SweDS 2018

The next Frontier: Distributed Deep Learning

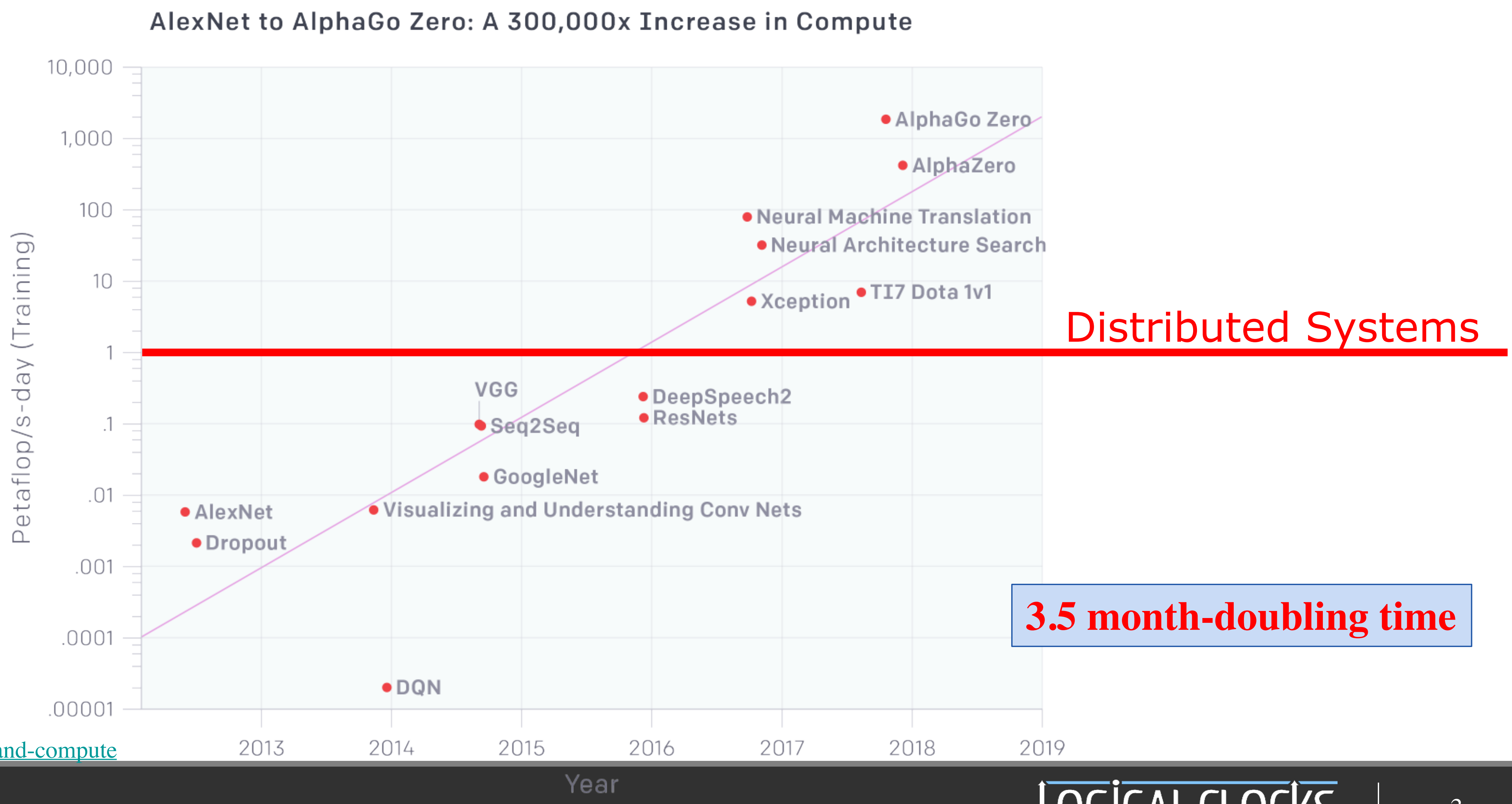
jim_dowling

“Methods that scale with computation are the future of AI”*

- Rich Sutton (Founding Father of Reinforcement Learning)

* https://www.youtube.com/watch?v=EeMCEQa85tw

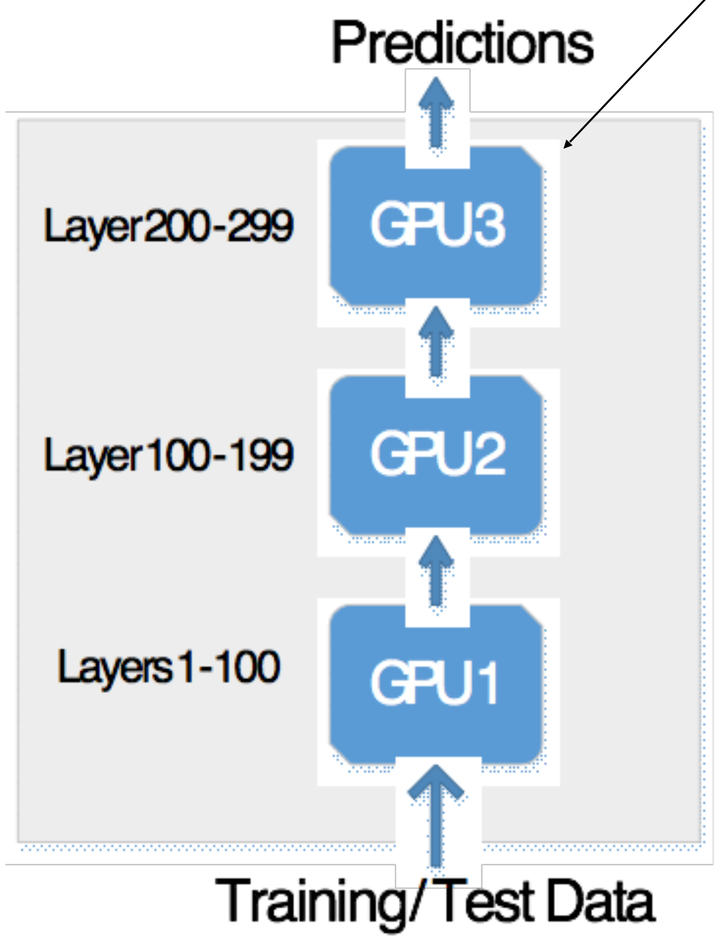
Massive Increase in Compute for AI*



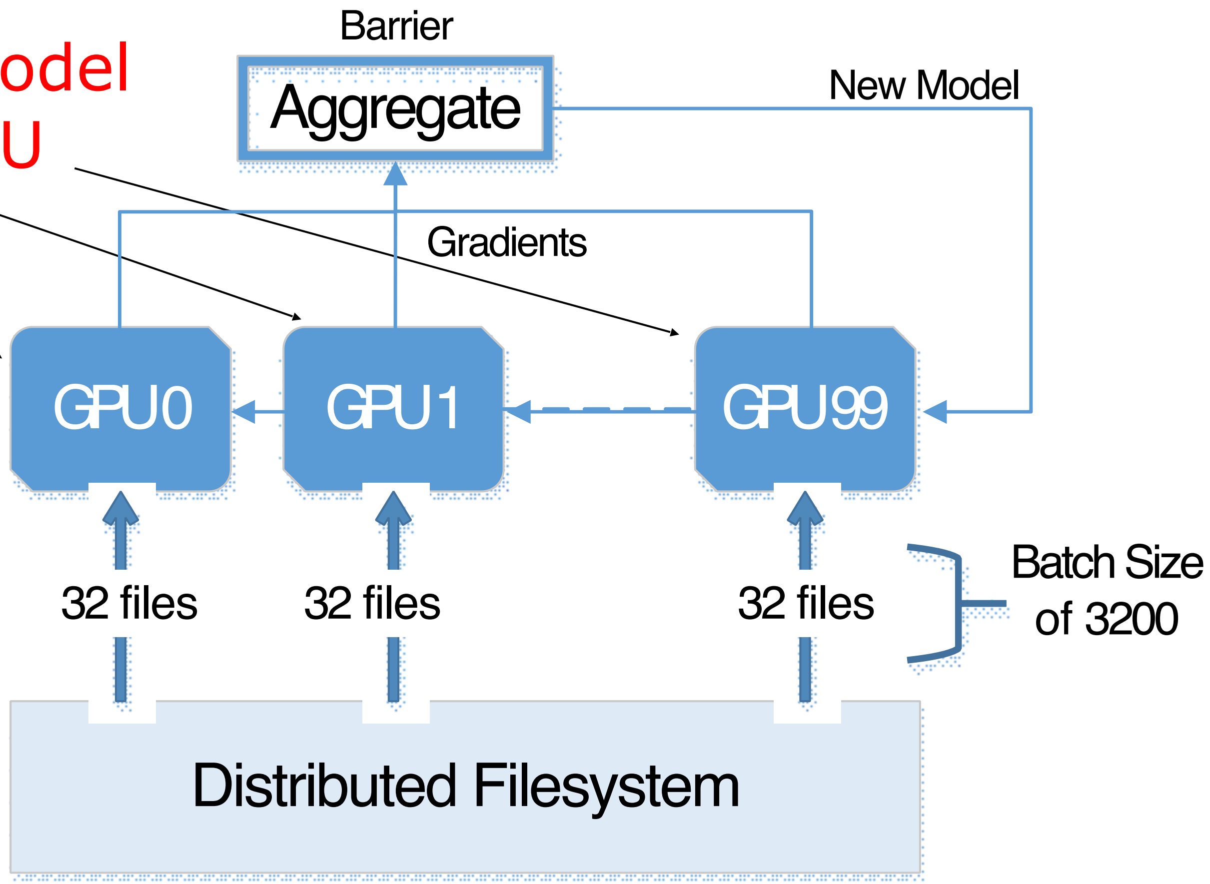
*https://blog.openai.com/ai-and-compute

Model Parallelism Data Parallelism

One Big Model on 3 GPUs



Copy of the Model on each GPU



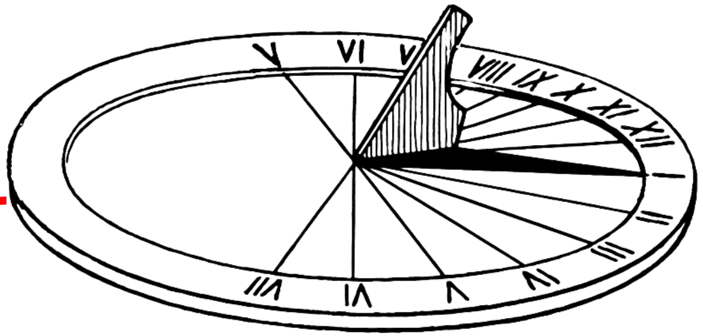
ImageNet

ImageNet Challenge

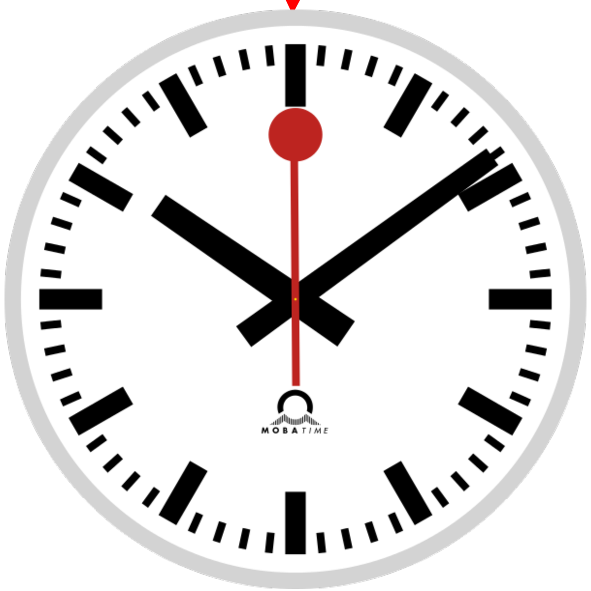
IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

