Hops

Introduction to Hopsworks

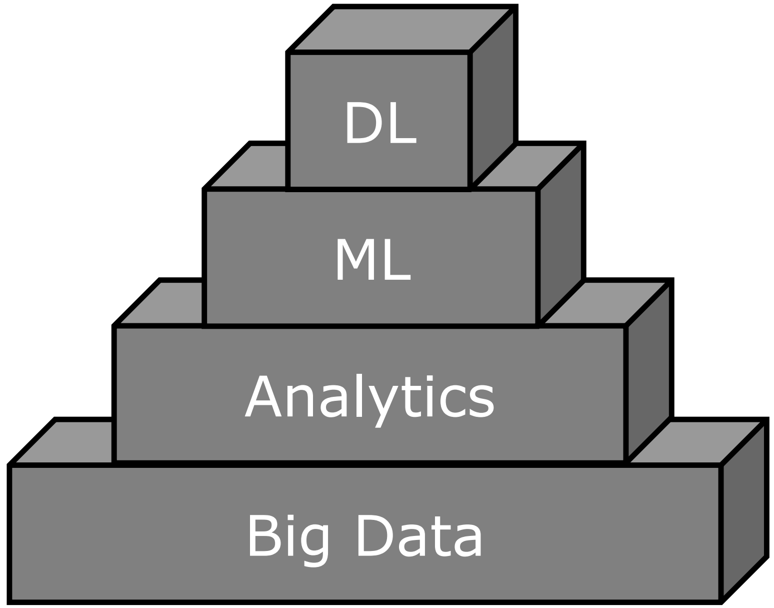
Jim Dowling
Assoc Prof @ KTH
CEO @ Logical Clocks AB

KTH ID2223, 16th Nov 2018

Deep Learning needs Big Data

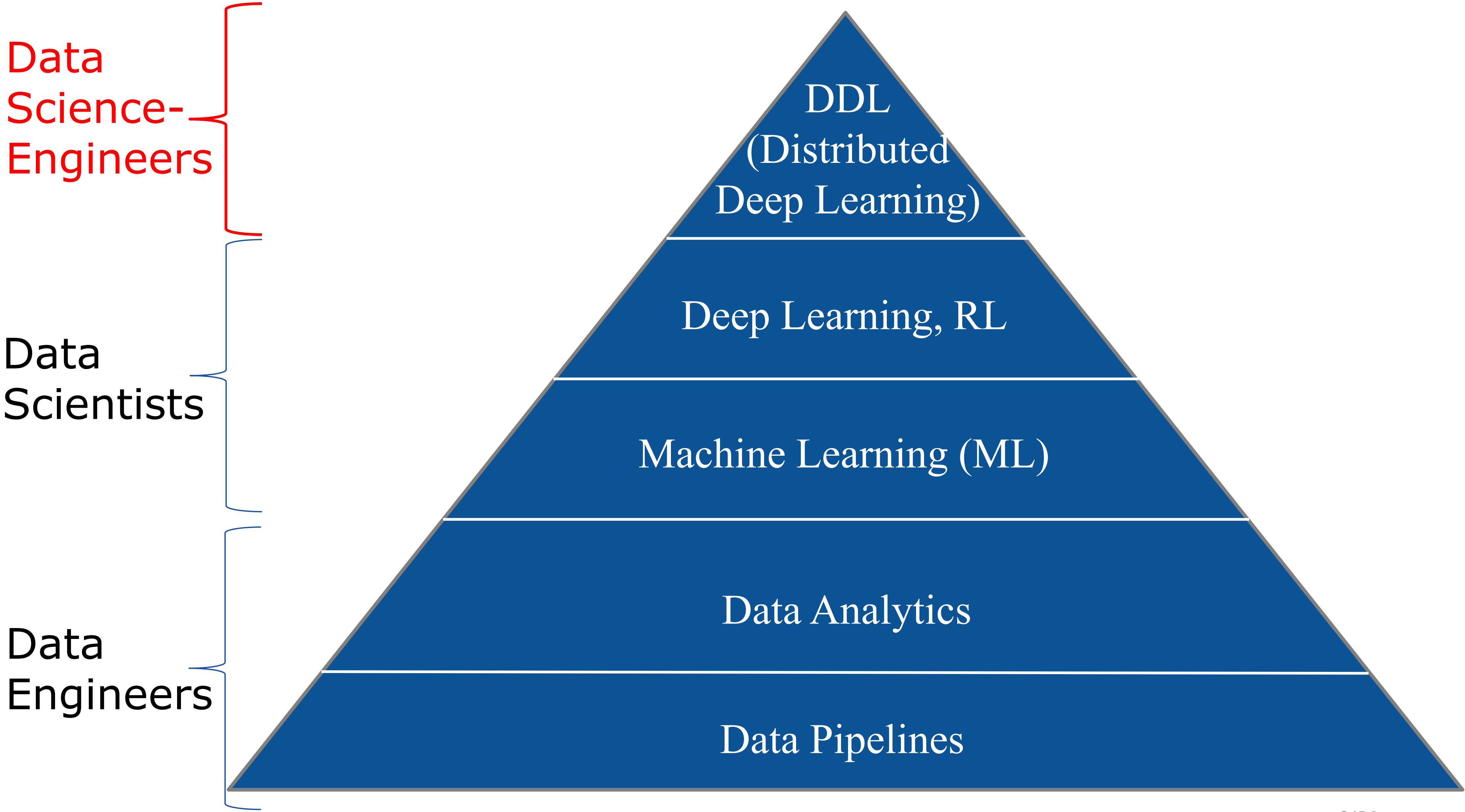
- Lots of data needed to train machine learning (**ML**) models

- Recent advances in ML down to Deep Learning (**DL**)
  - Deep Learning needs hardware accelerators (GPUs)



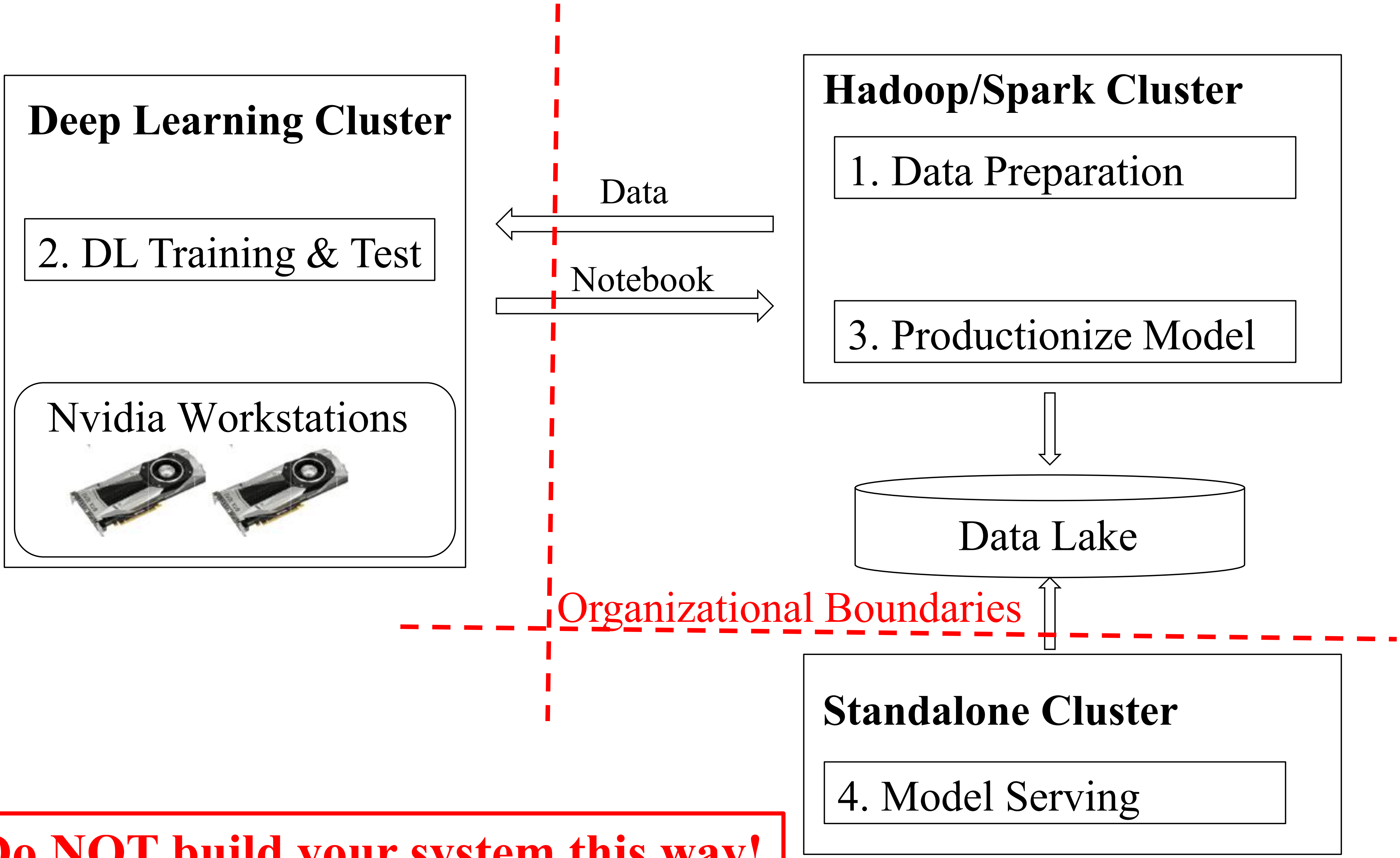
This is a GPU
(graphical processing unit)

AI Hierarchy of Needs



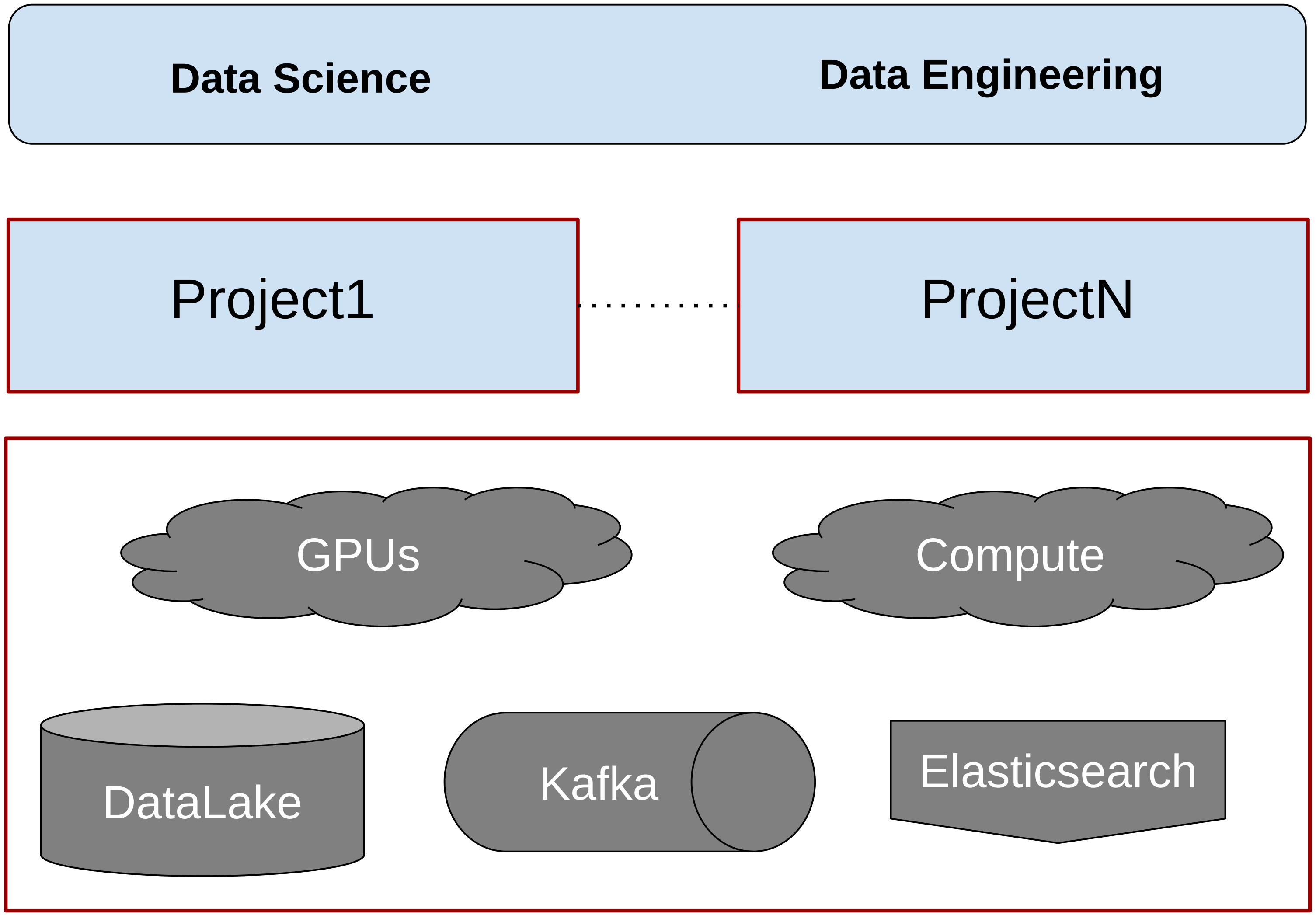
State-of-the-Art for On-Premise ML Pipelines

