tachyon File

Param-Server

Model Hosted in HTTP Server

Tachyon-CoProc

Model Stored *and* Updated by Tachyon

Experimental Parameters

Datasets

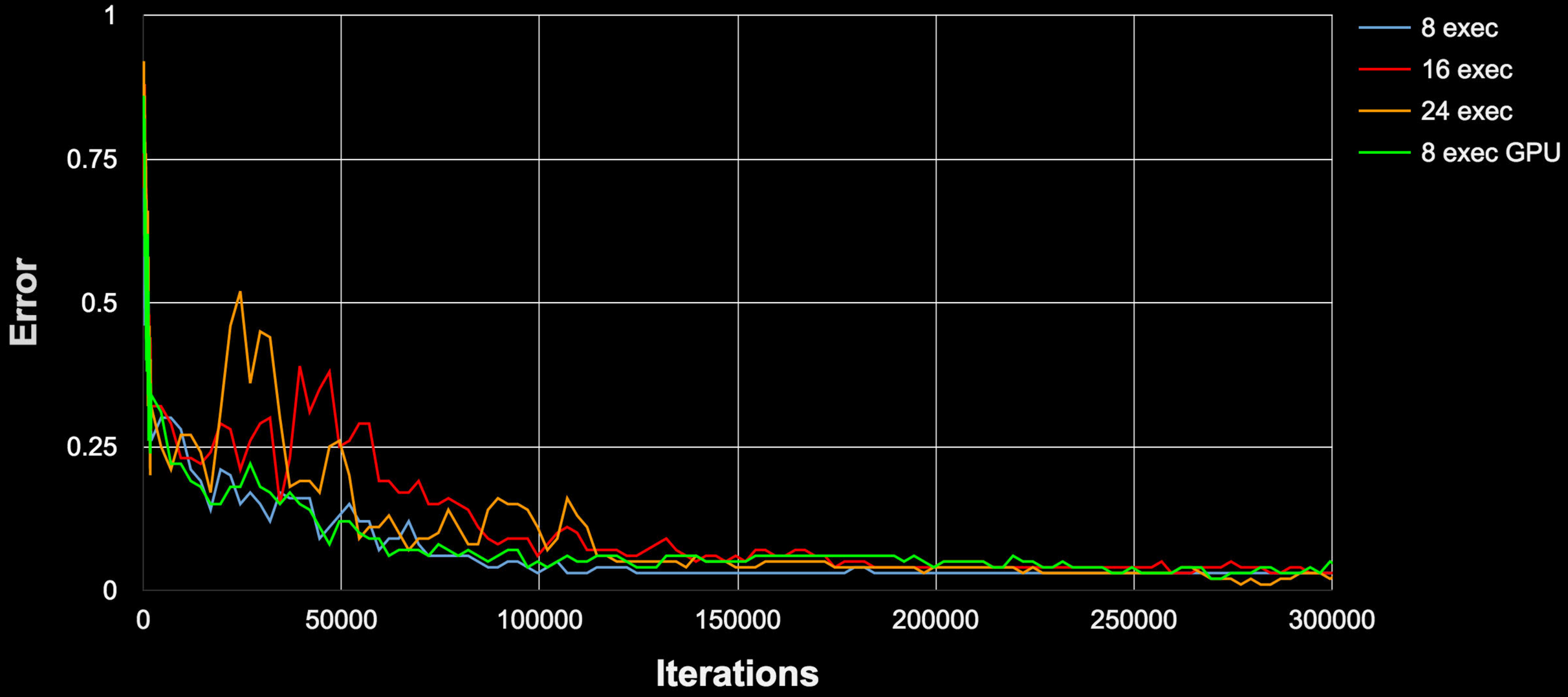
Data Set	**Size**	**Parameters**	Hidden Layers	Hidden Units
MNIST	50K x 784 =**39.2M**	**1.8M**	2	1024
Higgs	10M x 28 =**280M**	**8.5M**	3	2048
Molecular	150K x 2871 =**430M**	**14.2M**	3	2048