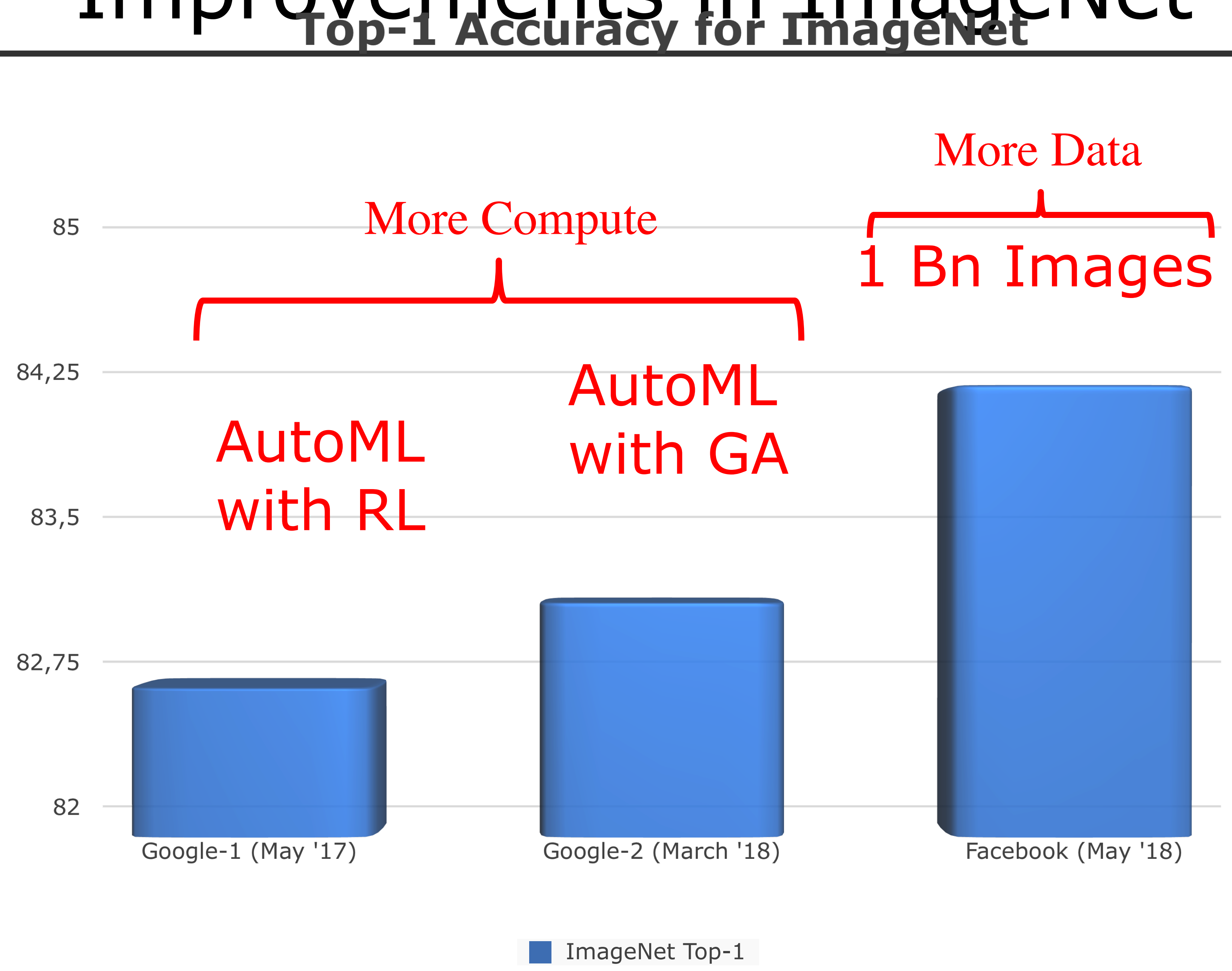
[Image from https://www.slideshare.net/xavigiro/image-classification-on-imagenet-d1l4-2017-upc-deep-learning-for-computer-vision]



Improved Accuracy for ImageNet



Improvements in ImageNet – Accuracy

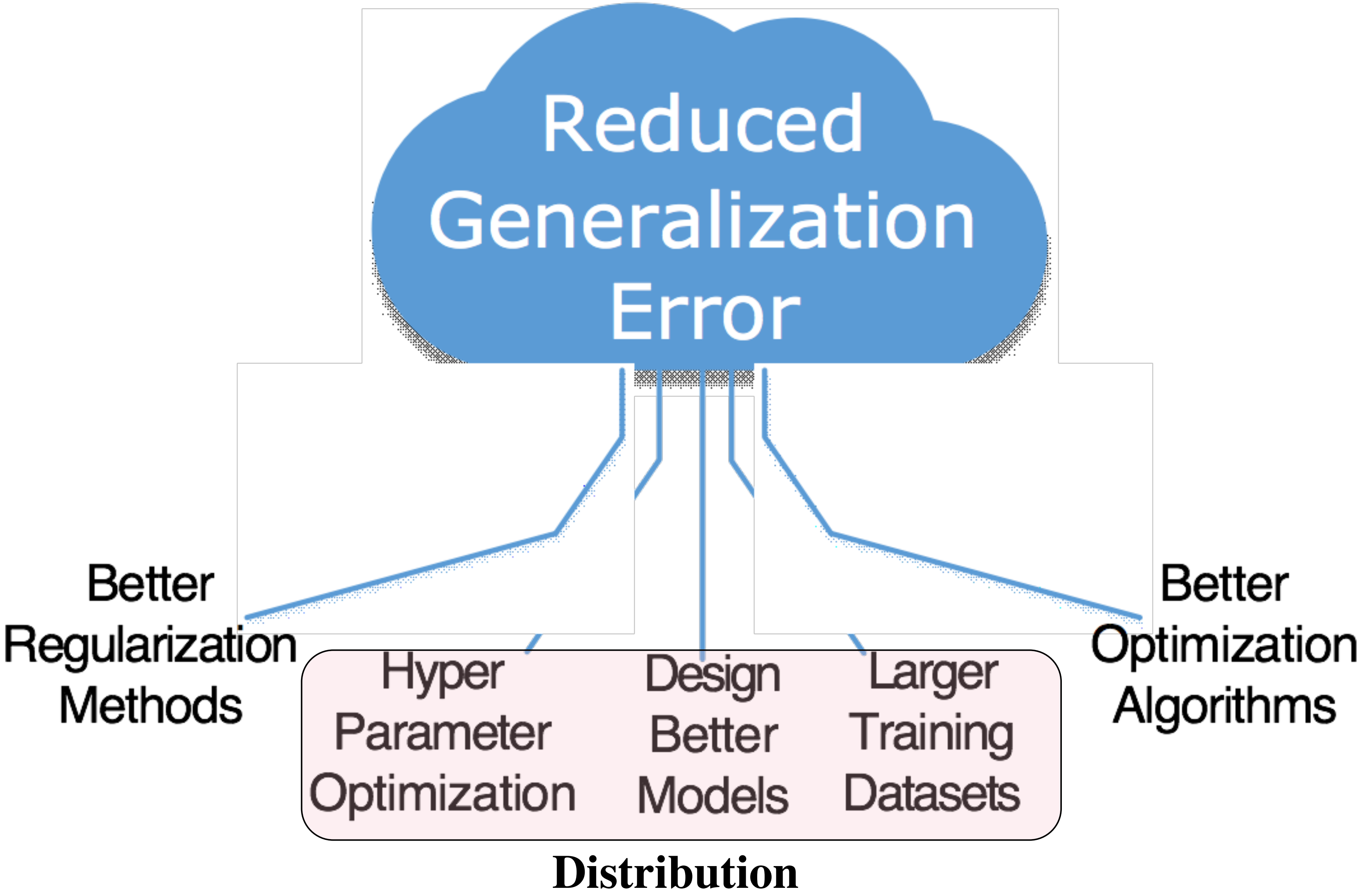


Facebook- https://goo.gl/ERpJyr

Google-1: https://goo.gl/EV7Xv1

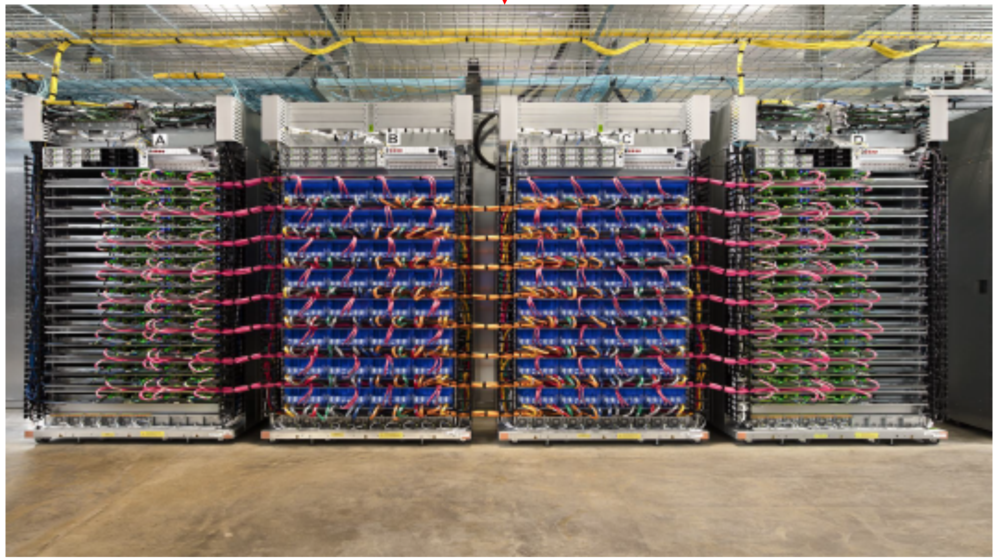
Google-2:https://goo.gl/eidnyQ

Methods for Improving Model Accuracy



Single GPU

Faster Training of ImageNet



Clustered GPUs

[Image from https://www.matroid.com/scaledml/2018/jeff.pdf]

Reduction in Training Time

Table 1 : Training time and top-1 1-crop validation accuracy with ImageNet/ResNet-50

	Batch Size	Processor	DL Library	Time	Accuracy
He et al. [7]	256	Tesla P100 x8	Caffe	29 hours	75.3%
Goyal et al. [1]	8K	Tesla P100 x256	Caffe2	1 hour	76.3%
Smith et al. [4]	8K→16K	full TPU Pod	TensorFlow	30 mins	76.1%
Akiba et al. [5]	32K	Tesla P100 x1024	Chainer	15 mins	74.9%
Jia et al. [6]	64K	Tesla P40 x2048	TensorFlow	6.6 mins	75.8%
This work	**34K→68K**	**Tesla V100 x2176**	**NNL**	**224 secs**	**75.03%**

300X in 3 years

[From https://arxiv.org/abs/1811.05233]