Separate Data Lake and Deep Learning Clusters*

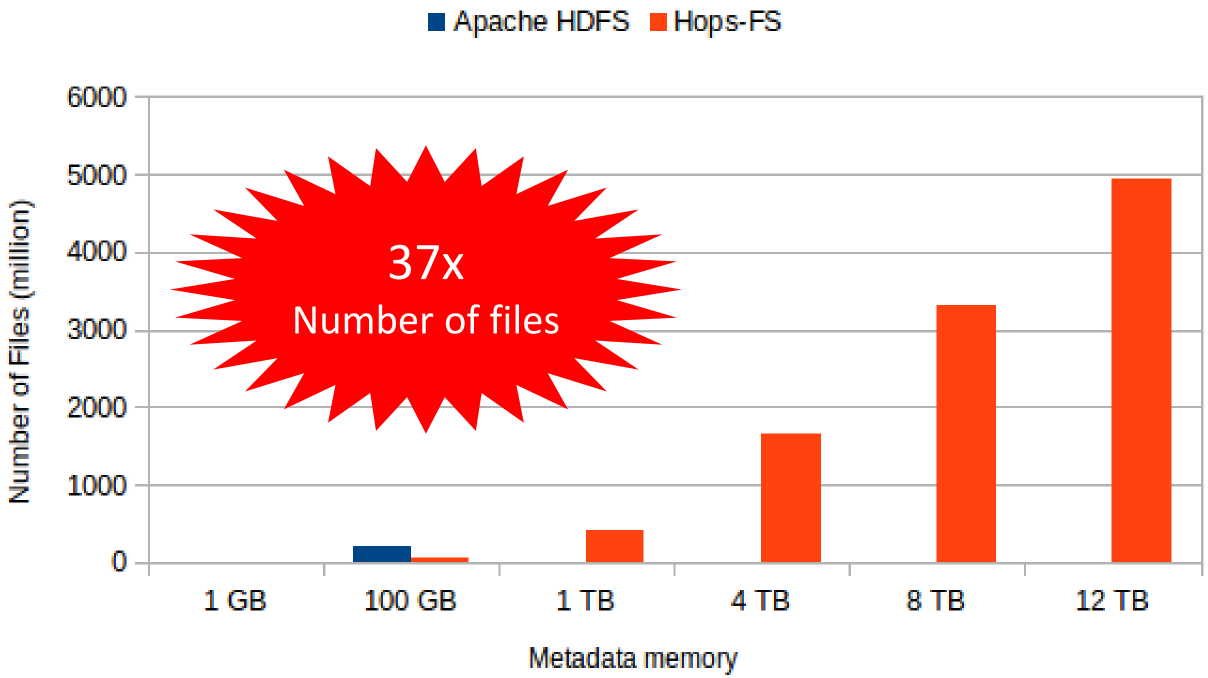


***Do NOT build your system this way!**

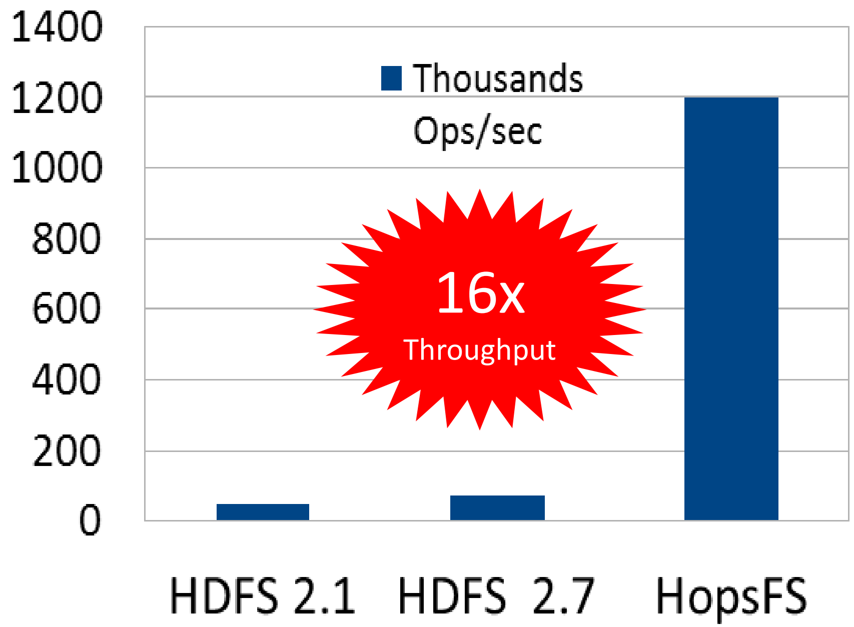
Hopsworks: Single ML and Big Data Cluster



HopsFS: Next Generation HDFS*



Bigger



Faster

IEEE Scale Challenge Winner (2017)

*https://www.usenix.org/conference/fast17/technical-sessions/presentation/niazi

GPU Resource Requests in Hops YARN

4 GPUs on any host

10 GPUs on 1 host

100 GPUs on 10 hosts with 'Infiniband'

HopsYARN

Hops supports a Hetrogenous Mix of GPUs

www.hops.site

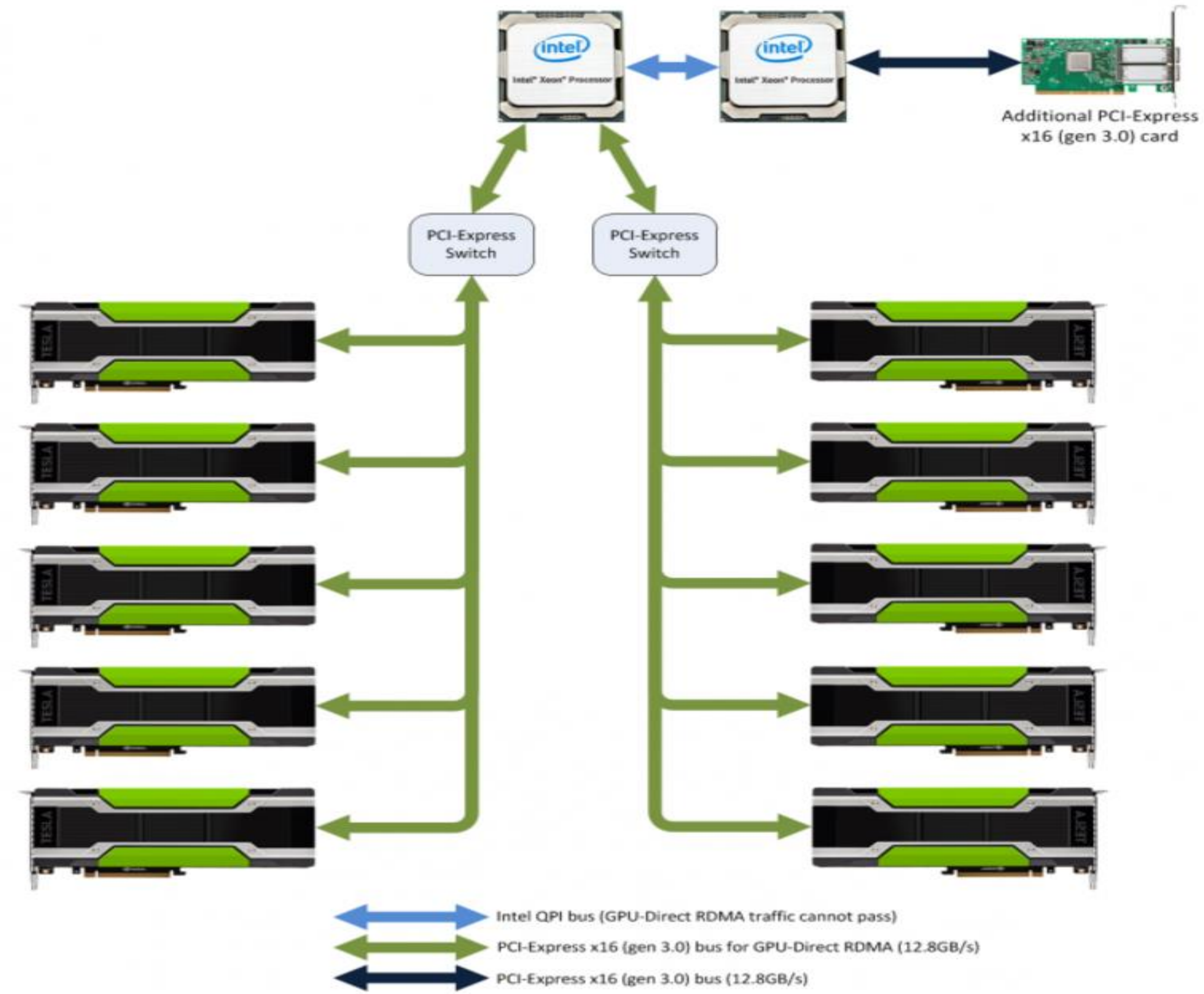
- www.hops.site
  - ~1000 cores
  - ~12 TB RAM
  - ~1 PB storage
  - 20 GPUs (Nvidia 1080 Ti)
  - 4 GPUs (Nvidia 1080)

- **RISE SICS ICE**
  - Research and test environment

https://www.sics.se/projects/sics-ice-data-center-in-lulea

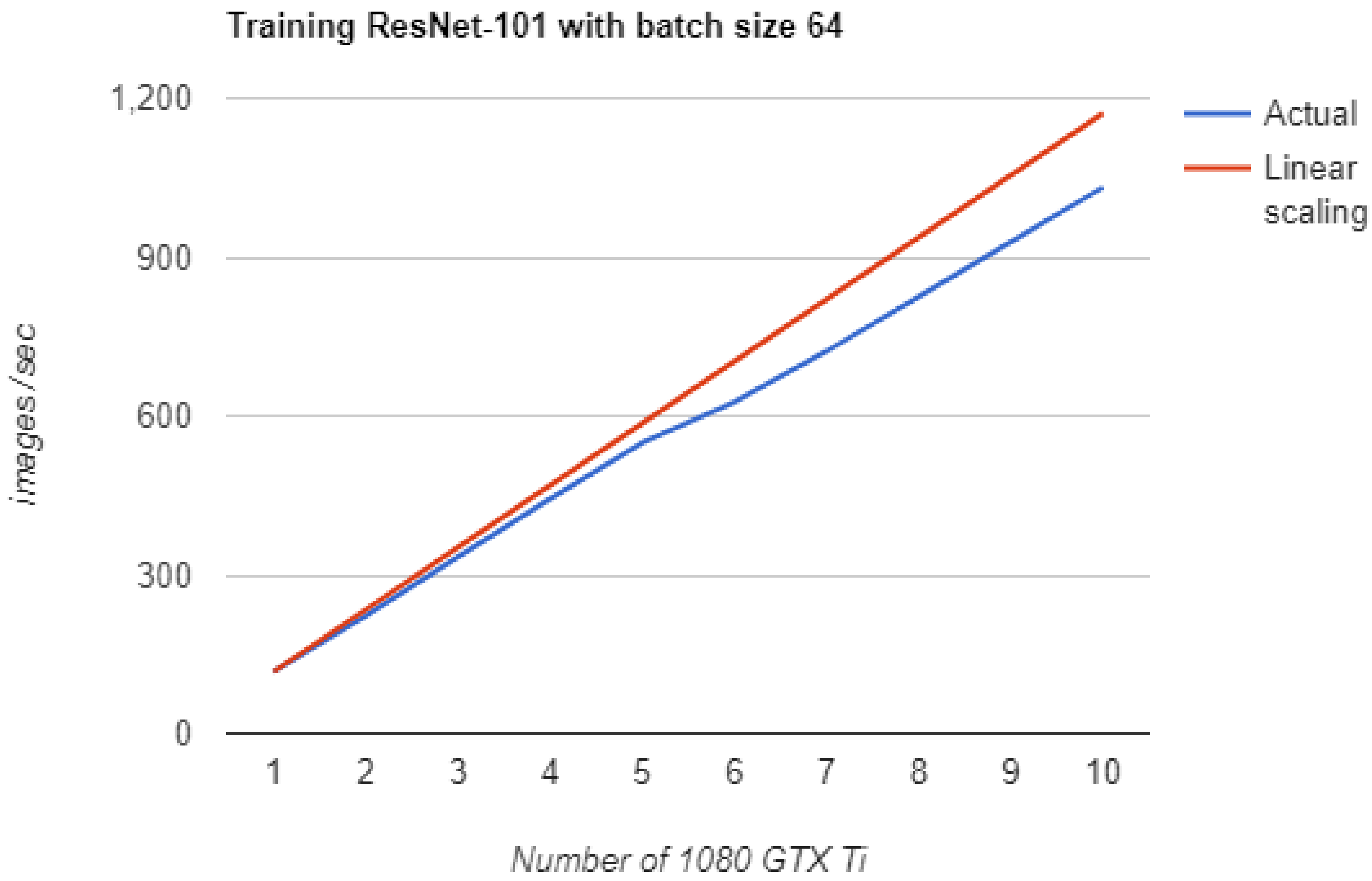
DeepLearning 11 Server

PCI-E – 16 GB/s



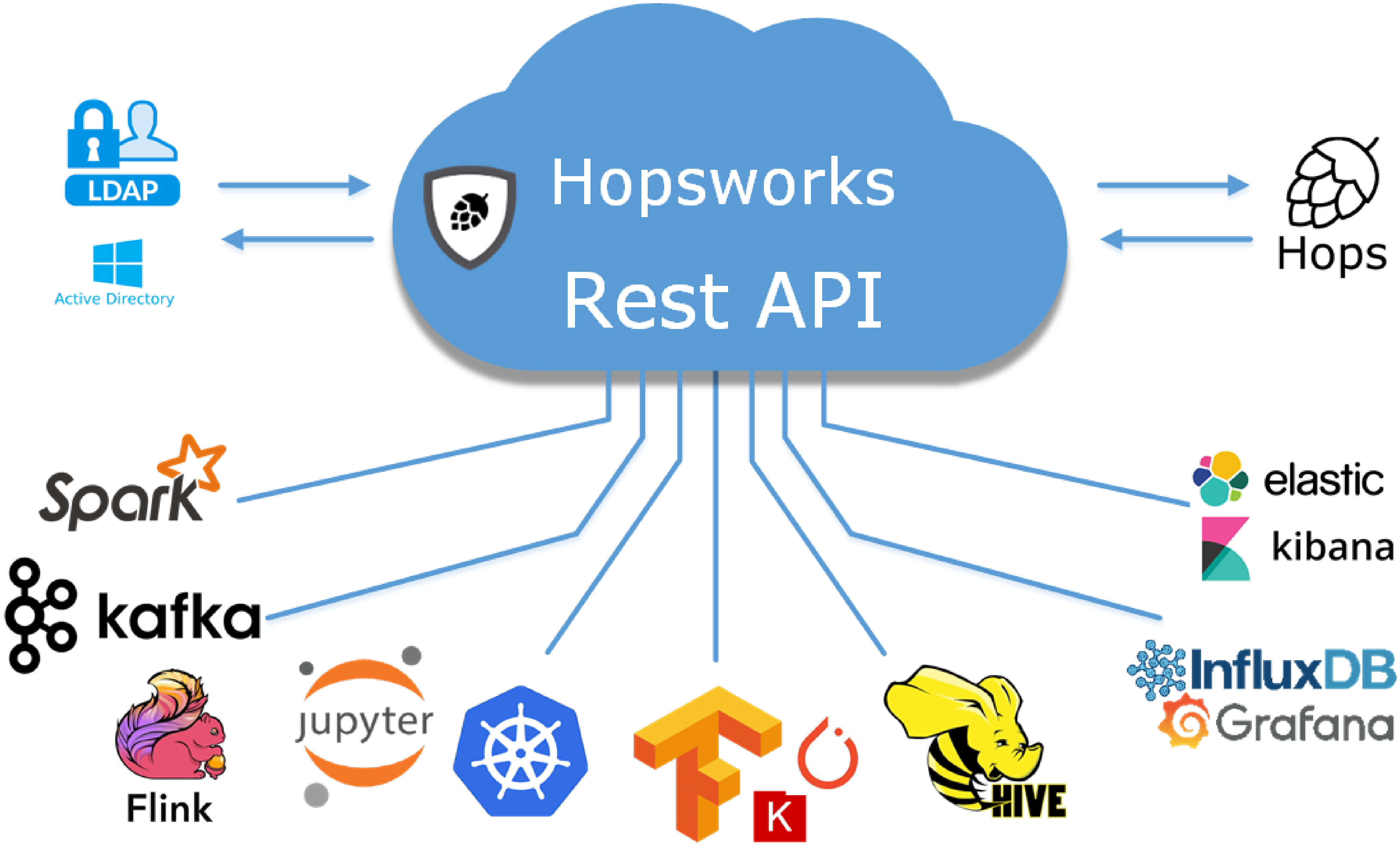
[Images from: https://www.microway.com/product/octoputer-4u-10-gpu-server-single-root-complex/]