Experimental Dimensions

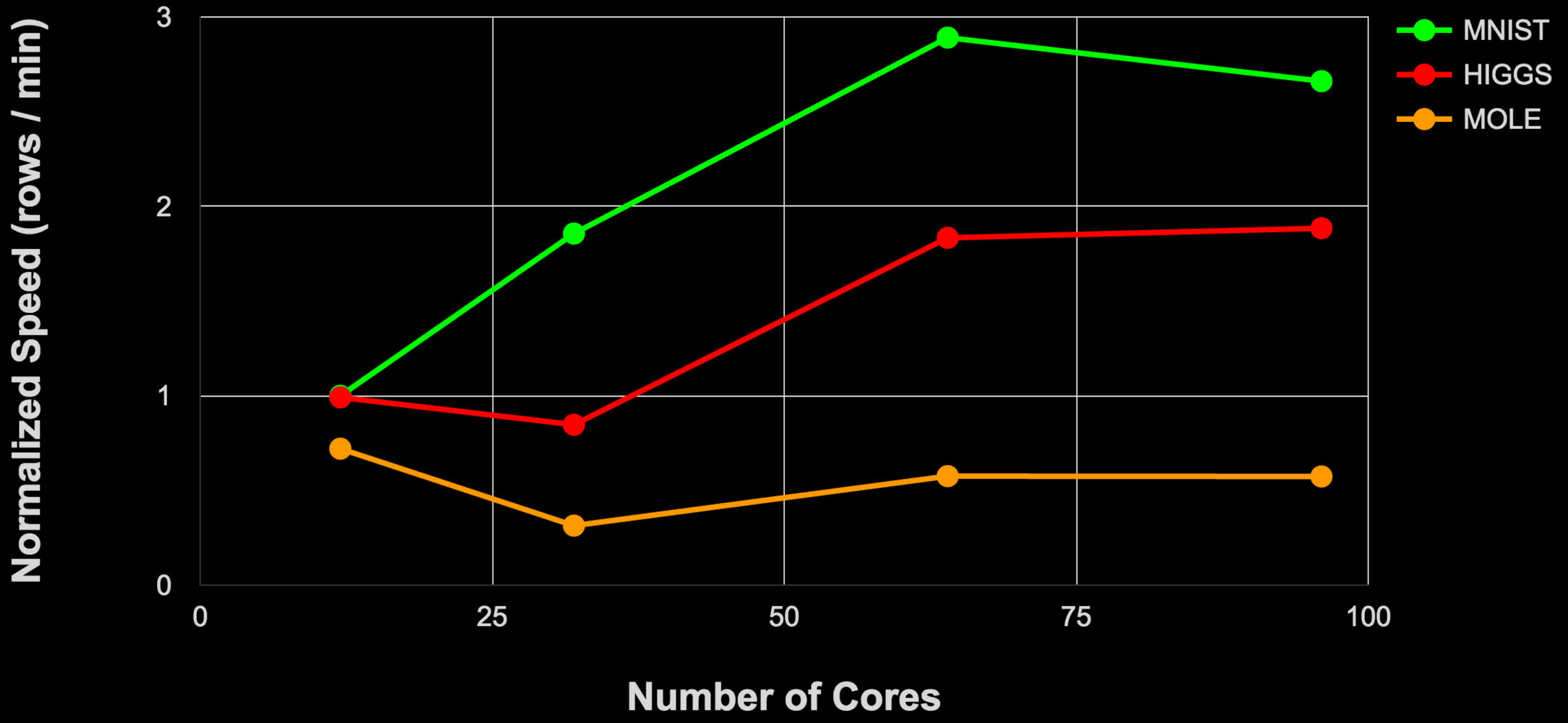
1. Dataset Size
2. Model Size
3. Compute Size
4. CPU vs. GPU