Scaling Efficiency

Network I/O Bound at 56 Gb/s

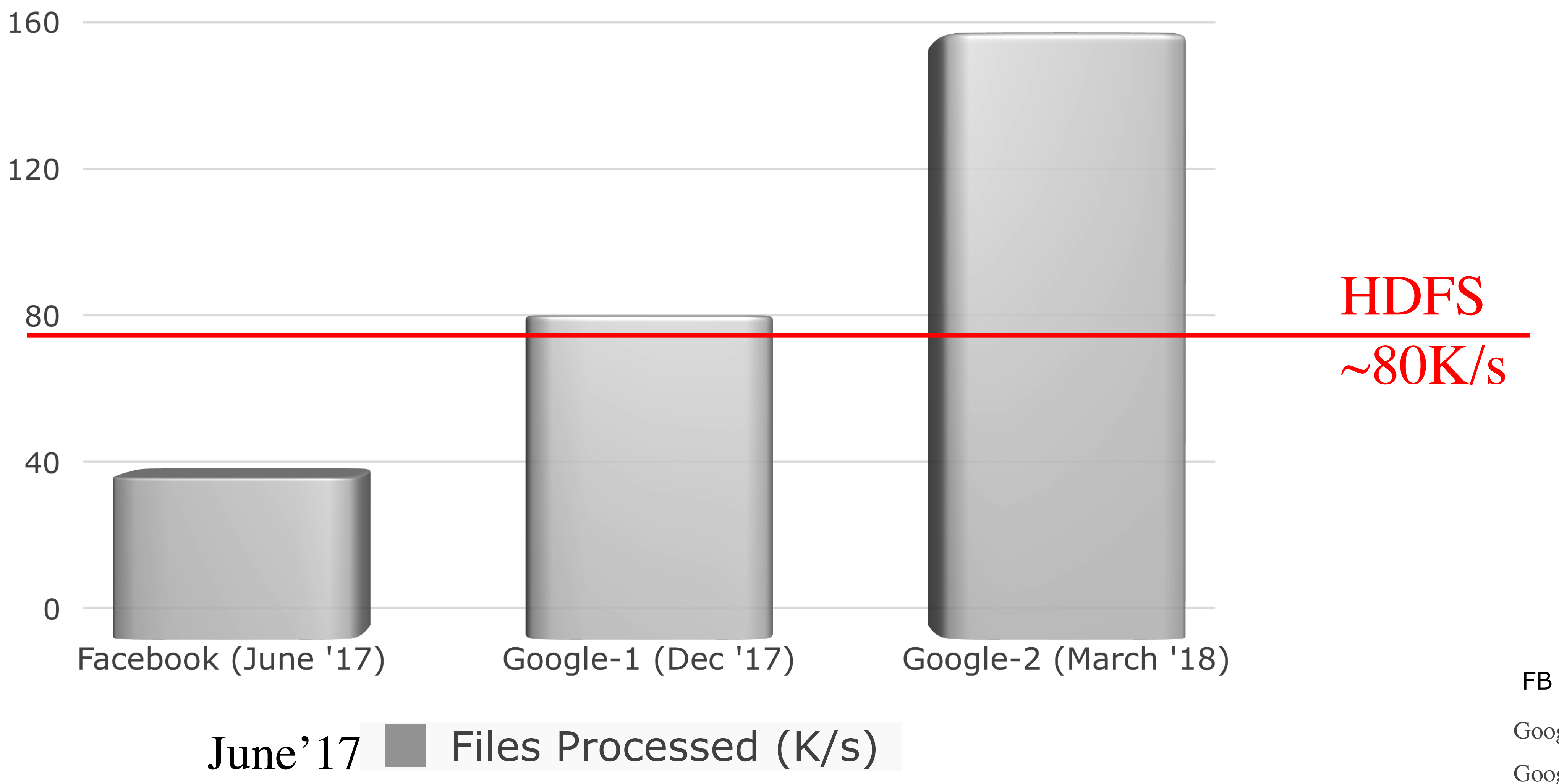
Table 2 : GPU scaling efficiency with ImageNet/ResNet-50 training

	Processor	Interconnect	GPU scaling efficiency
Goyal et al. [1]	Tesla P100 x256	50Gbit Ethernet	~90%
Akiba et al. [5]	Tesla P100 x1024	Infiniband FDR	80%
Jia et al. [6]	Tesla P40 x2048	100Gbit Ethernet	87.9%
This work	**Tesla V100 x1088**	**Infiniband EDR x2**	**91.62%**

Not Network I/O Bound at 200 Gb/s

[From https://arxiv.org/abs/1811.05233]