TensorFlow/Hops on 10 1080Ti GPUs



https://www.oreilly.com/ideas/distributed-tensorflow

Hopsworks*



*Ismail et al, Hopsworks: Improving User Experience and Development on Hadoop with Scalable, Strongly Consistent Metadata, ICDCS 2017

Hopsworks: Projects, Users, Datasets

Projects

Hopsworks

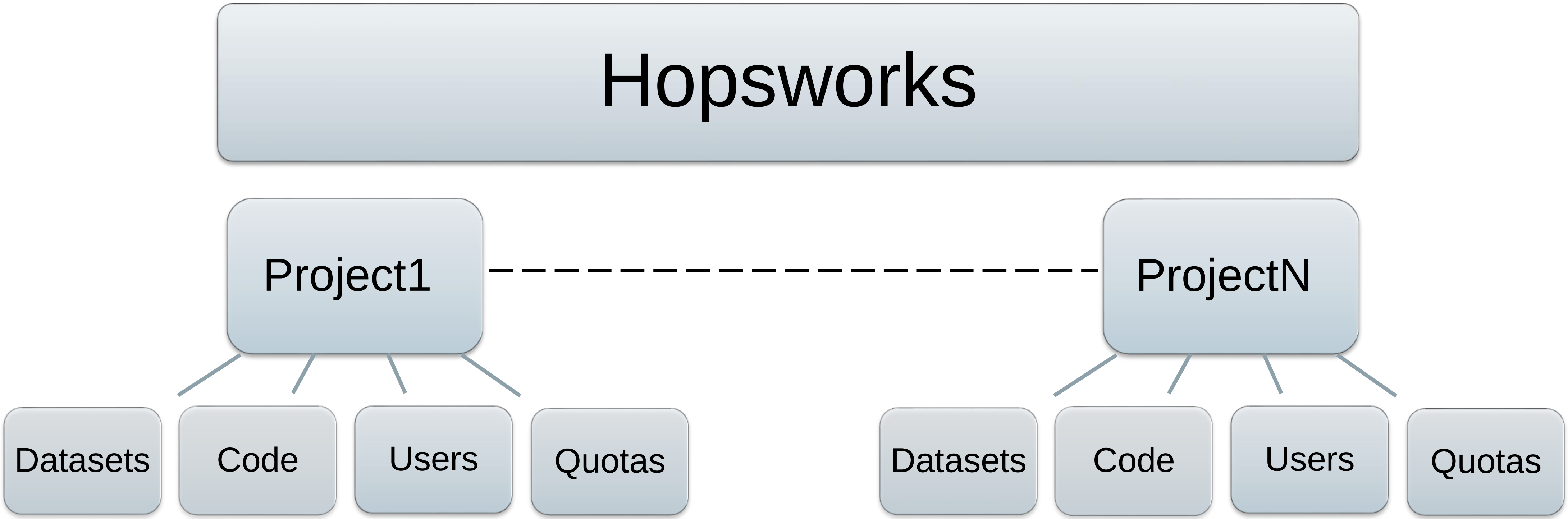
Project1

Project2

Project N

Projects are “safety deposit vaults” for data that only authorized users can access and open.

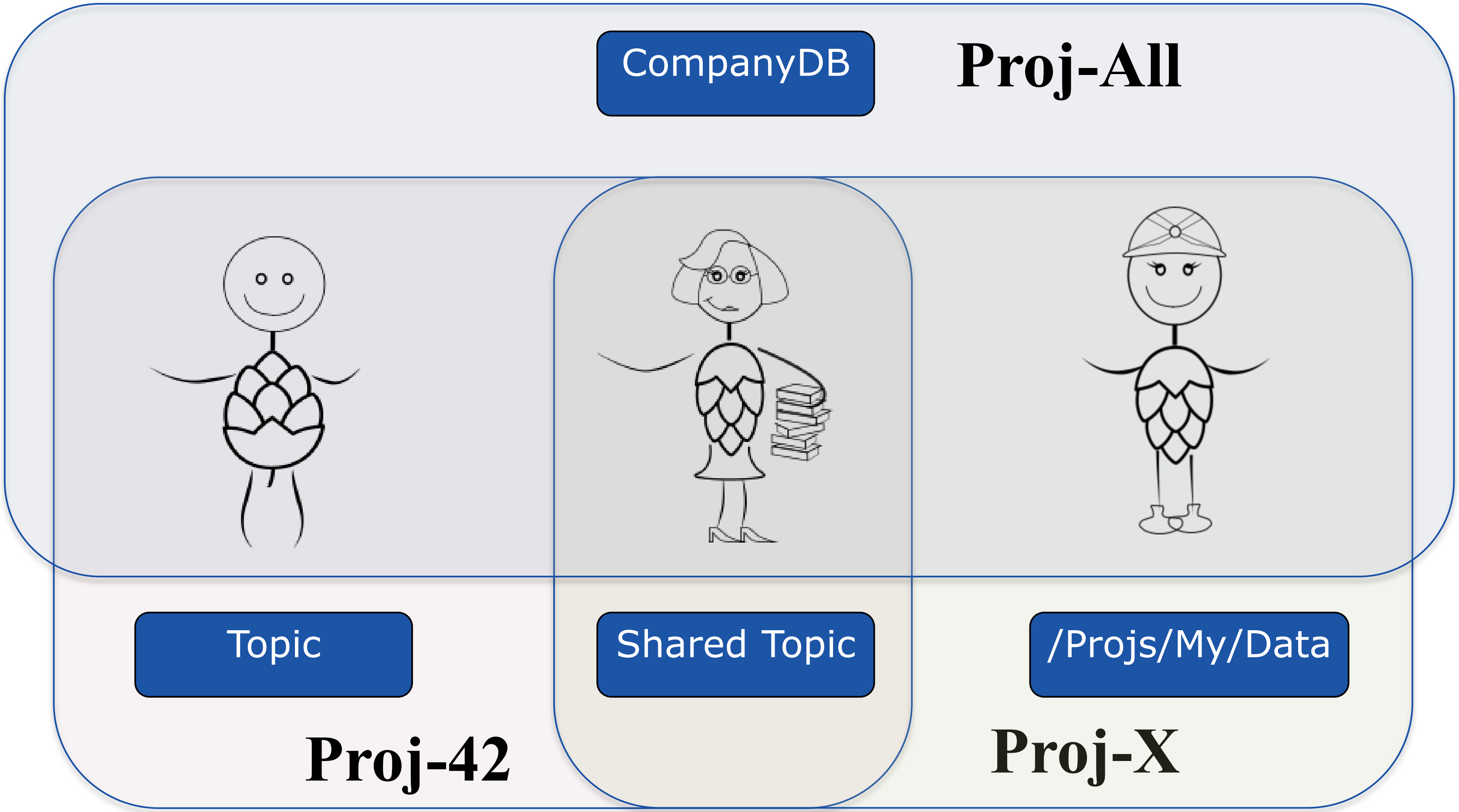
Projects



Hopsworks
Project1
ProjectN
Datasets
Code
Users
Quotas
Datasets
Code
Users
Quotas

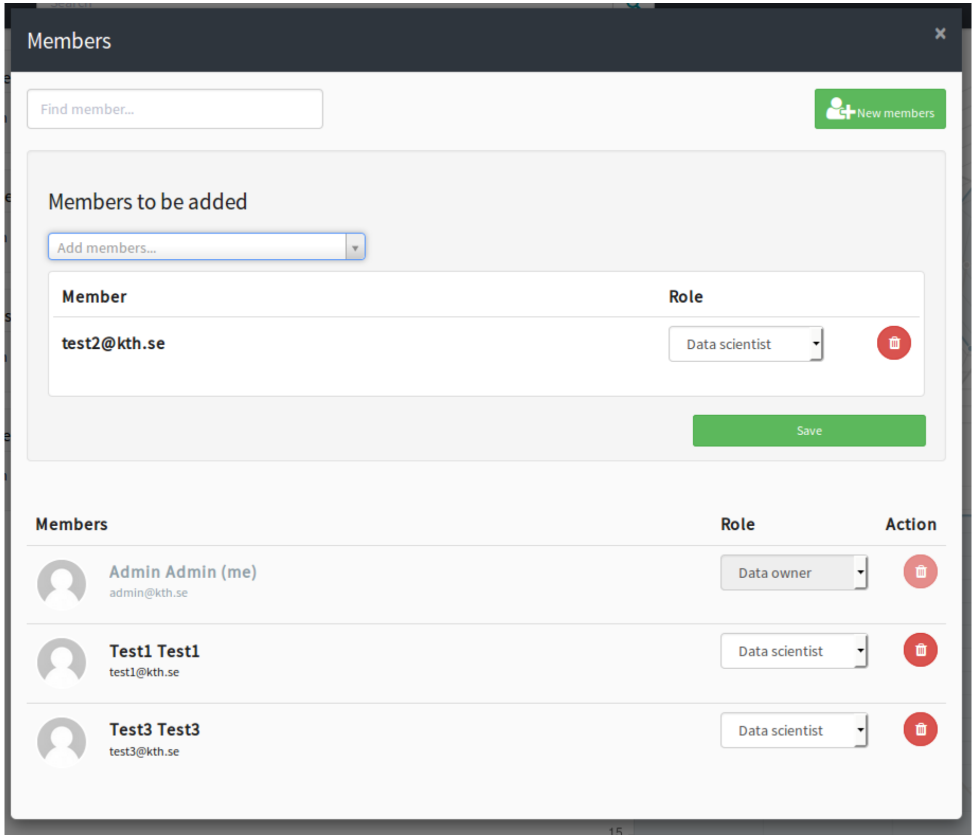
Projects

A **Project** is a Grouping of **Users** and **Data**



Ismail et al, Hopsworks: Improving User Experience and Development on Hadoop with Scalable, Strongly Consistent Metadata, ICDCS 2017

Project Roles



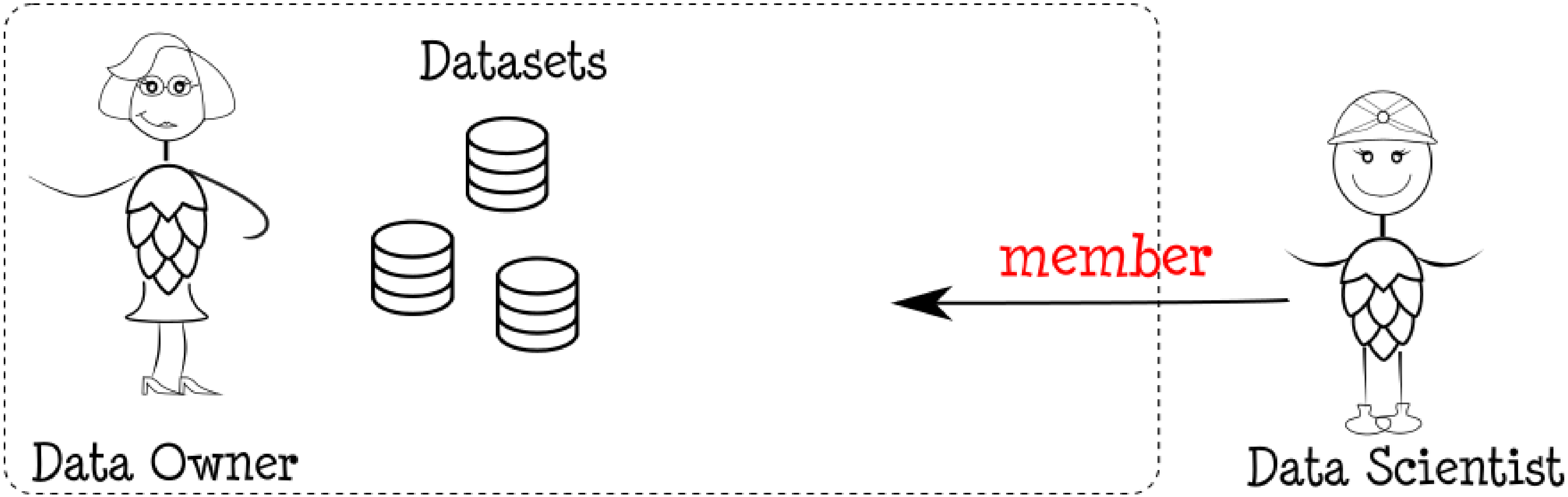
- Data Owner Privileges
  - Import/Export data
  - Manage Membership
  - Share DataSets,Topics
- Data Scientist Privileges
  - Write and Run code

We delegate administration of privileges to users

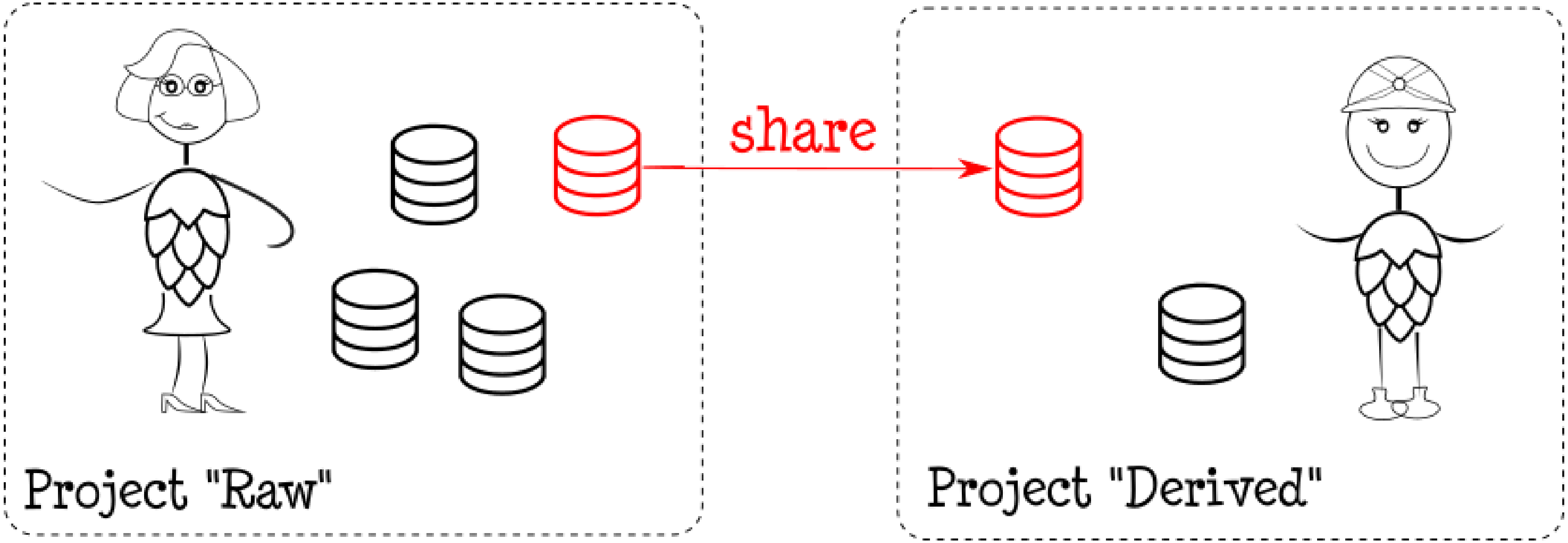
Projects in HopsFS (HDFS)