Results

Training Convergence

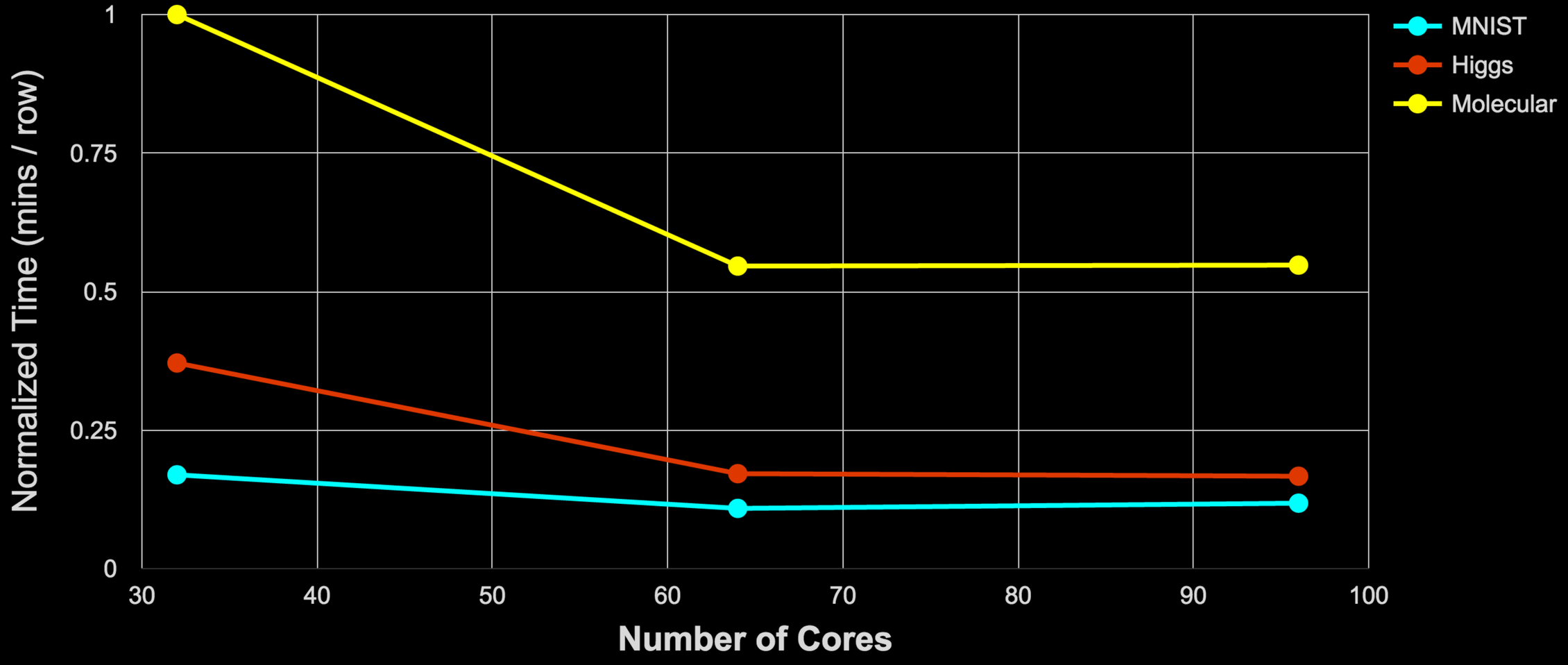


Speed Increases With Compute Size, but Saturates with Model Size

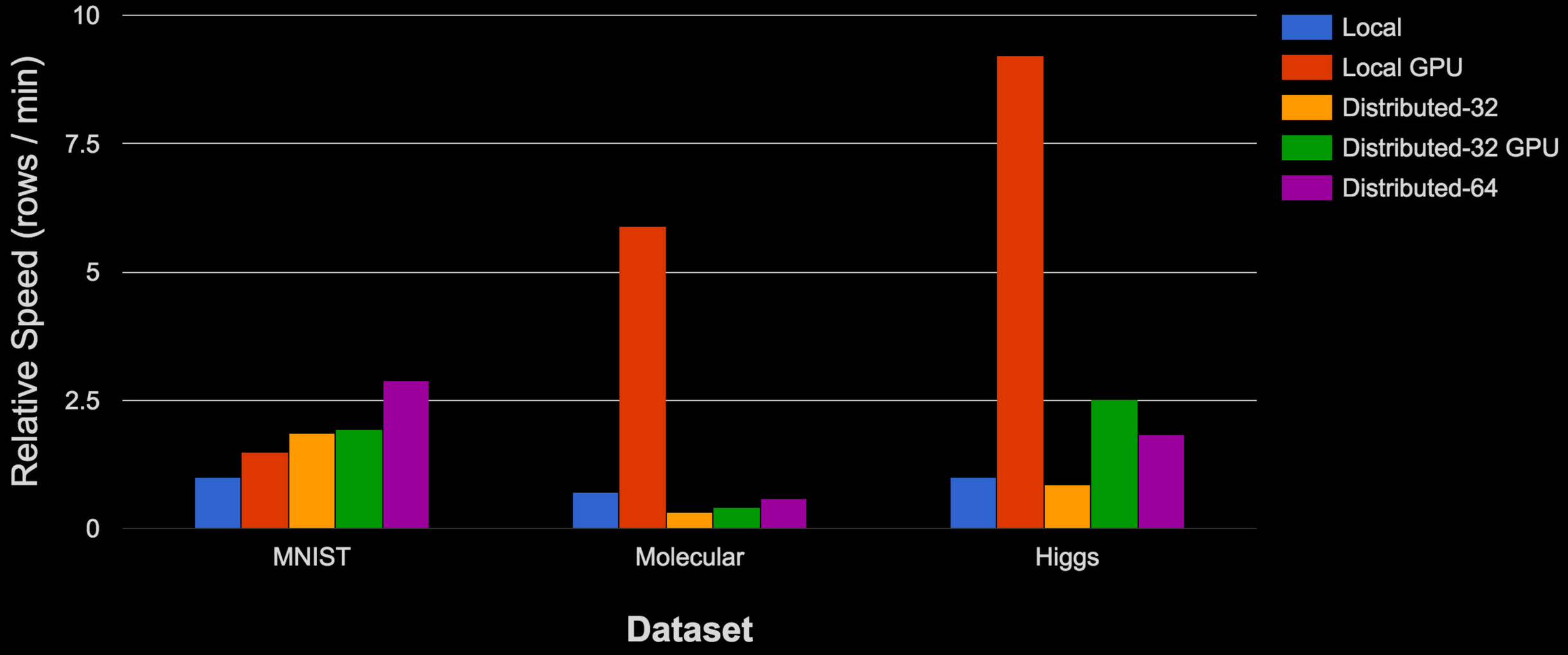


Speed Decreases with Model Size

Residual Time Due To Model Communication



Distribution Speeds Things Up, Except When Communication Dominates. GPU Speeds Things Up.



Lessons Learned

GPU vs CPU

- GPU is 8-10x faster on local; lower gains with Spark but better for larger data sets
- On local: GPU 8–10x faster
- On distributed: GPU 2–3x faster, more with larger datasets
- GPU memory is limited > Performance impact if hit
- AWS has old GPUs, need to build TF from source

Distributed Gradients

- Initial warmup phase on param server needed
- "Wobble" effect may still happen with convergence due to mini-batches
  - Drop "obsolete" gradients
  - Reduce learning rate and initial parameters
  - Reduce mini-batch size
- Work to maximize network throughput, e.g., model compression

Deployment Configuration

- Should run only 1 TF instance per machine
- Compress gradients/parameters (e.g., as binaries) to reduce network overhead
- Batch-size tradeoff: convergence vs. communication overhead

Conclusions

Conclusions

- This implementation best for large datasets & small models
  - For large models, we're looking into
    a) model parallelism b) model compression
- If data fits on single machine, single-machine GPU would be best (if you have it)
- For large datasets, leverage both speed and scale using Spark cluster with GPUs
- Future Work: a) Tachyon b) Ensembles (Hadoop Summit - June 2016)

ARIMO
Data Intelligence for All

VISIT US
BOOTH
K8

FAST COMPANY
Top 10 World's Most Innovative Companies in Data Science

We're OPEN SOURCING our work at

https://github.com/adatao/tensorspark