ImageNet – Files/Sec Processed

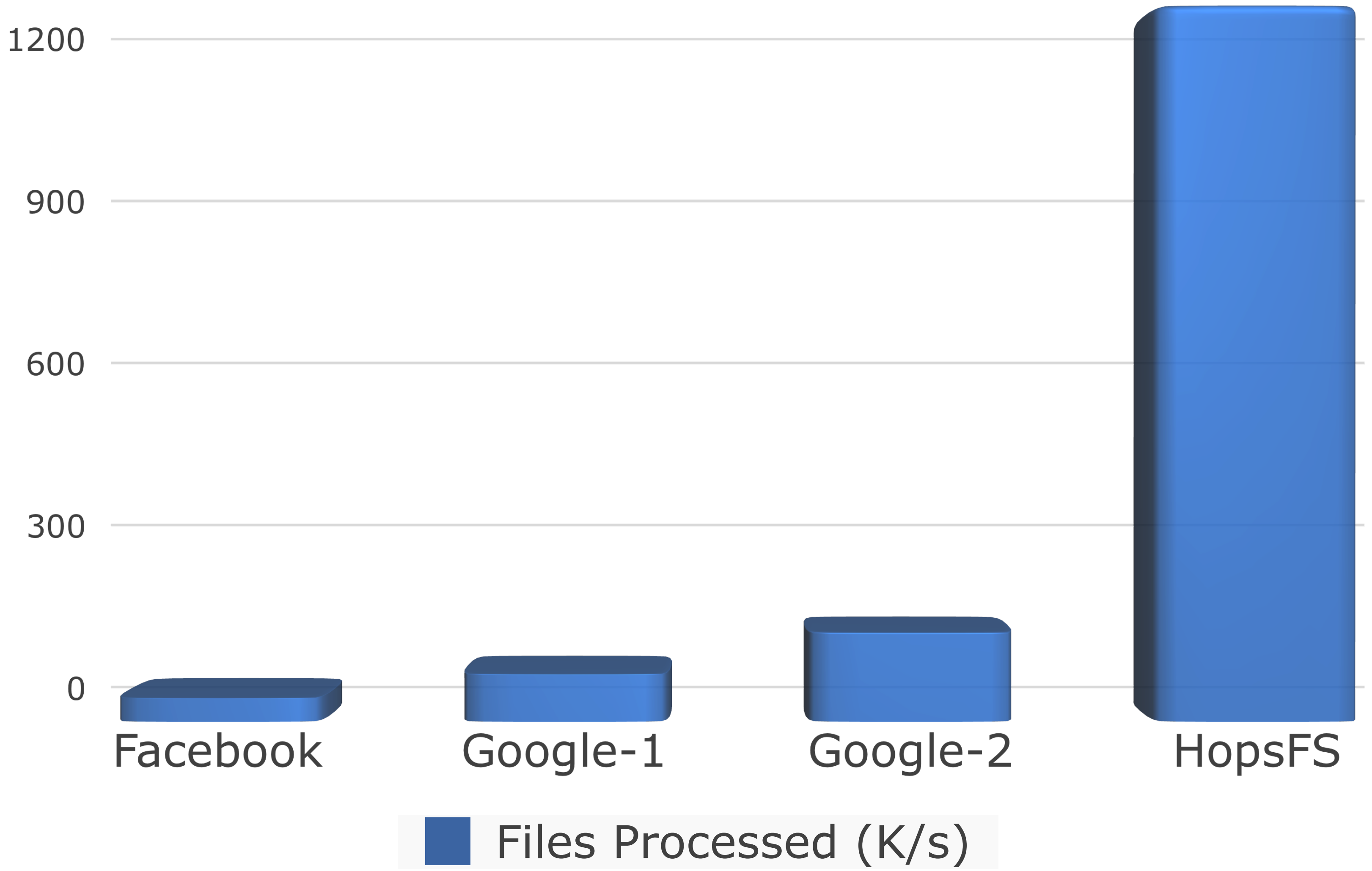


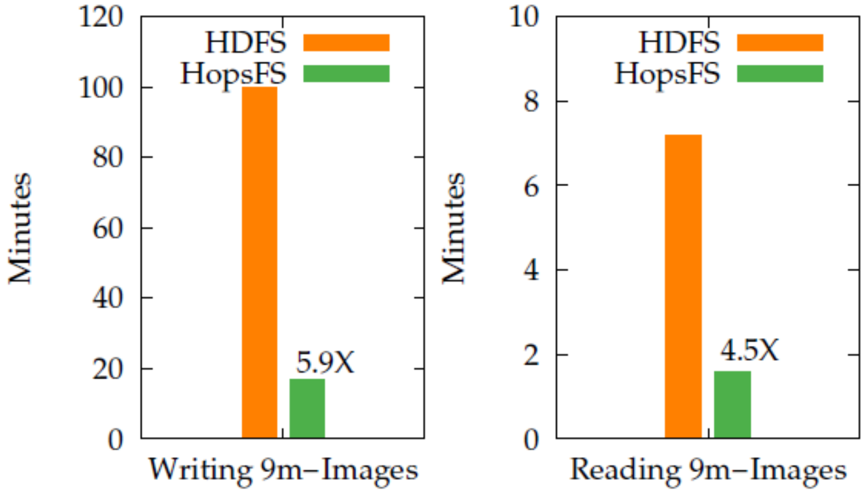
FB - https://goo.gl/ERpJyr

Google-1: https://goo.gl/EV7Xv1

Google-2:https://goo.gl/eidnyQ

Remove Bottleneck and Keep Scaling



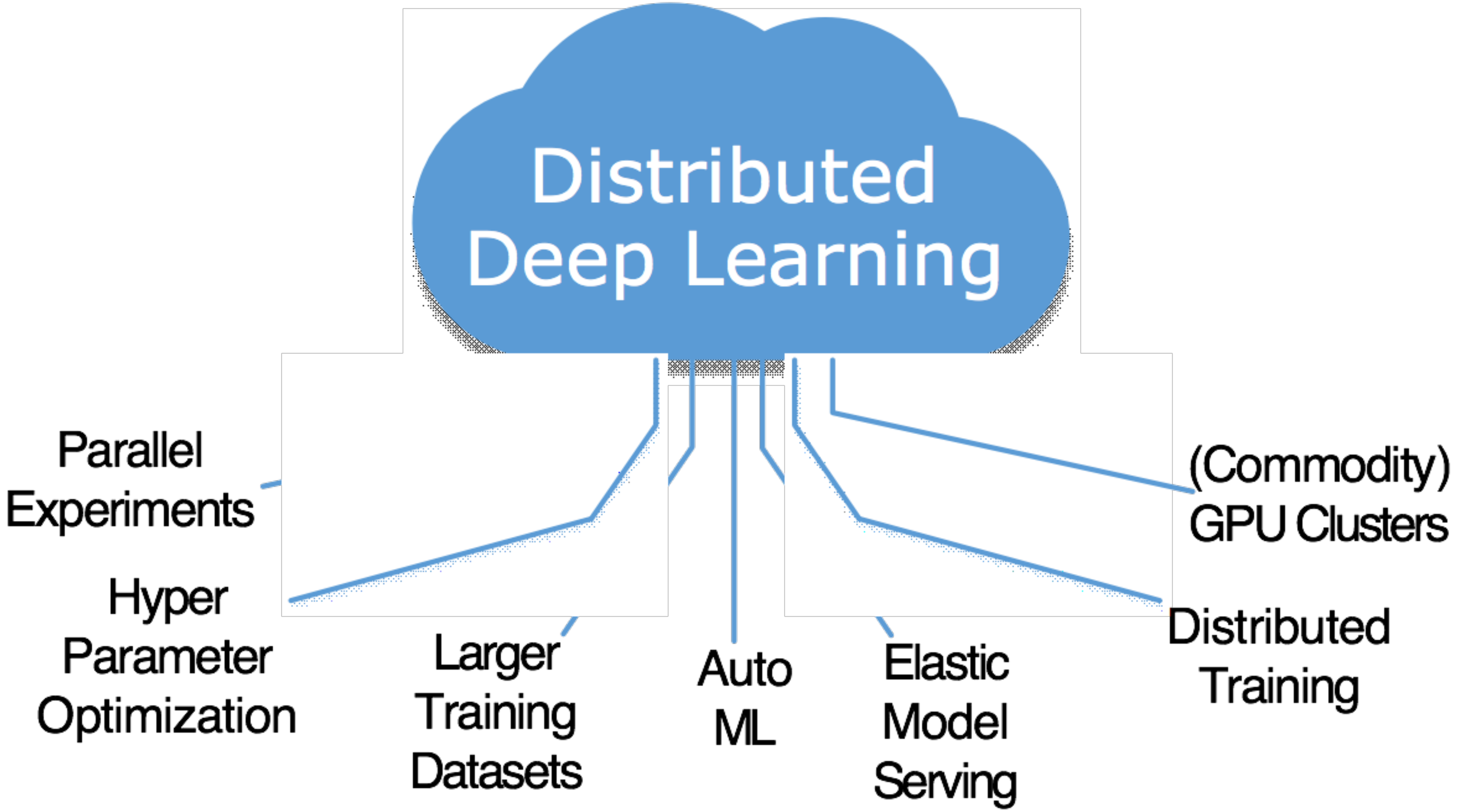


PLUG for Hops

IEEE Hops
Scale Challenge Winner (2017)