- Create a Project with some globally unique name
- HDFS:
  - /Projects/project_name
  - Storage space: ~200GB
- Compute and GPUs (YARN):
  - 100+ hrs
- Kafka
  - 100 topics

Manage Projects like GitHub

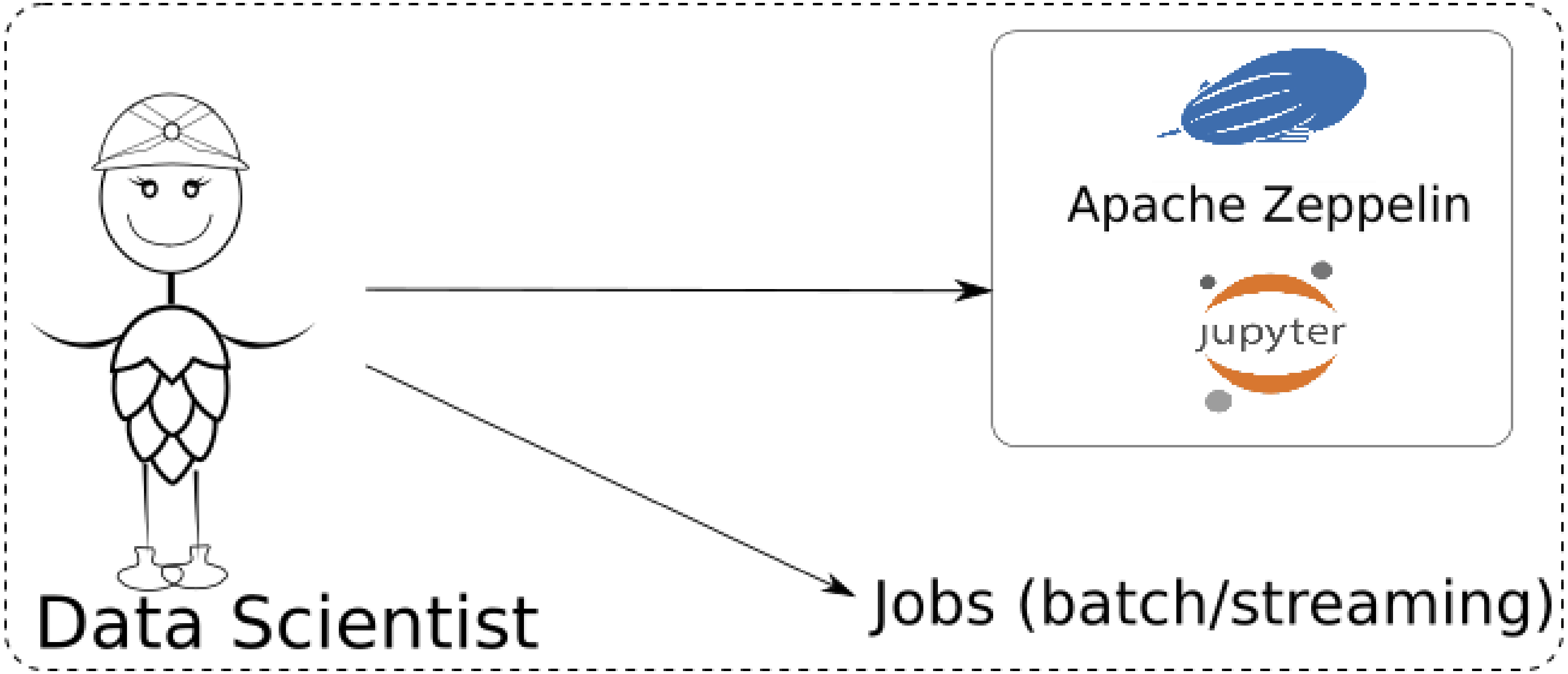


Share like in Dropbox

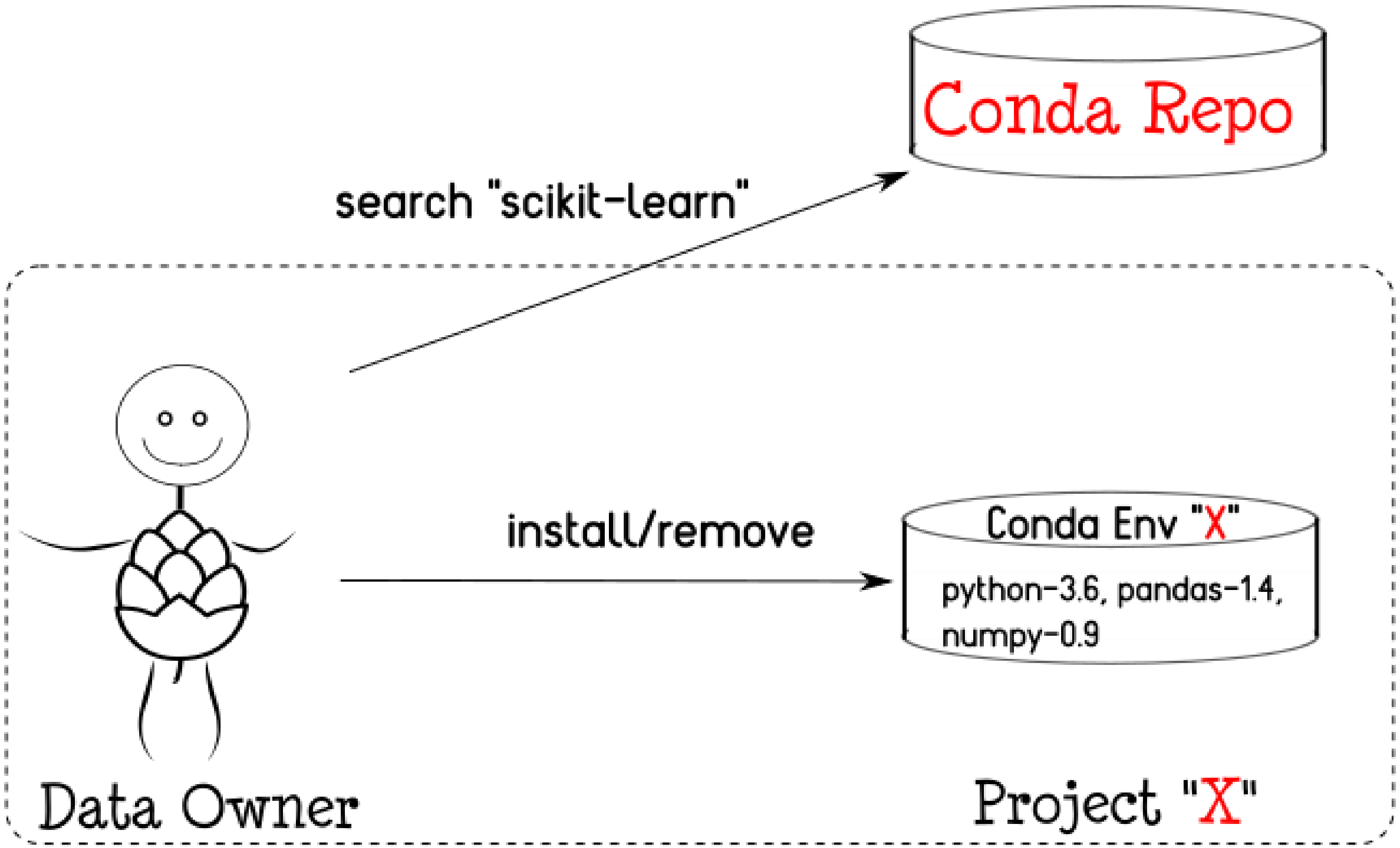


Share any Data Source/Sink: HDFS Datasets, Kafka Topics, etc

Workflow/Jobs and Notebook Support



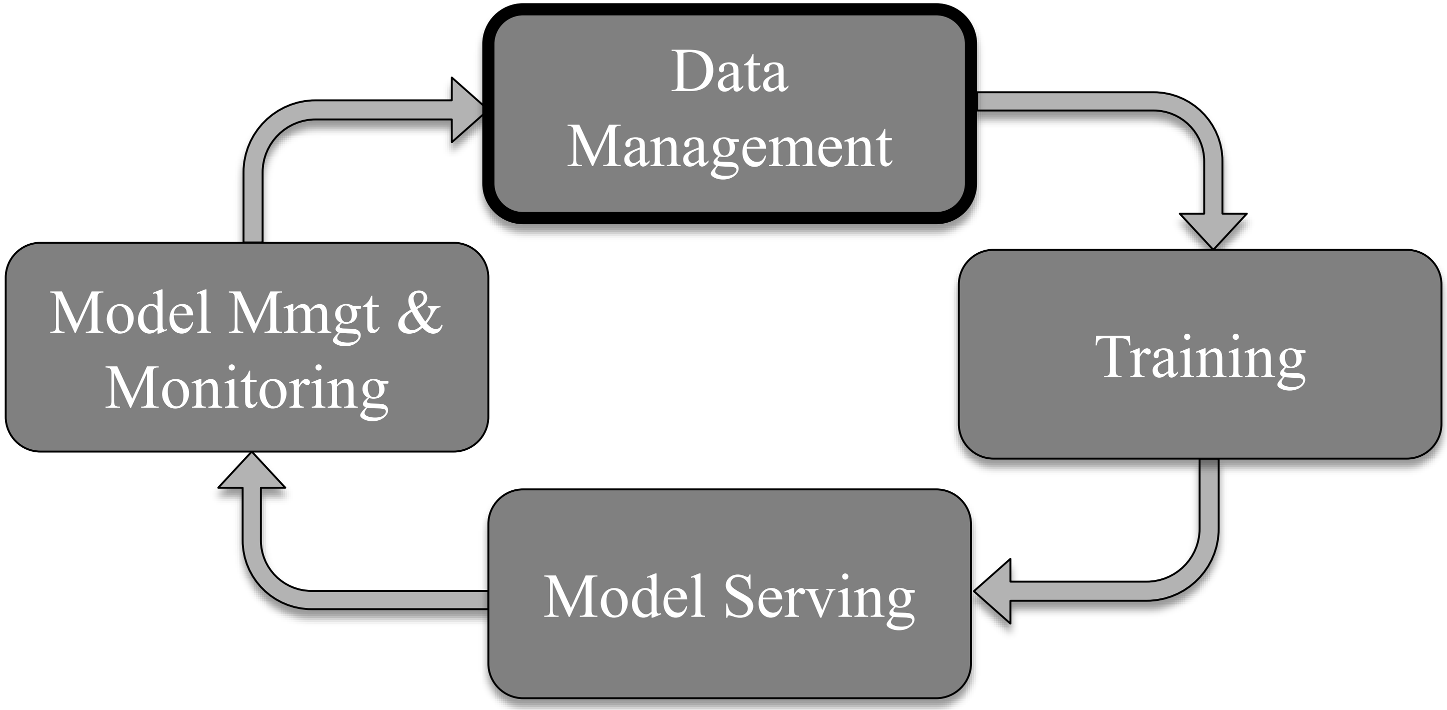
Custom Python Environments with Conda



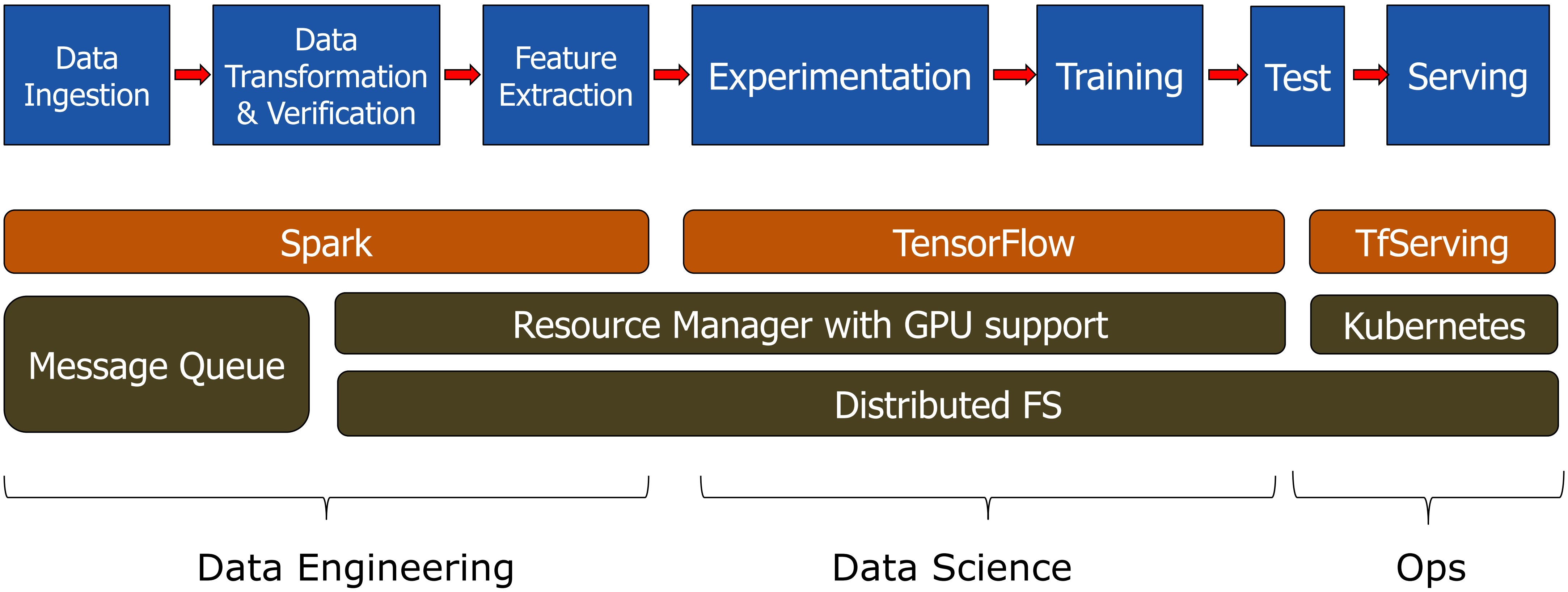
Python libraries are usable by Spark/Tensorflow

Machine Learning Pipelines in Hops

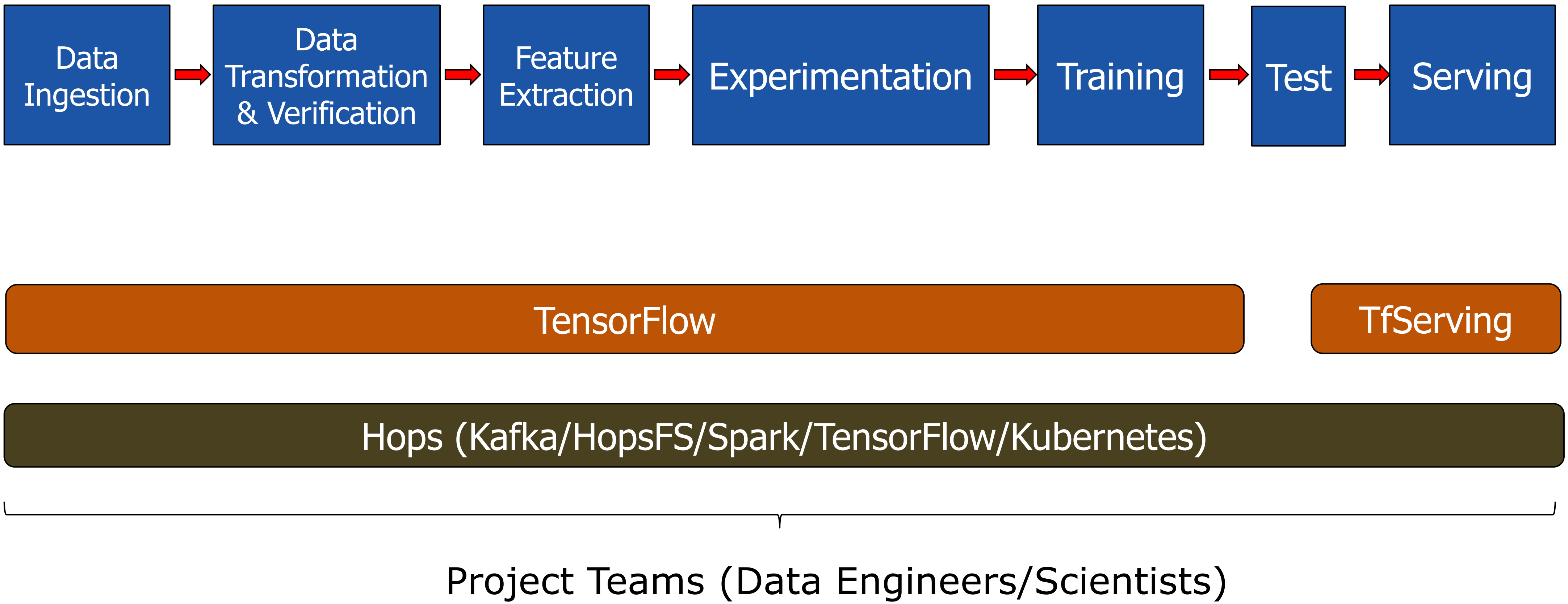
The Machine Learning Lifecycle



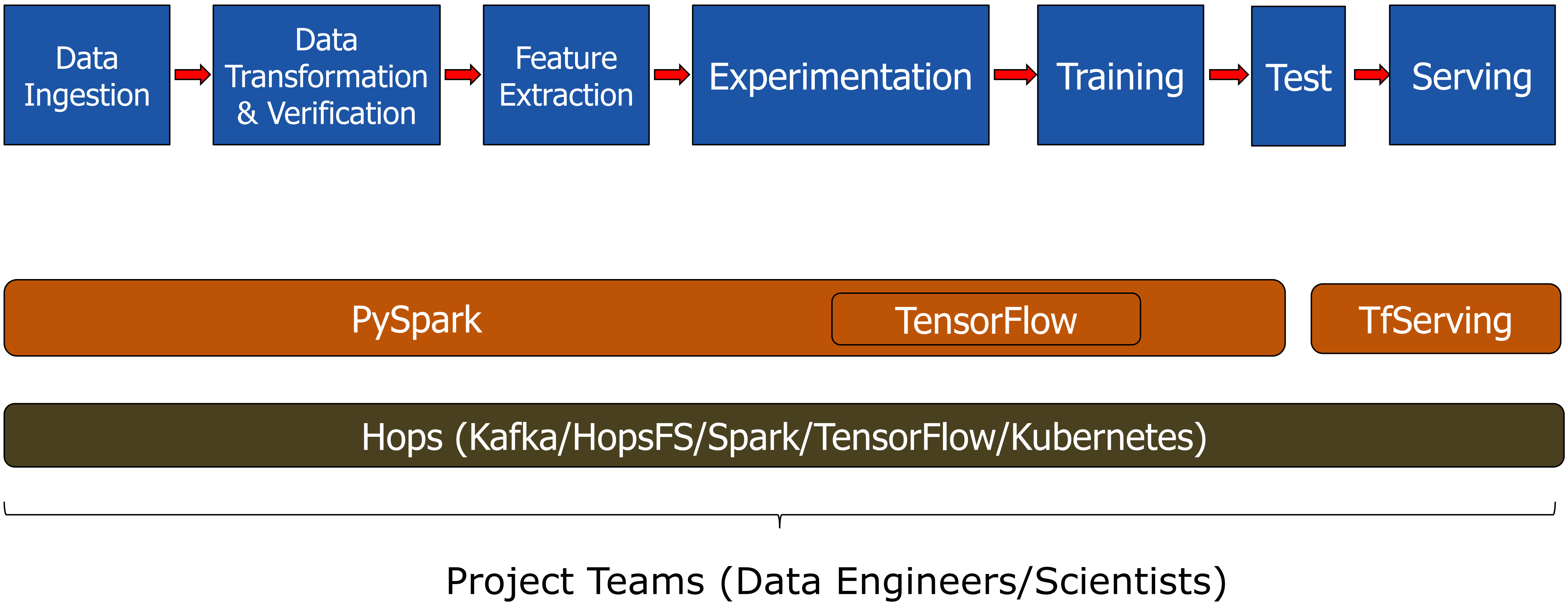
A Scale-Out Machine Learning Pipeline



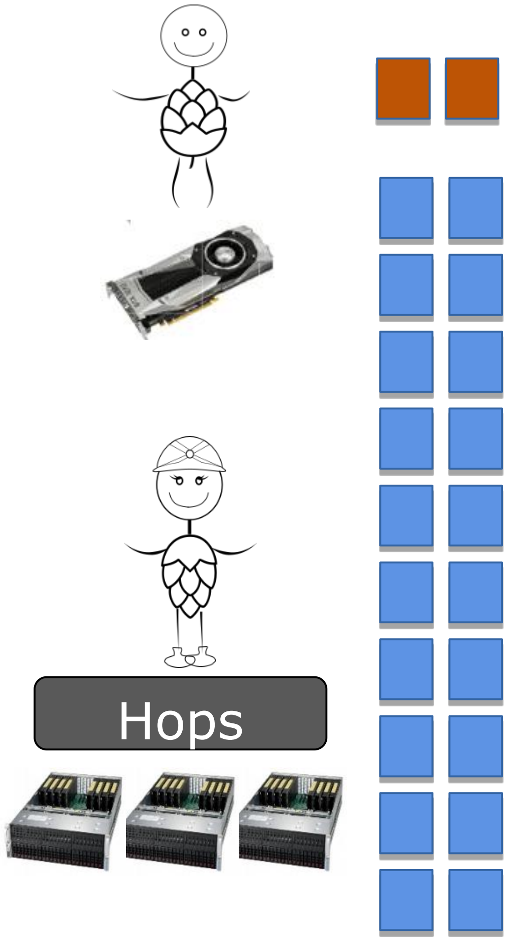
Hops Small Data ML Pipeline

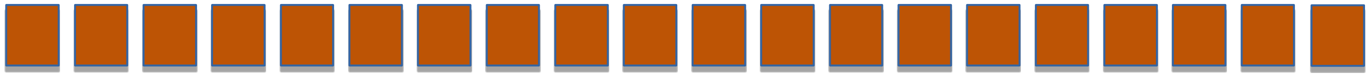


Hops Big Data ML Pipeline



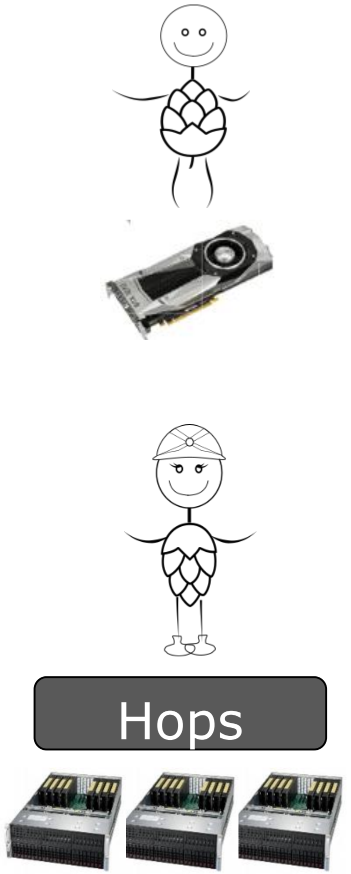
Parallel Experiments





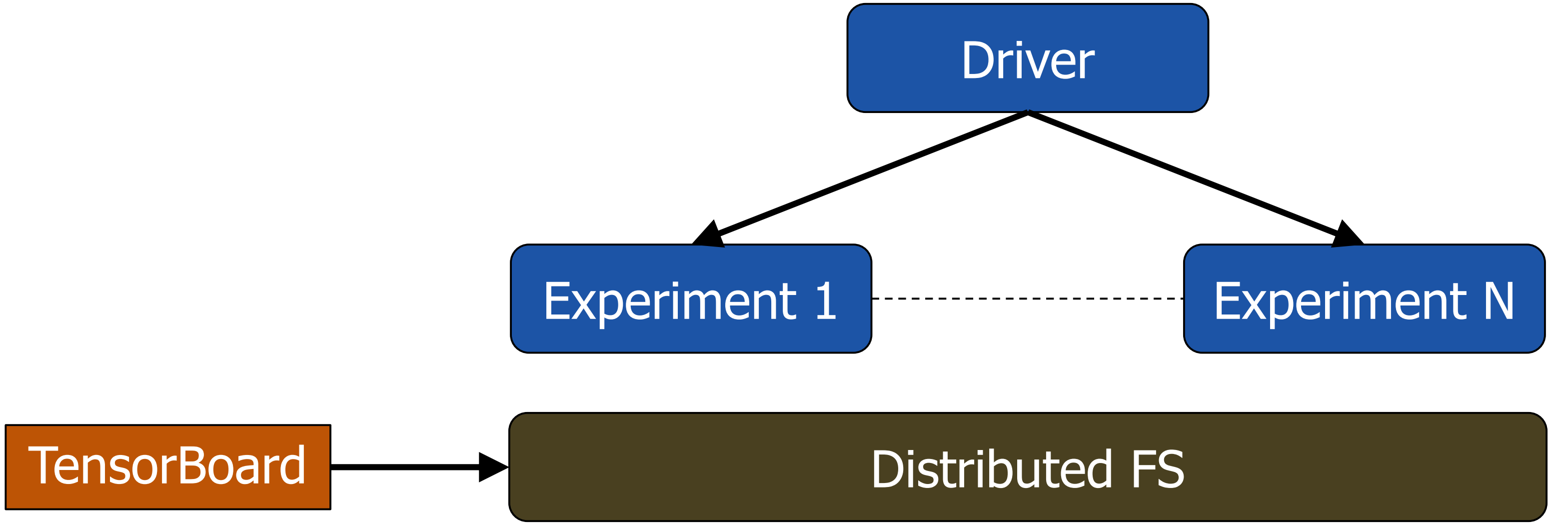
The Outer Loop (hyperparameters)*:*
“I have to run a hundred experiments to find the best model,” he complained, as he showed me his Jupyter notebooks. “That takes time. Every experiment takes a lot of programming, because there are so many different parameters.
[Rants of a Data Scientist]

Distributed Training



The Inner Loop (training):
"All these experiments took a lot of computation — we used hundreds of GPUs/TPUs for days. Much like a single modern computer can outperform thousands of decades-old machines, we hope that in the future these experiments will become household."
[Google SoTA ImageNet, Cifar-10, March18]

Need for a Distributed Filesystem



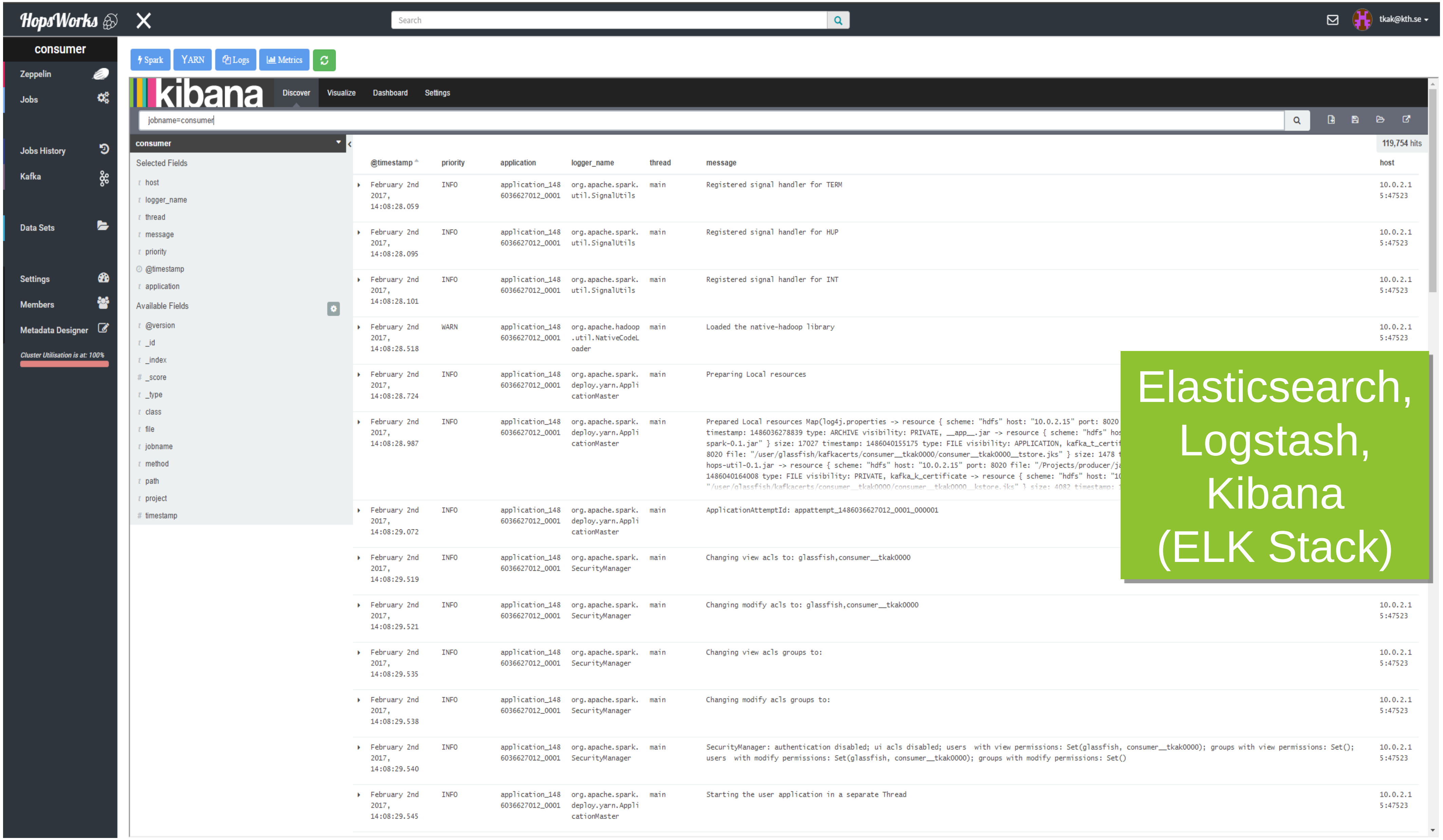
Training/test datasets,
experiment results,
experiment configurations,
model checkpoints,
hyperparameter optimization.