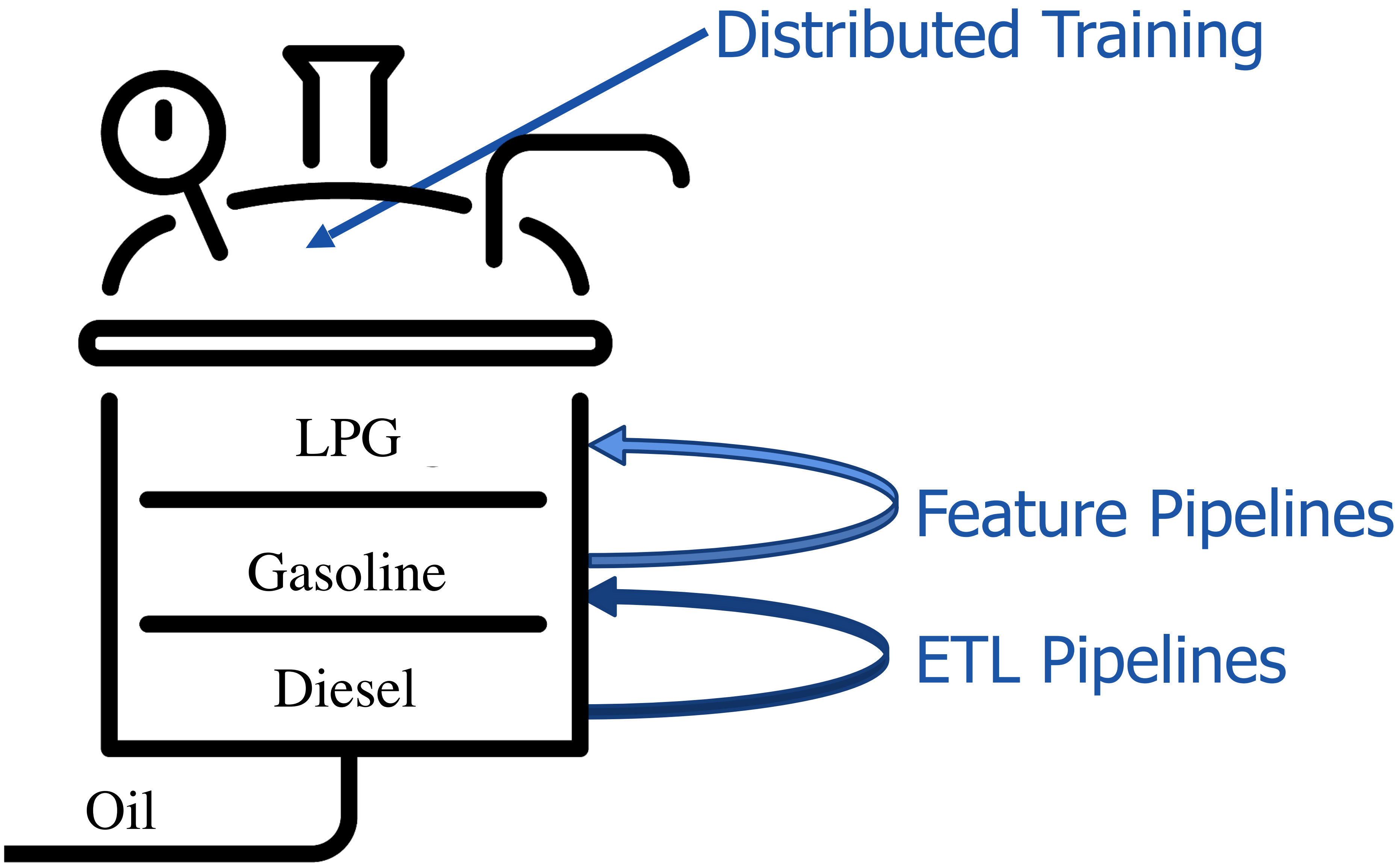
HopsFS - https://goo.gl/yFCsGc

All AI Roads Lead to Distribution

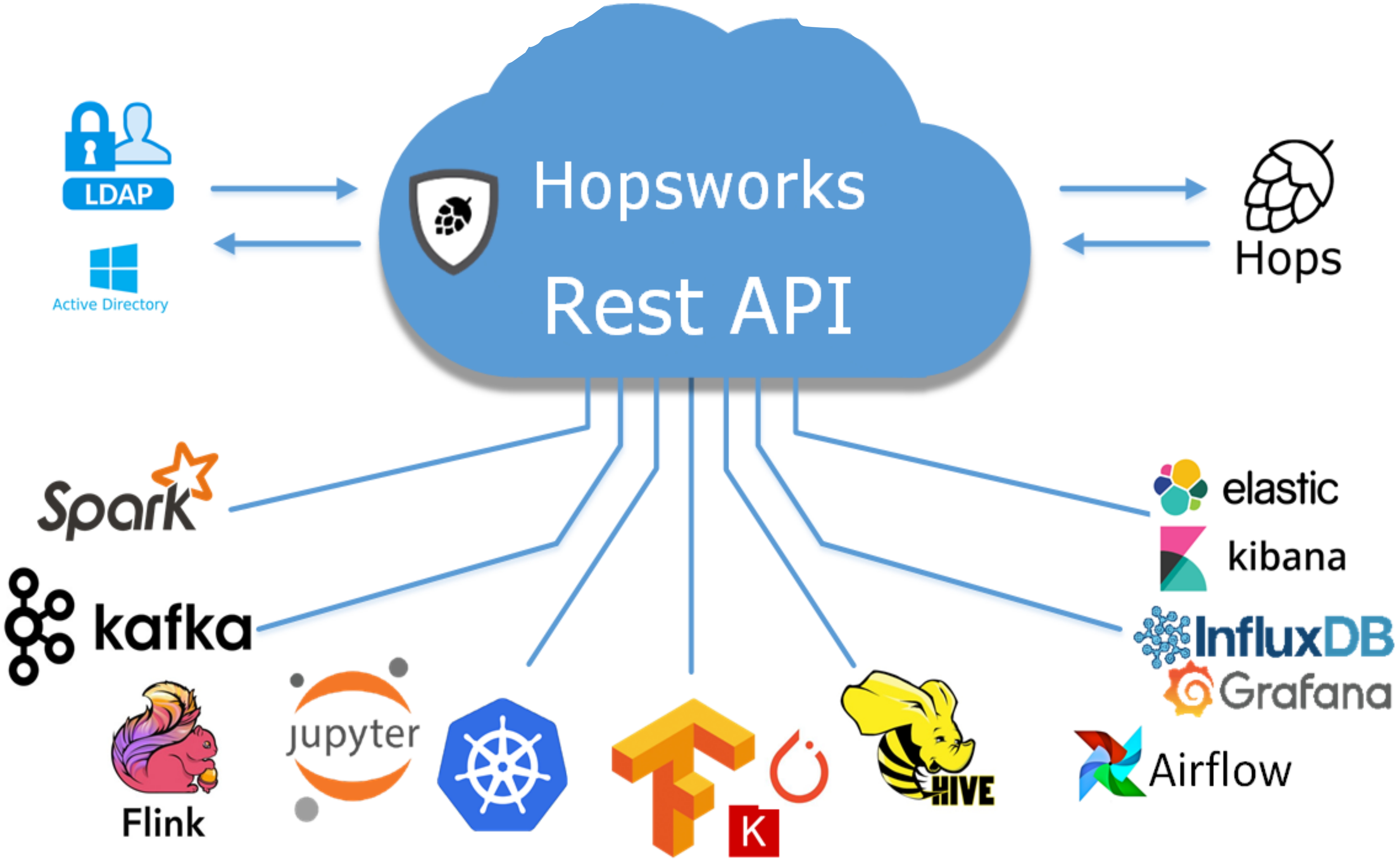


Data may be the new oil,
but refined data is the fuel for AI

Machine Learning is a Data Distillery



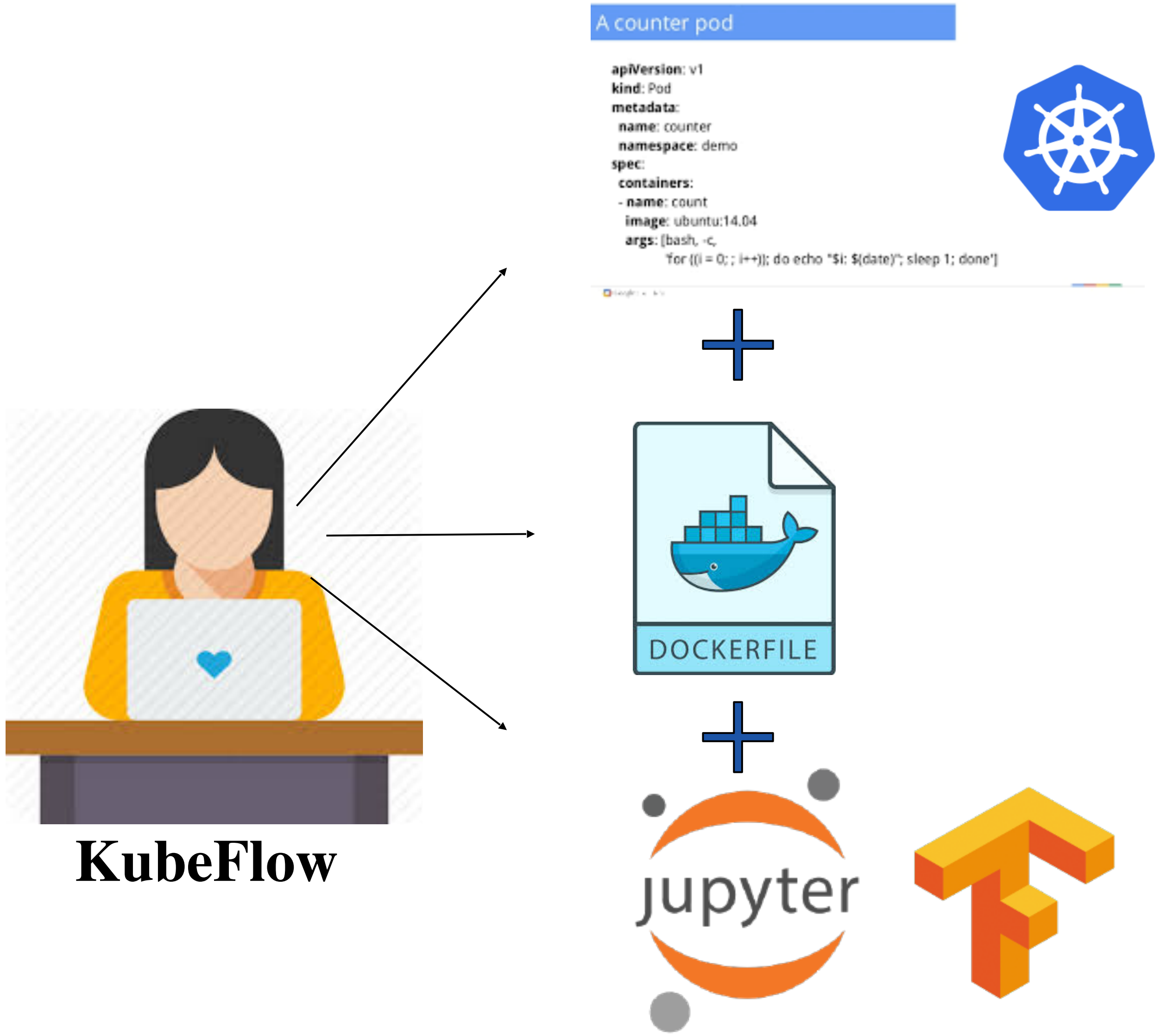
Hopsworks: Data Distiller for ML



Frameworks for Distributed ML

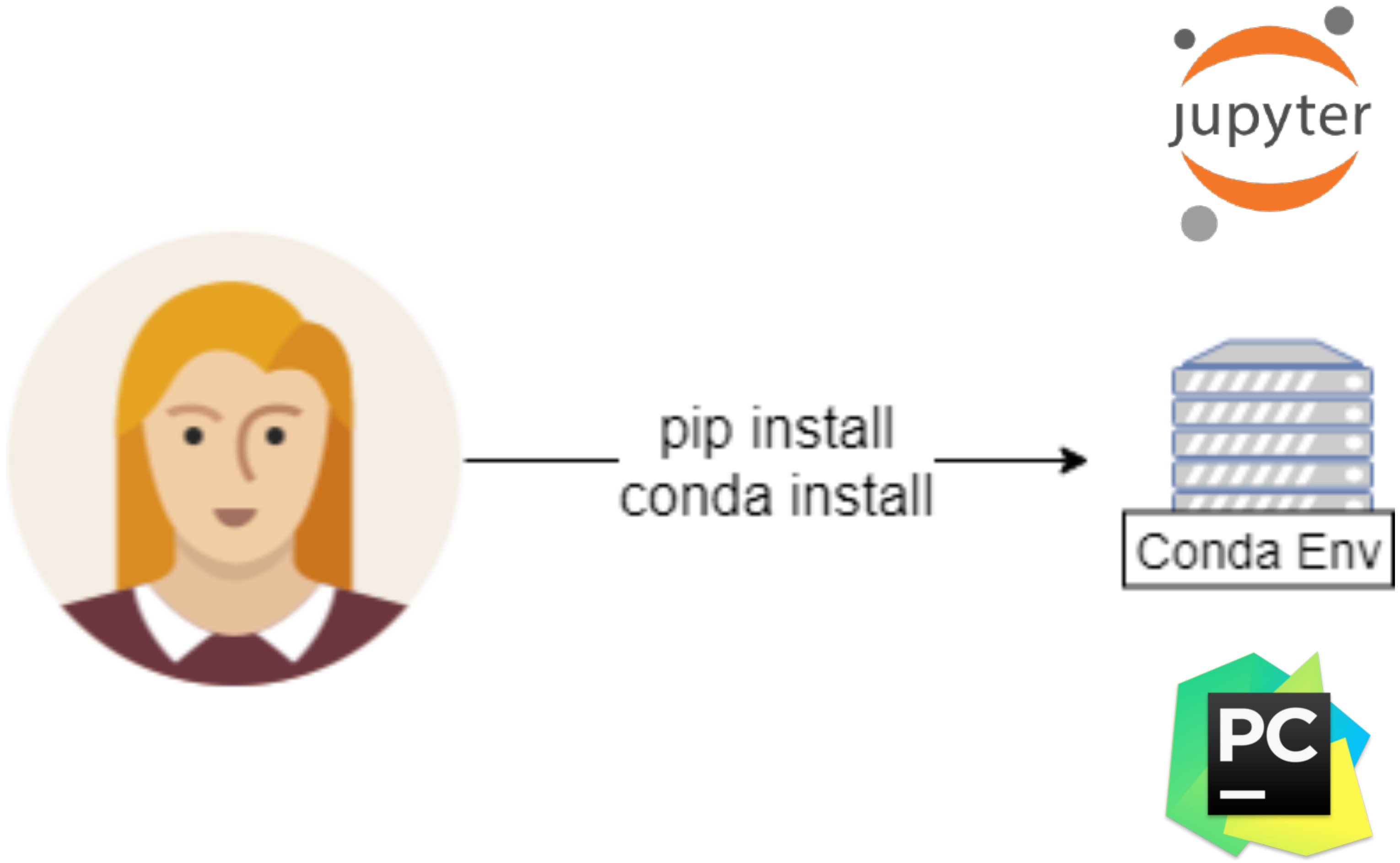
- Managed Cloud Platforms
  - Google Cloud ML
  - Microsoft Batch AI
  - AWS Sagemaker