Coding Machine Learning Pipelines in Hops

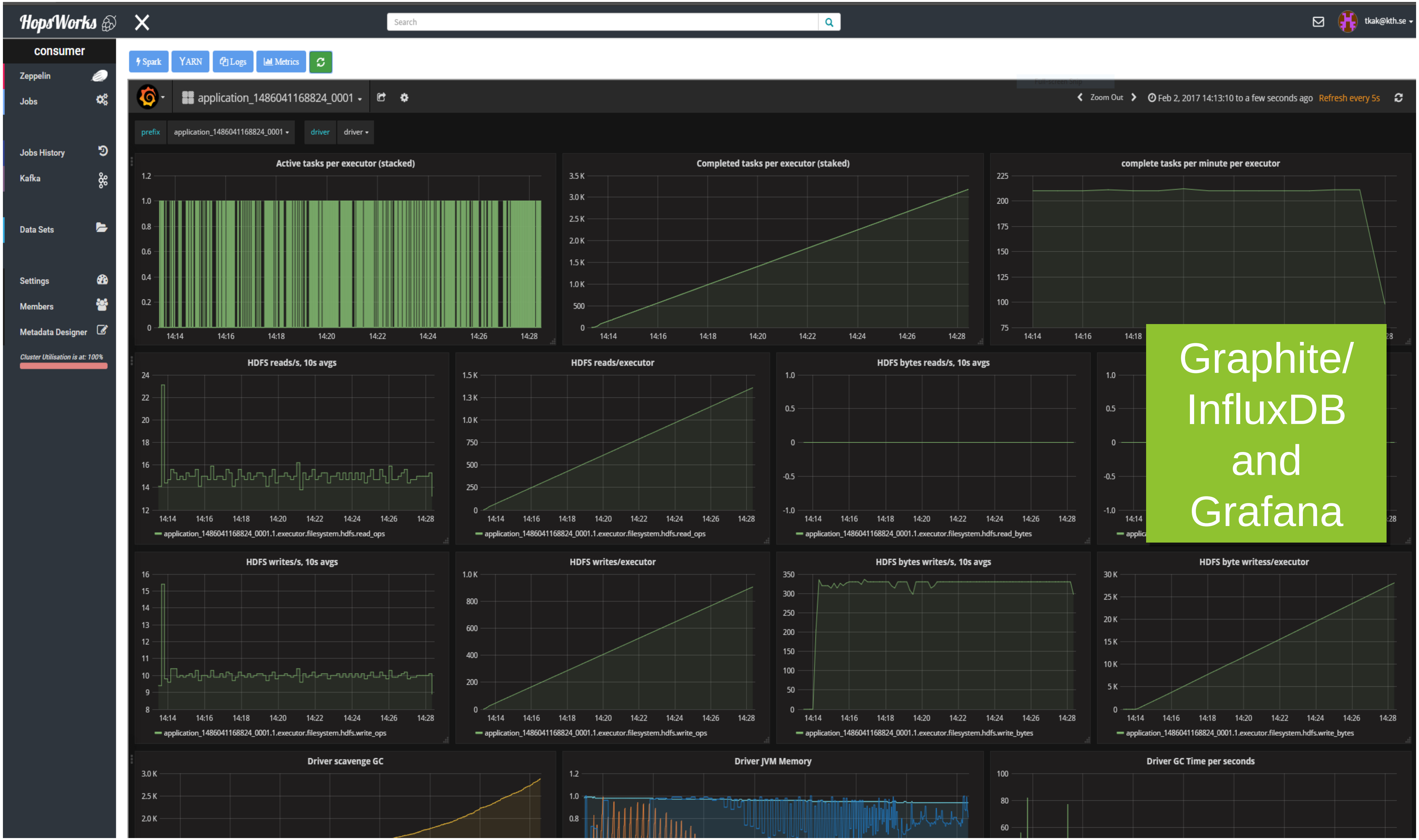
Other Development Tools

- Real-time Logging with Kibana
- Performance Monitoring with Grafana (InfluxDB)
- TensorFlow Debugging with TensorBoard

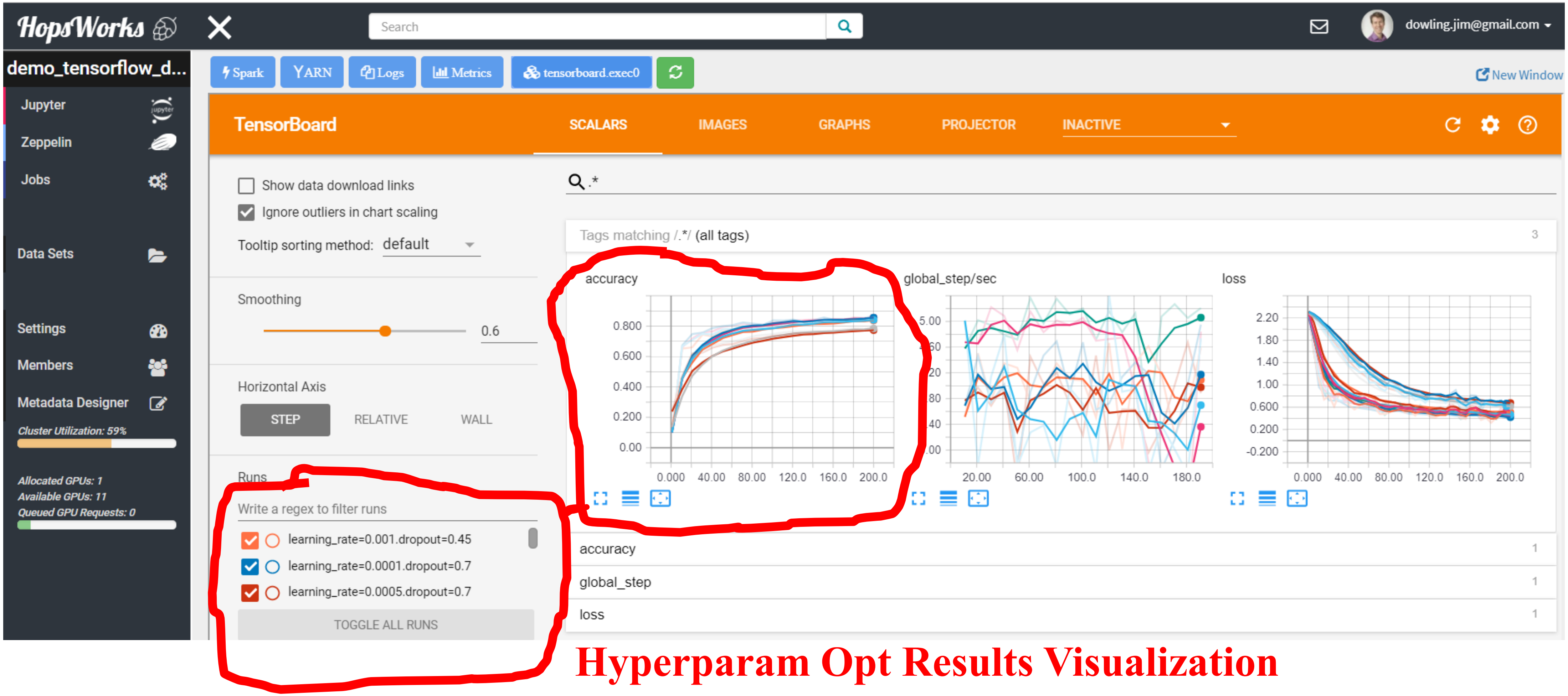
Realtime Logs



Resource Monitoring/Alerting



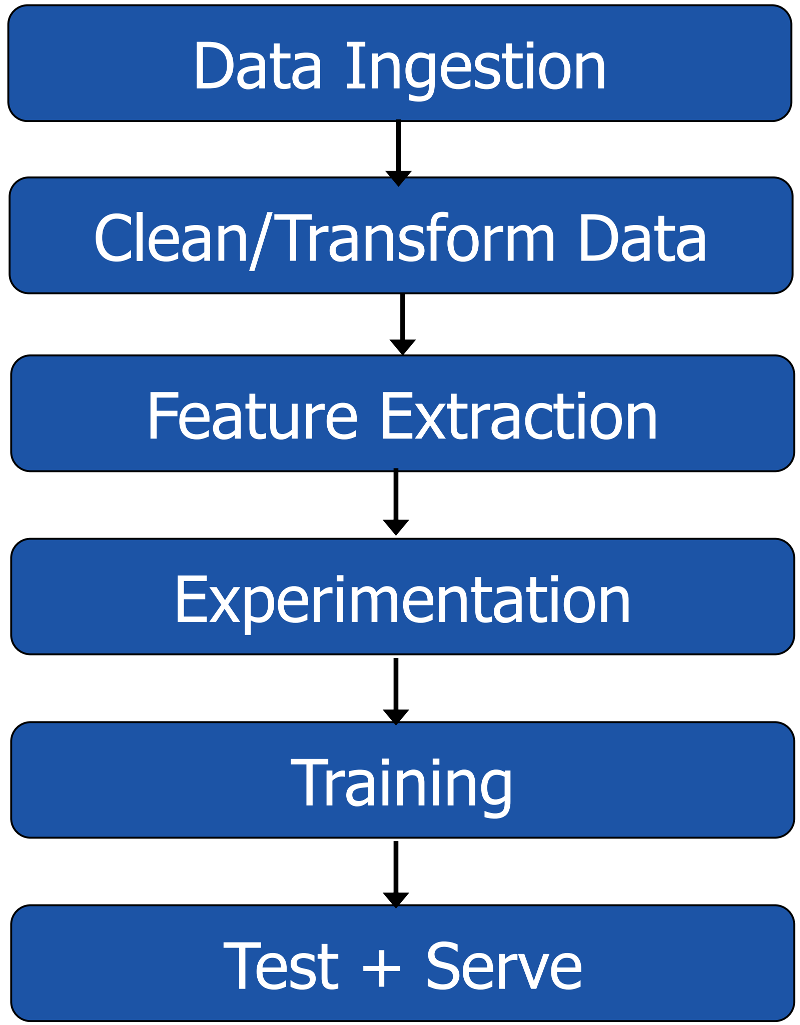
TensorBoard



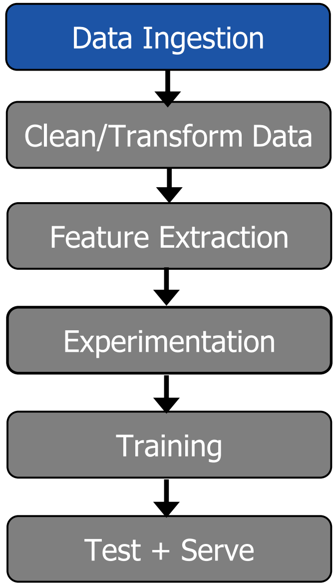
Hyperparam Opt Results Visualization

Hops API

- Python (also Java/Scala)
  - Manage tensorboard, Load/save models in HDFS
  - Distributed Training
  - Parallel experiments
    - Hyperparameter search0
    - Model Architecture Search with Genetic Algorithms
  - Secure Streaming Analytics with Kafka/Spark/Flink
    - SSL/TLS certs, Avro Schema, Endpoints for Kafka/Zookeeper

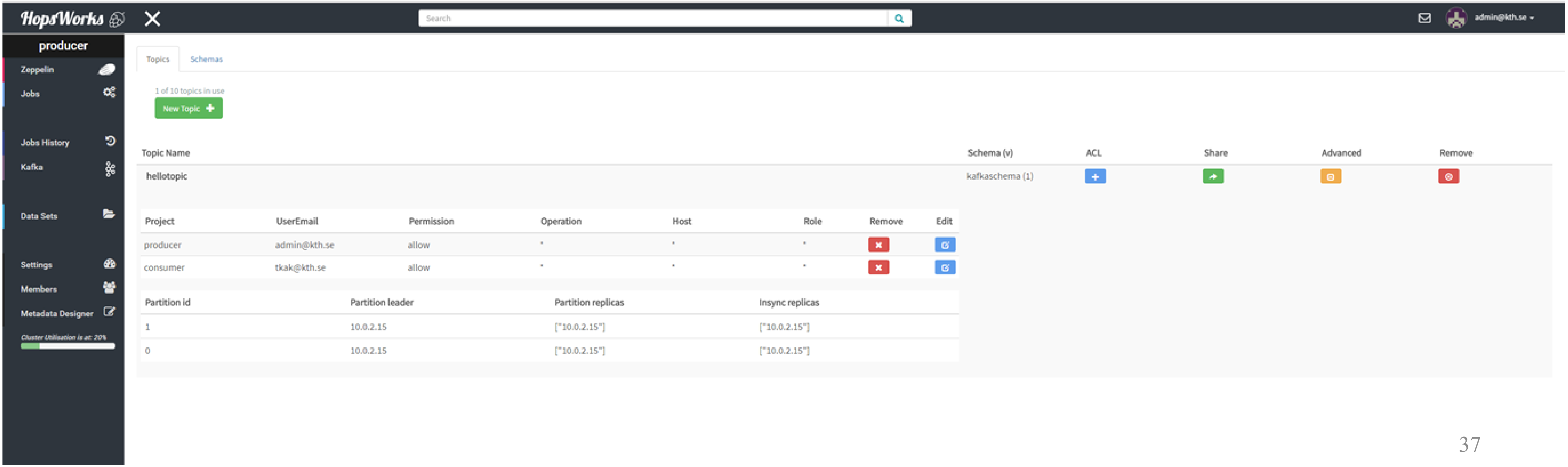


Kafka Self-Service UI

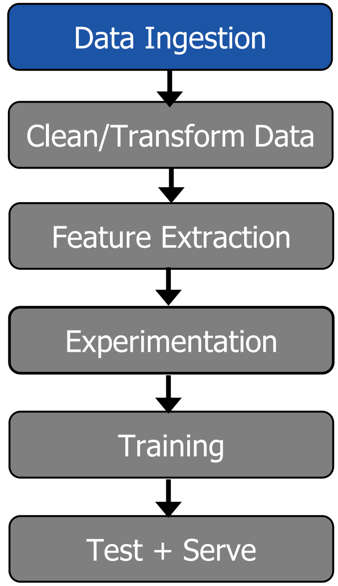


Manage & Share

- Topics
- ACLs
- Avro Schemas



Data Ingestion (Kafka)



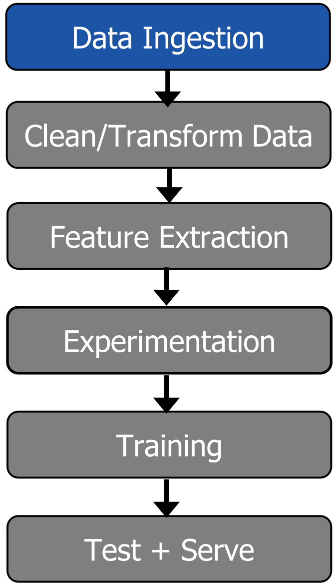
The Hops API simplifies consuming events from and producing events to Kafka.

```
Properties props = new Properties();
props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, brokerList);
props.put(SCHEMA_REGISTRY_URL, restApp.restConnect);
props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
org.apache.kafka.common.serialization.StringSerializer.class);
props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
io.confluent.kafka.serializers.KafkaAvroSerializer.class);
props.put("producer.type", "sync");
props.put("serializer.class","kafka.serializer.StringEncoder");
props.put("request.required.acks", "1");
props.put("ssl.keystore.location","/var/ssl/kafka.client.keystore.jks")
props.put("ssl.keystore.password","test1234")
props.put("ssl.key.password","test1234")
ProducerConfig config = new ProducerConfig(props);
String userSchema = "{\"namespace\": \"example.avro\", \"type\":
\"record\", \"name\": \"User\"," +
                        "\"fields\": [{\"name\": \"name\", \"type\":
\"string\"}]}";
Schema.Parser parser = new Schema.Parser();
Schema schema = parser.parse(userSchema);
GenericRecord avroRecord = new GenericData.Record(schema);
avroRecord.put("name", "testUser");
Producer<String, String> producer = new Producer<String,
String>(config);
ProducerRecord<String, Object> message = new
ProducerRecord<>("topicName", avroRecord );
producer.send(data);
```

```
SparkProducer producer =
HopsUtil.getSparkProducer();
```

https://github.com/hopshadoop/hops-kafka-examples

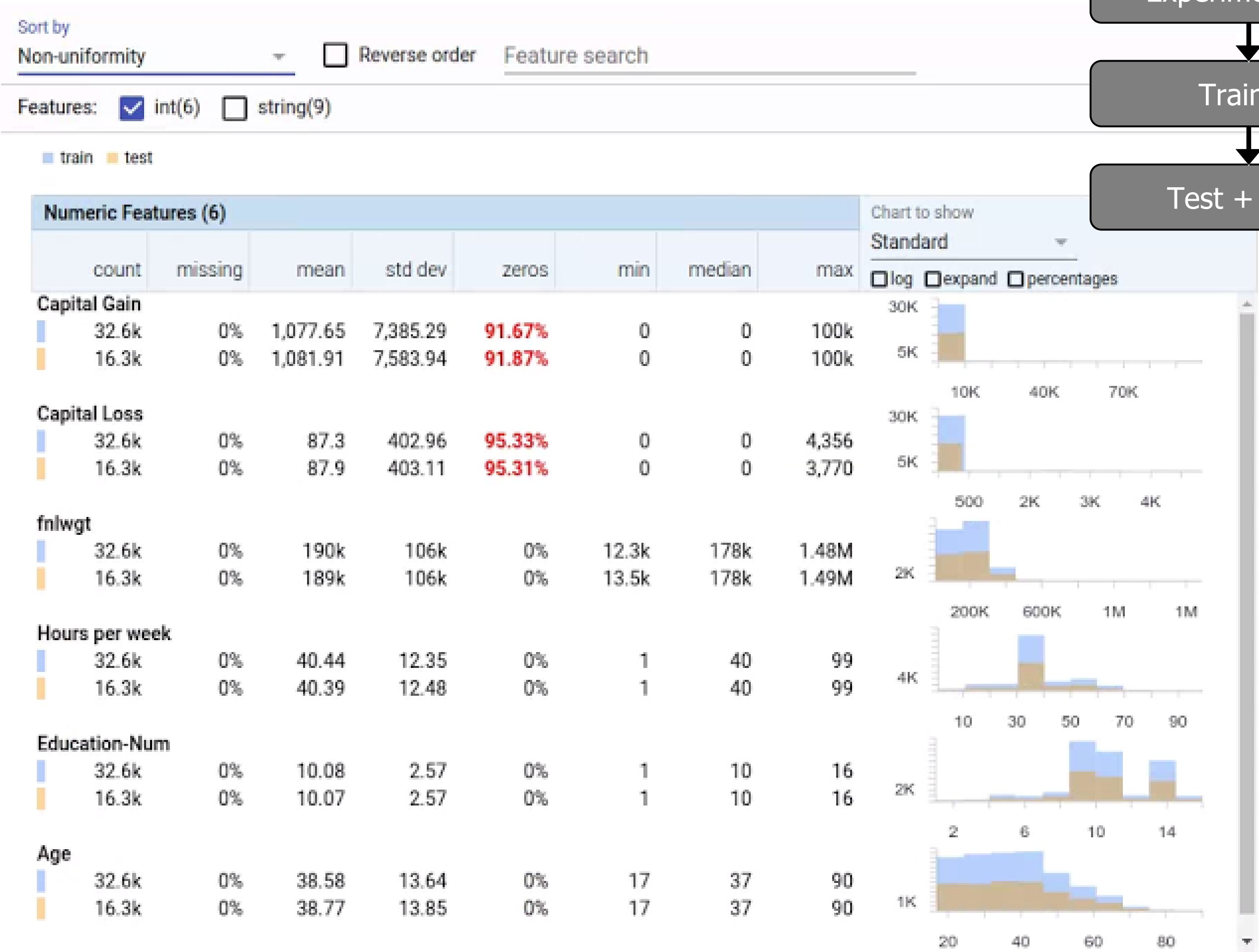
Hive LLAP vs SparkSQL



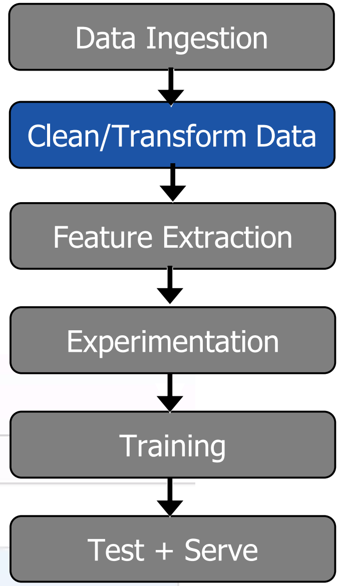
- Hive LLAP
  - ORC format
  - Fast startup (LLAP Daemons)
  - Zeppelin support
  - Integration with BI tools (Tableau, Qlik, etc)
- Spark SQL
  - Parquet Format
  - Slow startup w/ YARN
  - Integrated with SparkML / GraphX
  - DataFrames can be written as TfRecords

Google Facets Overview

- Visualize data distributions
- Min/max/mean/media values for features
- Missing values in columns
- Facets Overview expects test/train datasets as input

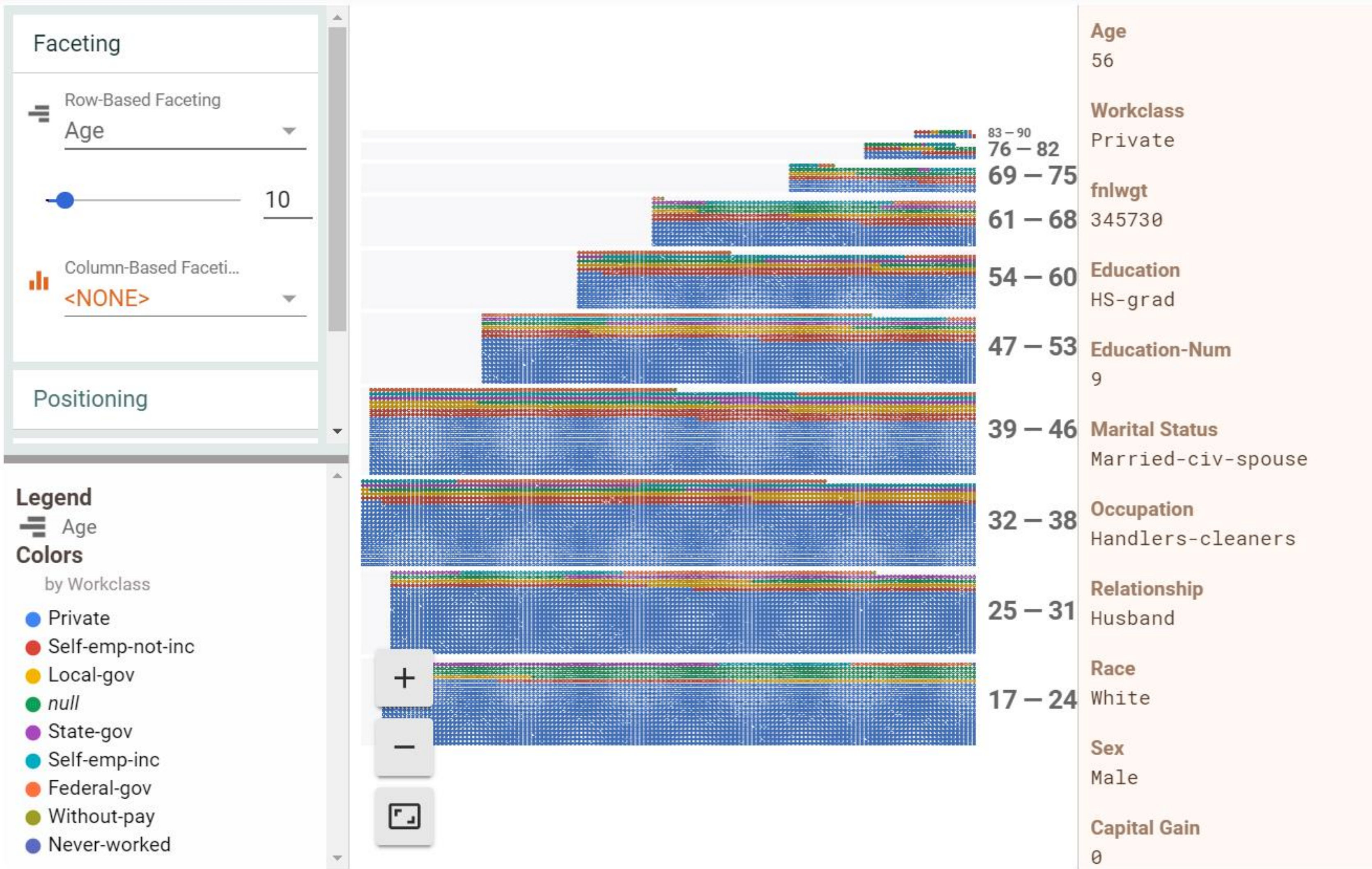


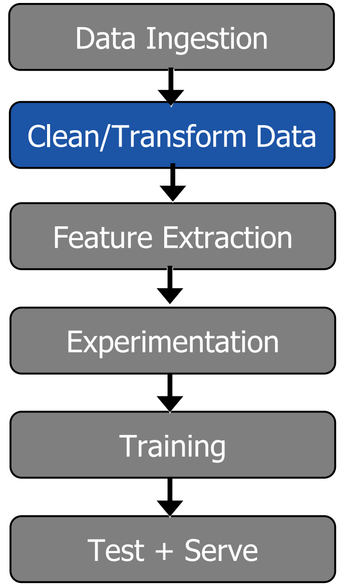
	count	missing	mean	std dev	zeros	min	median	max
Capital Gain								
train	32.6k	0%	1,077.65	7,385.29	91.67%	0	0	100k
test	16.3k	0%	1,081.91	7,583.94	91.87%	0	0	100k
Capital Loss								
train	32.6k	0%	87.3	402.96	95.33%	0	0	4,356
test	16.3k	0%	87.9	403.11	95.31%	0	0	3,770
fnlwgt								
train	32.6k	0%	190k	106k	0%	12.3k	178k	1.48M
test	16.3k	0%	189k	106k	0%	13.5k	178k	1.49M
Hours per week								
train	32.6k	0%	40.44	12.35	0%	1	40	99
test	16.3k	0%	40.39	12.48	0%	1	40	99
Education-Num								
train	32.6k	0%	10.08	2.57	0%	1	10	16
test	16.3k	0%	10.07	2.57	0%	1	10	16
Age								
train	32.6k	0%	38.58	13.64	0%	17	37	90
test	16.3k	0%	38.77	13.85	0%	17	37	90



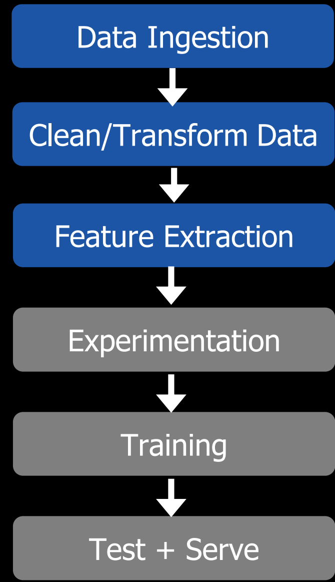
Google Facets Dive

- Visualize the relationship between the data points across the different features of a dataset
- Facets Dive expects input dataset as json