- On-Premise/Cloud Platforms
  - KubeFlow
  - Hopsworks

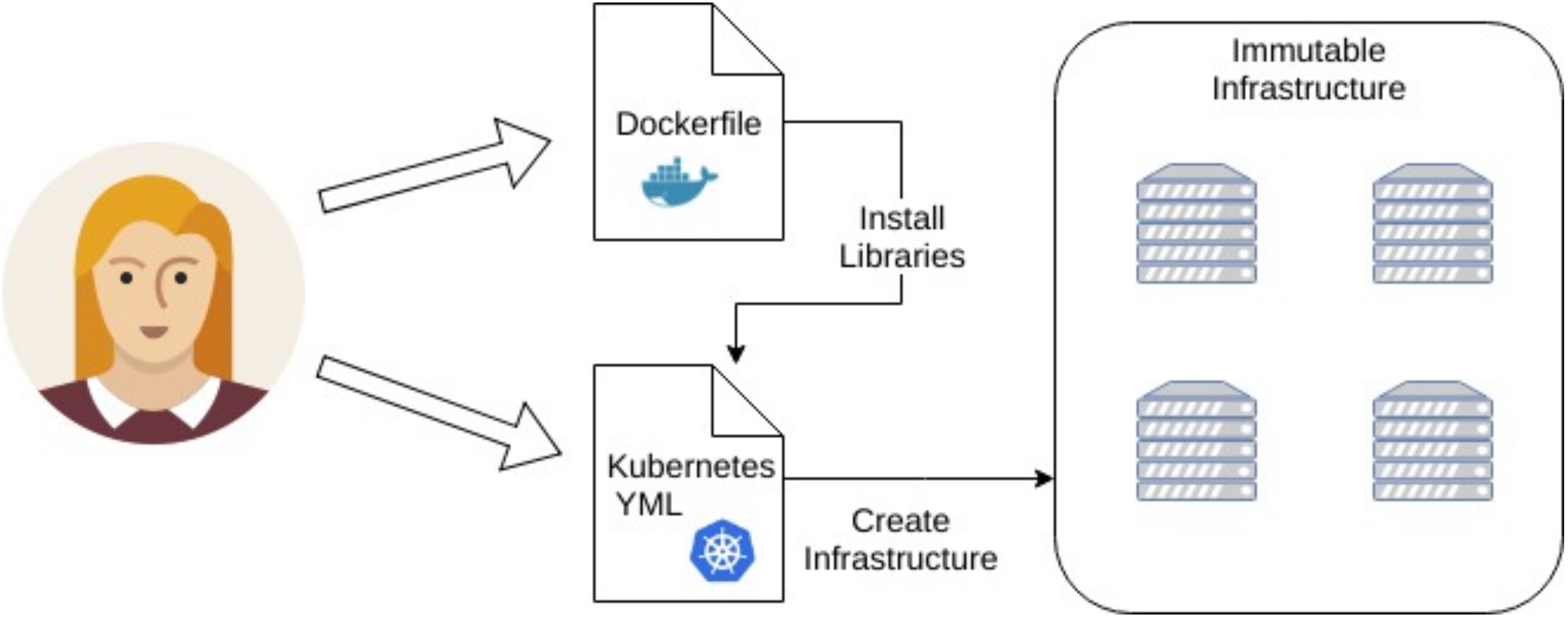


Challenges in Moving to “Distributed” Python

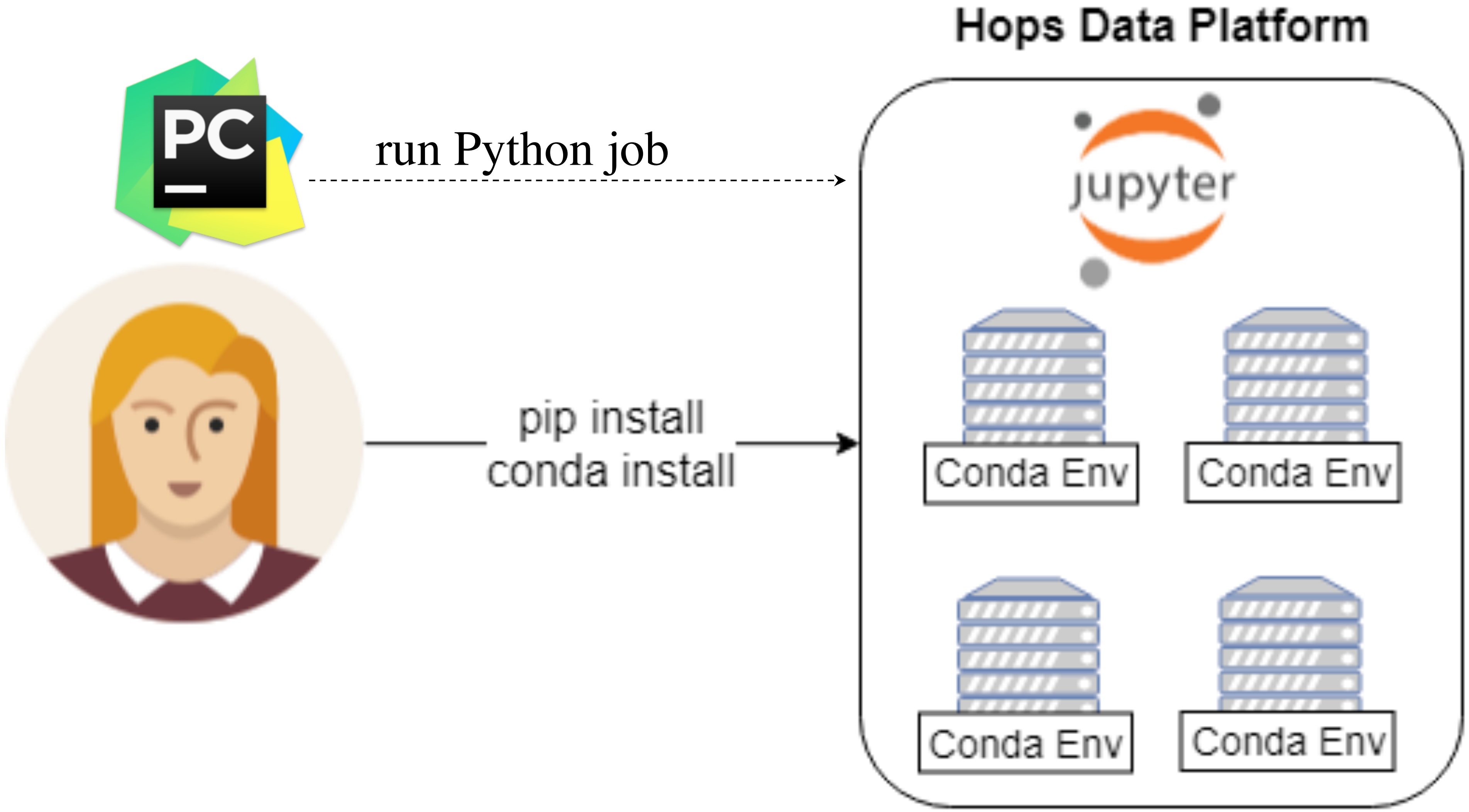
Classic Python Experience



"Cloud-Native" with KubeFlow



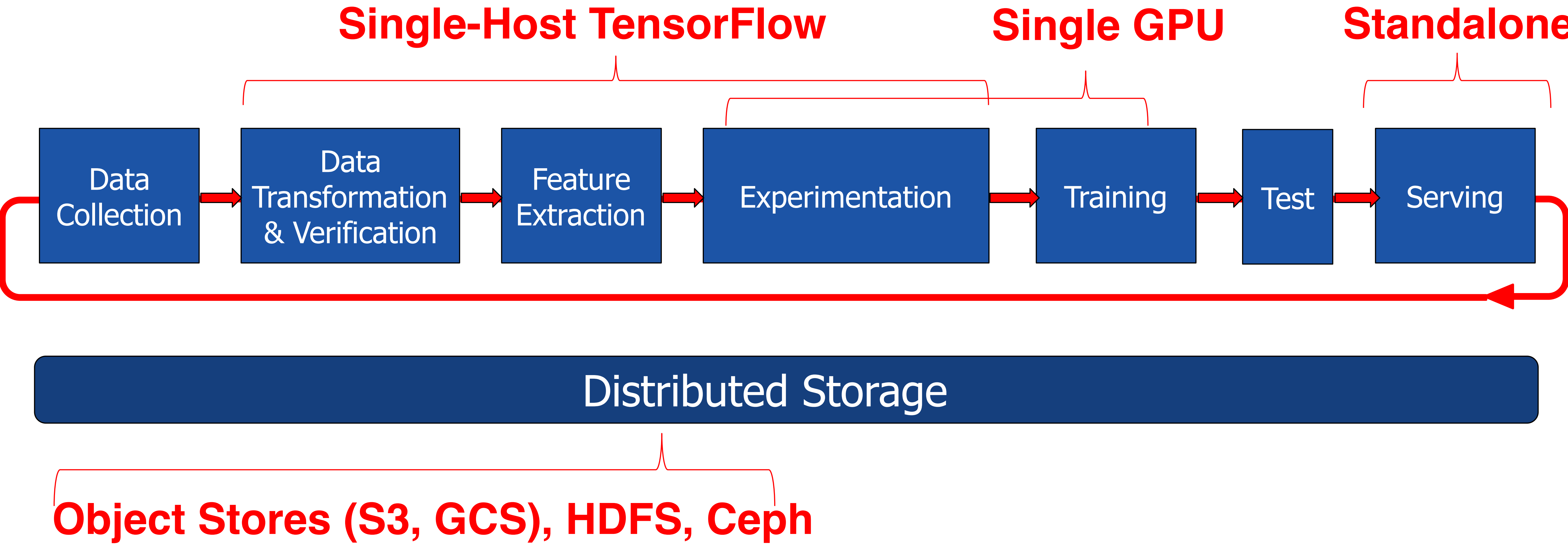
Python on Hopsworks



A Conda Environment per Project

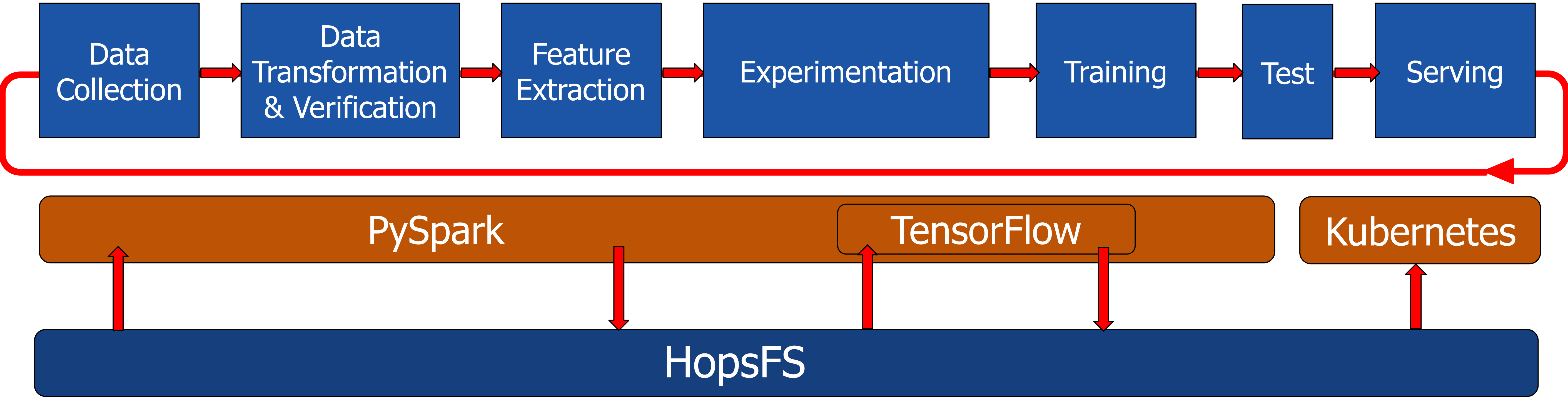
Scalable Machine Learning Pipelines

Scalable ML Pipeline

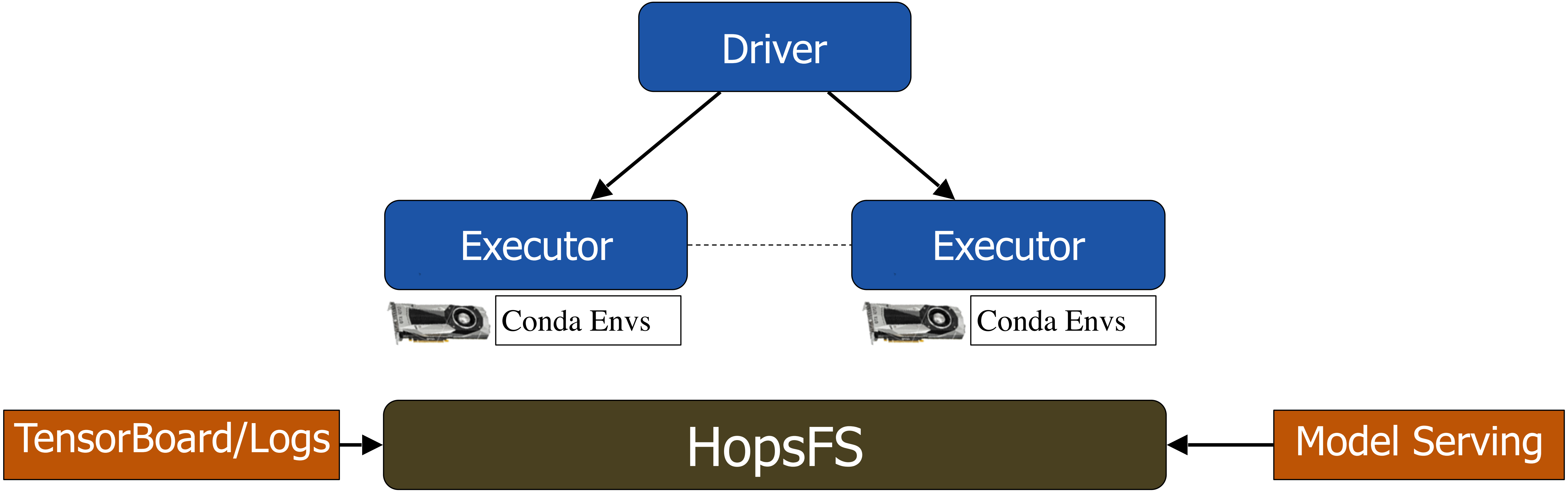


Potential Bottlenecks

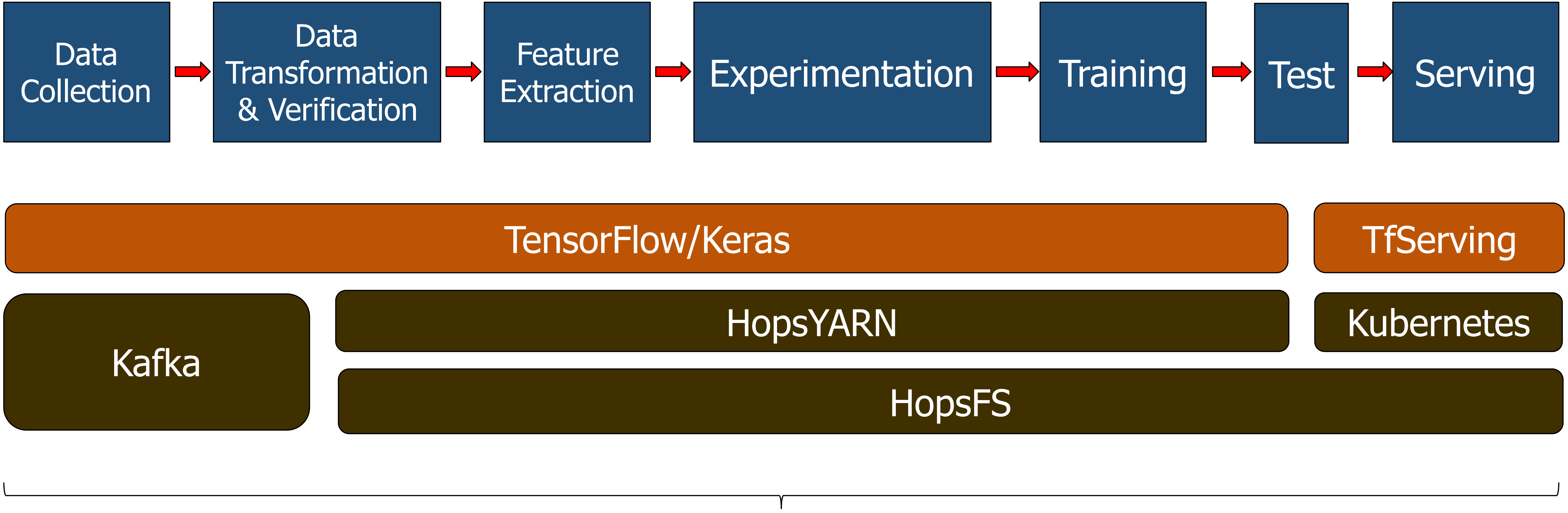
Scalable ML Pipeline in Hopsworks



Spark for Distribution, HopsFS for State

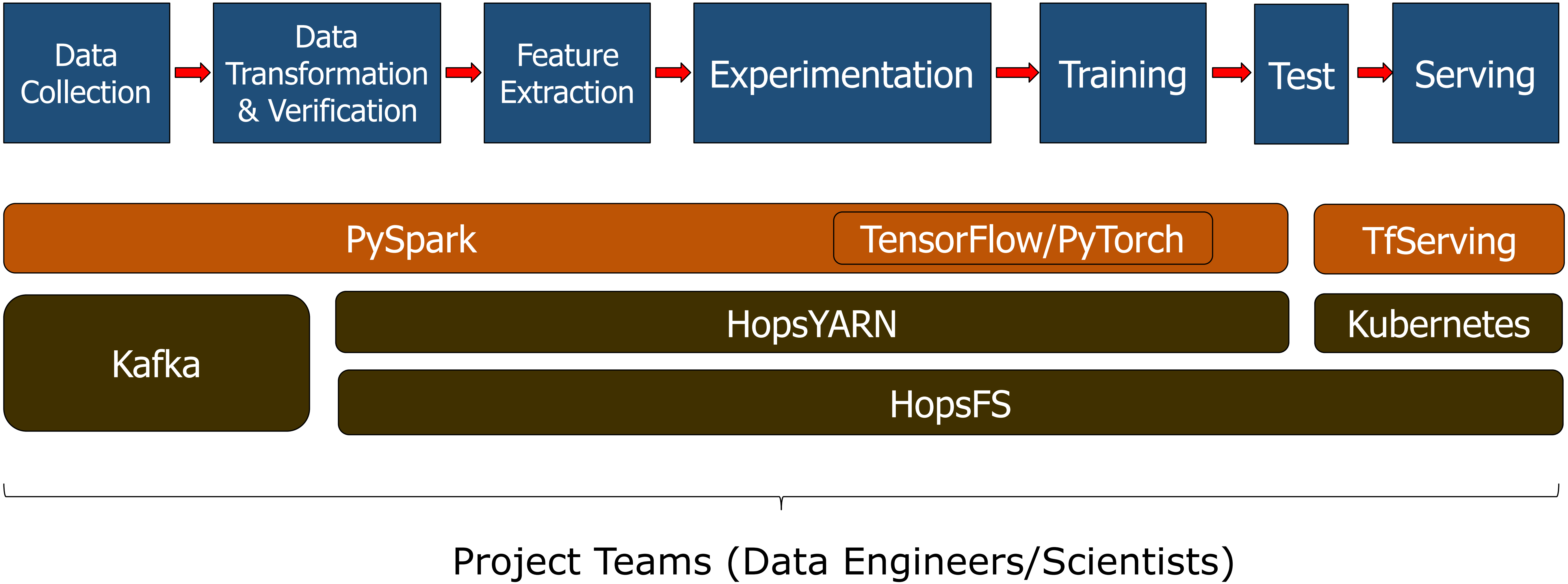


Hops Small Data ML Pipeline



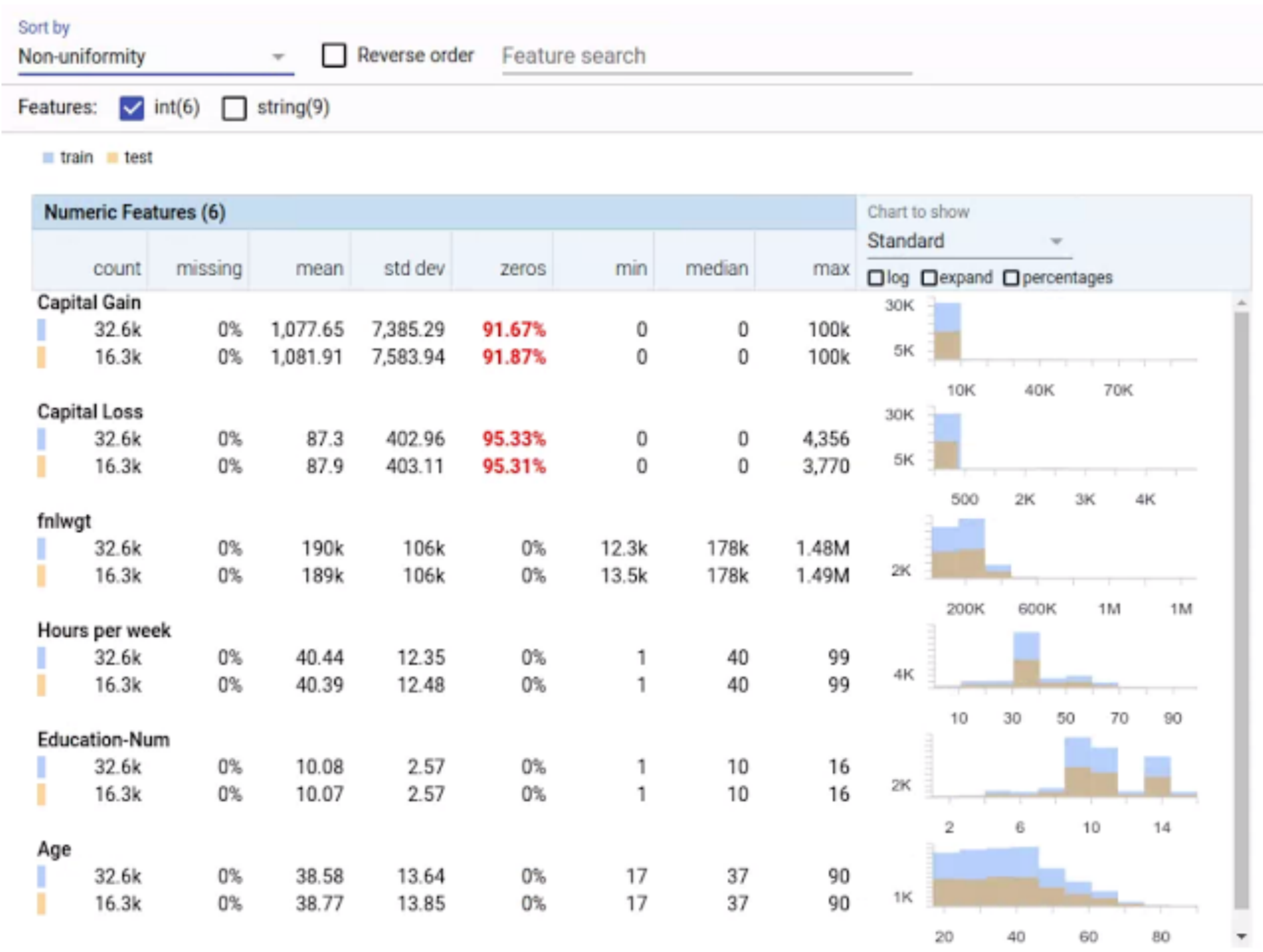
Project Teams (Data Engineers/Scientists)

Hops Big Data ML Pipeline

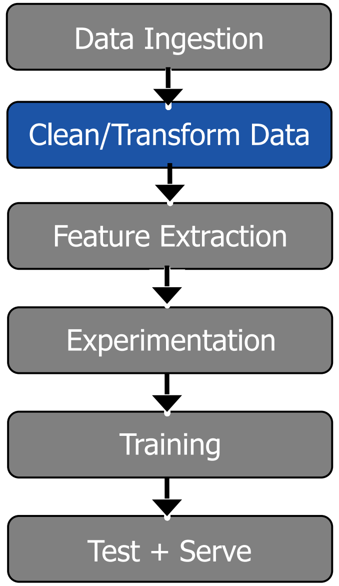


Google TFX + Facets

- Jupyter Plugin
- Visualize data distributions
- Min/max/mean/media values for features