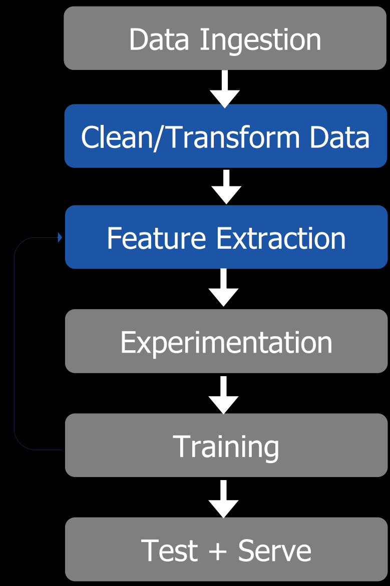


Data Ingestion (HopsFS) and Google Facets



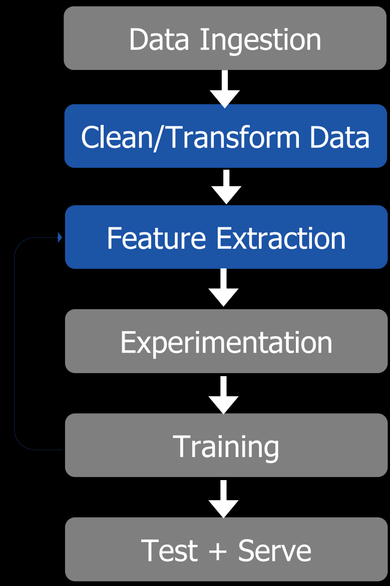

```
import hops.hdfs as hdfs

features = ["Age", "Occupation", "Sex", …, "Country"]
h = hdfs.get_fs()
with h.open_file(hdfs.project_path() +
 "/TestJob/data/census/adult.data", "r") as trainFile:
    train_data =pd.read_csv(trainFile, names=features,
        sep=r'\s*,\s*', engine='python', na_values="?")
with h.open_file(hdfs.project_path() +
"/TestJob/data/census/adult.test", "r") as testFile:
    test_data =pd.read_csv(testFile, names=features, sep=r'\s*,\s*',
                   engine='python', skiprows=[0], na_values="?")

from hops import facets
facets.overview(train_data, test_data)
facets.dive(test_data.to_json(orient='records'))
```

Small Data Preparation with tf.data API



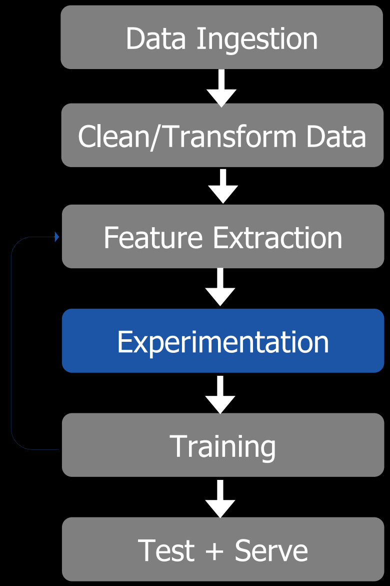

```
def input_fn(batch_sz):
  files = tf.data.Dataset.list_files(IMAGES_DIR)

  def tfrecord_dataset(filename):
   return tf.data.TFRecordDataset(filename,
       num_parallel_reads=32, buffer_size=8*1024*1024)

  dataset = files.apply(tf.data.parallel_interleave
                  (tfrecord_dataset, cycle_length=32, sloppy=True)
  dataset = dataset.apply(tf.data.map_and_batch(parser_fn, batch_sz,
                        num_parallel_batches=4))
  dataset = dataset.prefetch(4)
  return dataset
```

Big Data Preparation with PySpark




```
from mmlspark import ImageTransformer

images = spark.readImages(IMAGE_PATH, recursive = True,
                 sampleRatio = 0.1).cache()

tr = (ImageTransformer().setOutputCol("transformed")
     .resize(height = 200, width = 200)
     .crop(0, 0, height = 180, width = 180) )
smallImages = tr.transform(images).select("transformed")
```

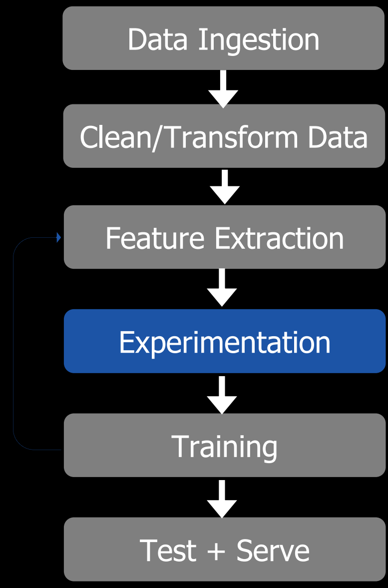
Hyperparam Opt. with Tf/Spark on Hops




```
from hops import experiment

def model_fn(learning_rate, dropout):
  import tensorflow as tf
  from hops import tensorboard, hdfs, devices

 [TensorFlow Code here]

args_dict = {'learning_rate': [0.001, 0.005, 0.01],
             'dropout': [0.5, 0.6]}
experiment.launch(spark, model_fn, args_dict)
```

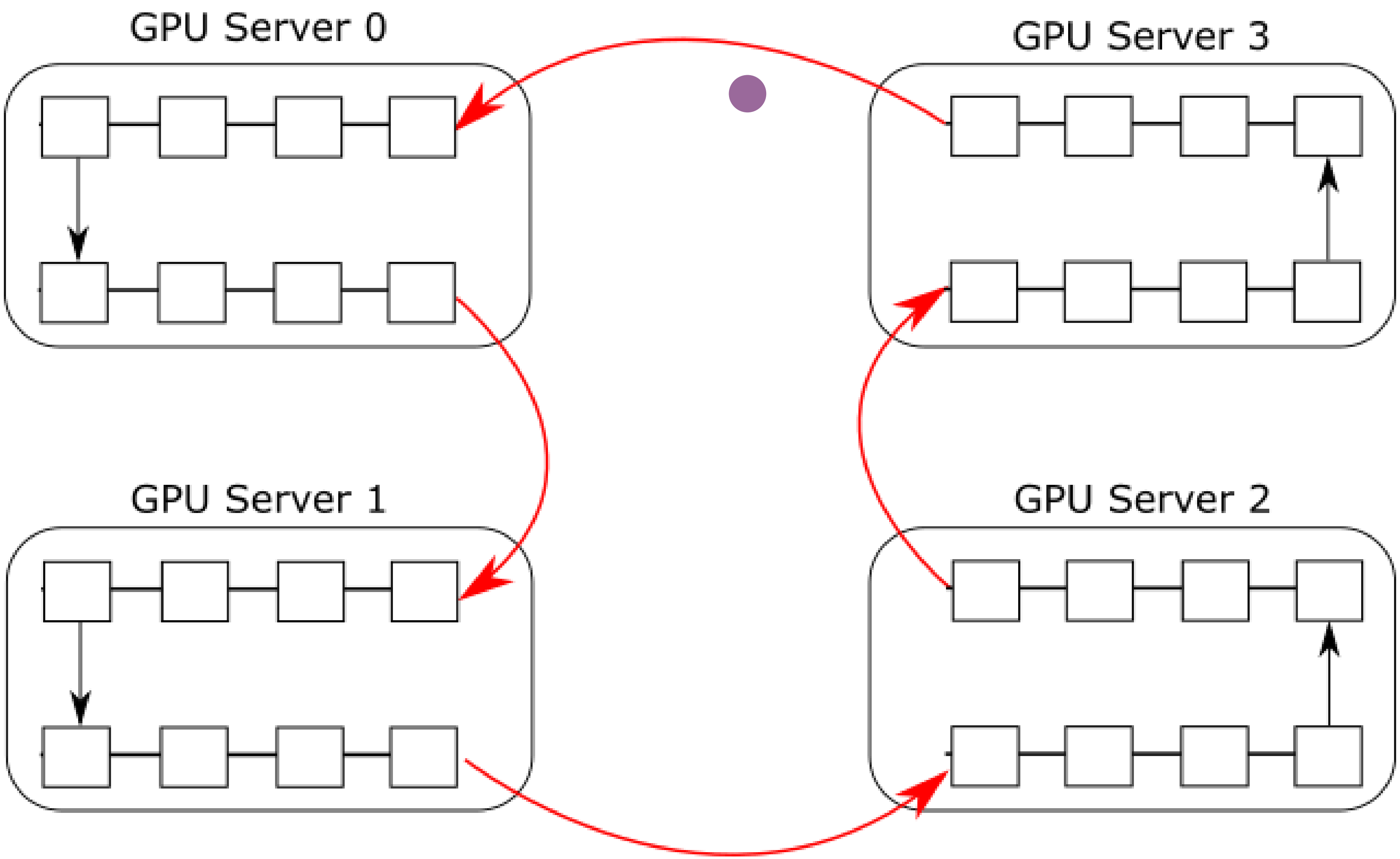
Launch TF jobs in Spark Executors

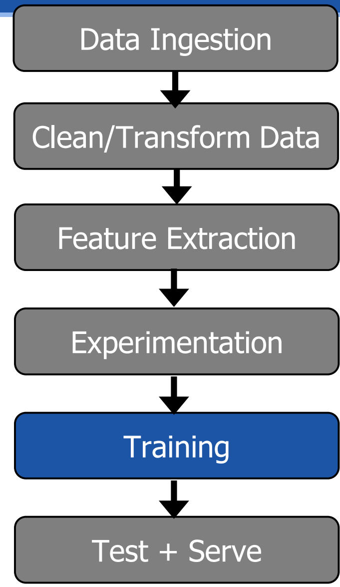
Model Architecture Search Tf/Spark on Hops




```
def model_fn(learning_rate, dropout):
  import tensorflow as tf
  from hops import tensorboard, hdfs, devices

 [TensorFlow Code here with Estimator/Experiment]

from hops import experiment
boundary_dict = {'learning_rate': [0.005, 0.00005],
'dropout': [0.01, 0.99], 'num_layers': [1,3]}

# Differential Evolution searches for good models
tensorboard_hdfs_logdir =
experiment.evolutionary_search(spark, wrapper,
boundary_dict, direction='max', popsize=10, generations=3,
crossover=0.7, mutation=0.5)
```

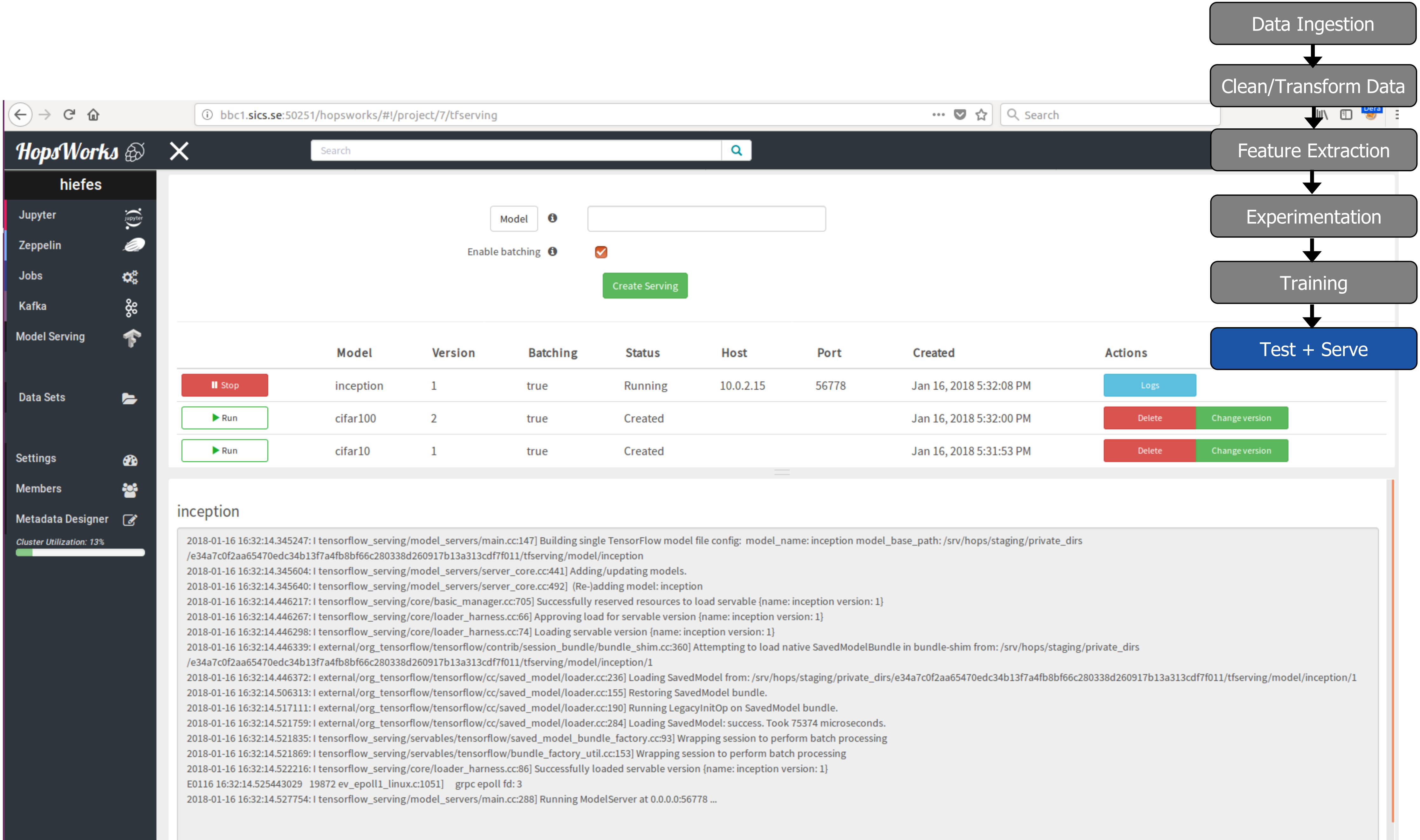
Distributed Training with Ring-AllReduce





Only one slow worker or bus or n/w link is needed to bottleneck training time.

TensorFlow Model Serving



Summary

- Europe’s Only Hadoop Distribution – Hops Hadoop
  - Fully Open-Source
  - Supports larger/faster Hadoop Clusters with GPUs

- Hopsworks is a new Data Platform built on HopsFS with first-class support for Data Science
  - Spark
  - TensorFlow
  - Support services for ML

The Team

Active:

Jim Dowling, Seif Haridi, Tor Björn Minde, Gautier Berthou, Salman Niazi, Mahmoud Ismail, Theofilos Kakantousis, Ermias Gebremeskel, Antonios Kouzoupis, Alex Ormenisan, Fabio Buso, Robin Andersson, August Bonds, Filotas Siskos, Mahmoud Hamed.

www.hops.io

@hopshadoop

Alumni:

Vasileios Giannokostas, Johan Svedlund Nordström,Rizvi Hasan, Paul Mälzer, Bram Leenders, Juan Roca, Misganu Dessalegn, K “Sri” Srijeyanthan, Jude D’Souza, Alberto Lorente, Andre Moré, Ali Gholami, Davis Jaunzems, Stig Viaene, Hooman Peiro, Evangelos Savvidis, Steffen Grohsschmiedt, Qi Qi, Gayana Chandrasekara, Nikolaos Stanogias, Daniel Bali, Ioannis Kerkinos, Peter Buechler, Pushparaj Motamari, Hamid Afzali, Wasif Malik, Lalith Suresh, Mariano Valles, Ying Lieu, Fanti Machmount Al Samisti, Braulio Grana, Adam Alpire, Zahin Azher Rashid, ArunaKumari Yedurupaka, Tobias Johansson , Roberto Bampi.

Hops

Thank You.

Follow us: @hopshadoop

Star us: http://github.com/hopshadoop/hopsworks

Join us: http://www.hops.io