- Missing values in columns
- Test/train datasets



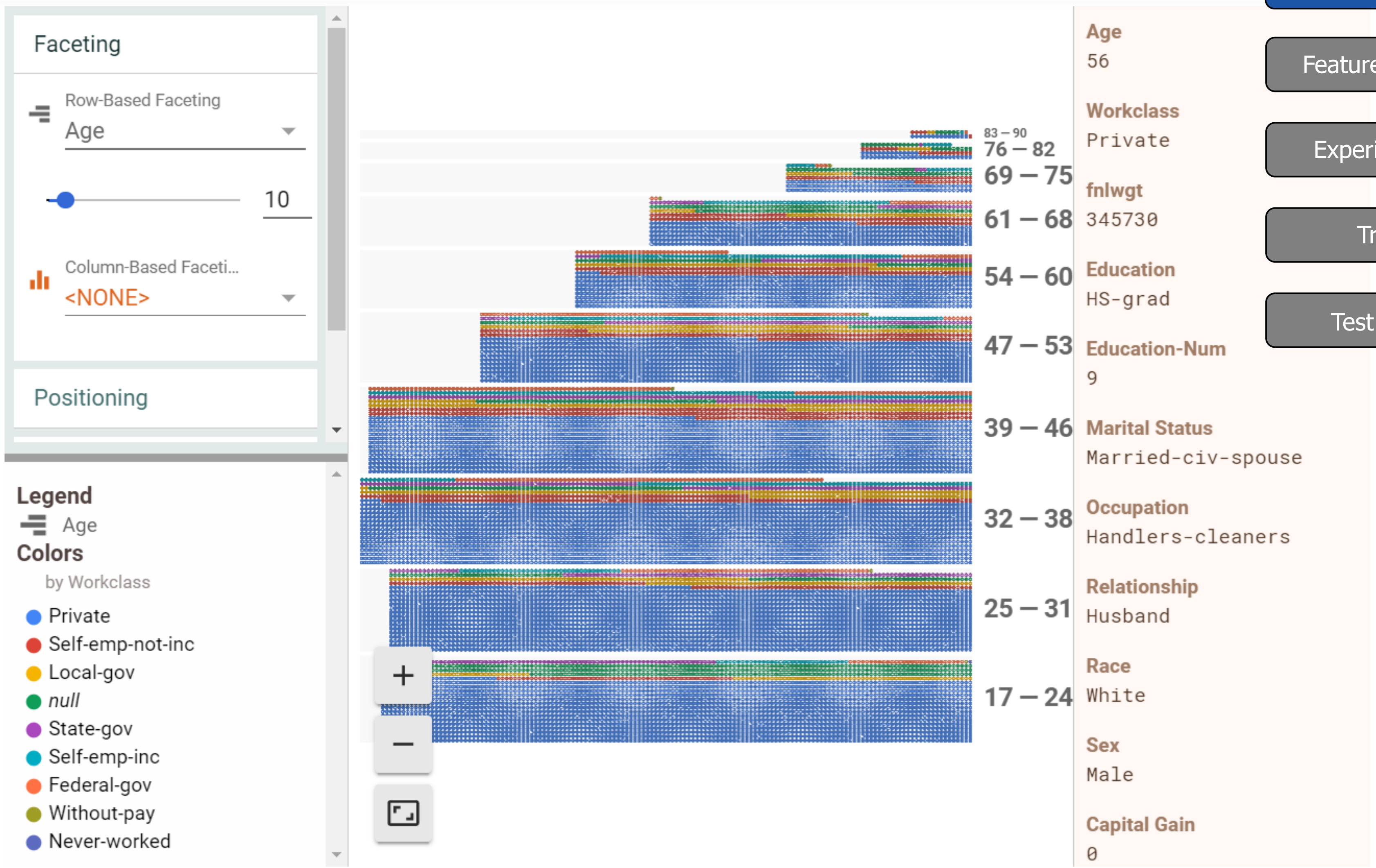
Feature	count	missing	mean	std dev	zeros	min	median	max
Capital Gain (train)	32.6k	0%	1,077.65	7,385.29	91.67%	0	0	100k
Capital Gain (test)	16.3k	0%	1,081.91	7,583.94	91.87%	0	0	100k
Capital Loss (train)	32.6k	0%	87.3	402.96	95.33%	0	0	4,356
Capital Loss (test)	16.3k	0%	87.9	403.11	95.31%	0	0	3,770
fnlwgt (train)	32.6k	0%	190k	106k	0%	12.3k	178k	1.48M
fnlwgt (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
Hours per week (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
Education-Num (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
Age (test)	16.3k	0%	38.77	13.85	0%	17	37	90

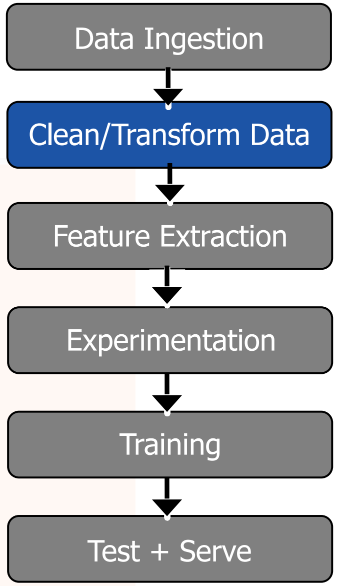


https://medium.com/tensorflow/introducing-tensorflow-data-validation-data-understanding-validation-and-monitoring-at-scale-d38e3952c2f0

Google Facets Dive

- Visualize the relationship between the data points across the different features of a dataset.





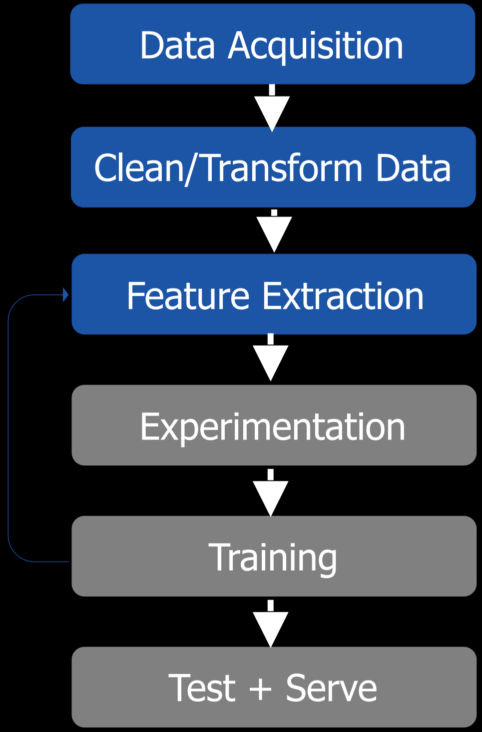
Data Ingestion and Google Facets

- Data Acquisition
- Clean/Transform Data
- Feature Extraction
- Experimentation
- Training
- Test + Serve

```
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="?")
    test_data = …


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

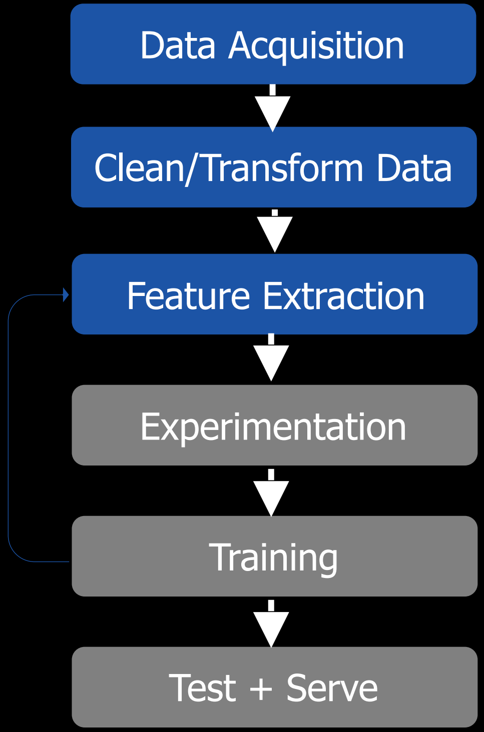
Now we want to pre-process some Images…

Small Data Preparation with tf.data API



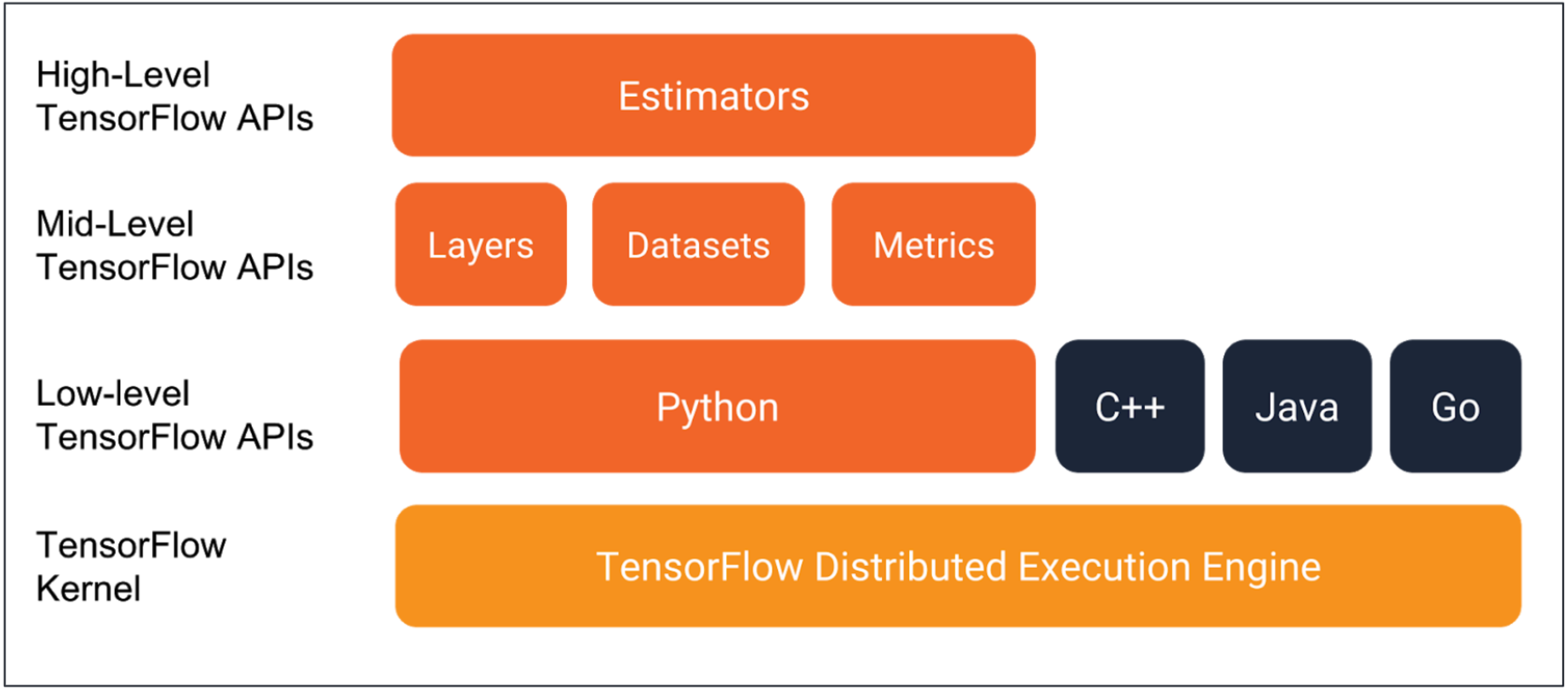

```
def input_fn(batch_size):
  files = tf.data.Dataset.list_files(IMAGES_DIR)

  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_size,
                       num_parallel_batches=4))
  dataset = dataset.prefetch(4)
  return dataset
```

Big Data Preparation with PySpark




```
images = spark.readImages(IMAGE_PATH, recursive = True,
                 numPartitions=10, 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")

# Output .tfrecords using TensorFlowOnSpark utility
dfutil.saveAsTFRecords(smallImages, OUTPUT_DIR)
```

Estimator APIs in TensorFlow



Estimators log to the Distributed Filesystem

```
#over-simplified code – see
#notebook for full example

tf.estimator.RunConfig(
'CollectiveAllReduceStrategy' model_dir
tensorboard_logs
checkpoints
)
experiment.collective_all_reduce(…)
```

<u>HopsFS (HDFS)</u>

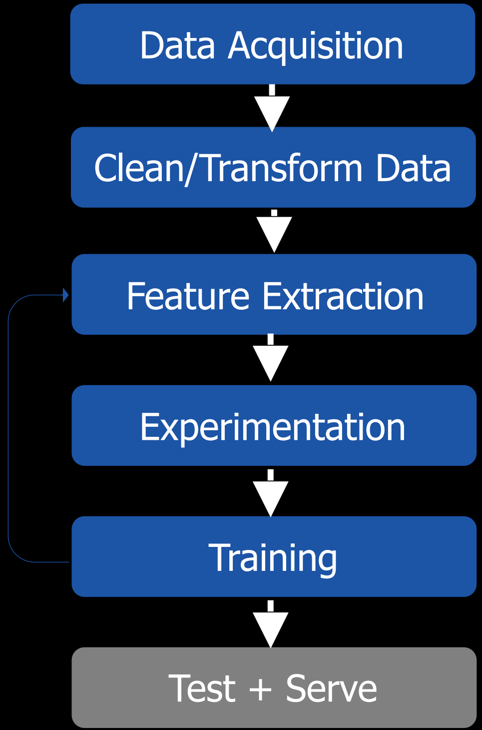
/Experiments/appId/run.ID/<name>

/Experiments/appId/run.ID/<name>/eval

/Experiments/appId/run.ID/<name>/checkpoint

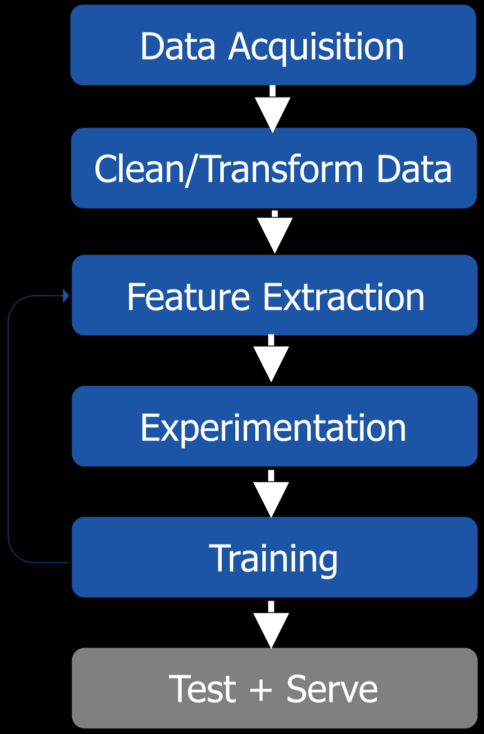
/Experiments/appId/run.ID/<name>/*.ipynb
/Experiments/appId/run.ID/<name>/conda.yml

HopsML CollectiveAllReduceStrategy with Keras




```
#over-simplified code – see notebook
def distributed_training():
  def input_fn(): # return dataset
  model = …
  optimizer = …
  model.compile(…)
  rc = tf.estimator.RunConfig('CollectiveAllReduceStrategy')
  keras_estimator = tf.keras.estimator.model_to_estimator(….)
  tf.estimator.train_and_evaluate(keras_estimator, input_fn)

experiment.collective_all_reduce(distributed_training)
```

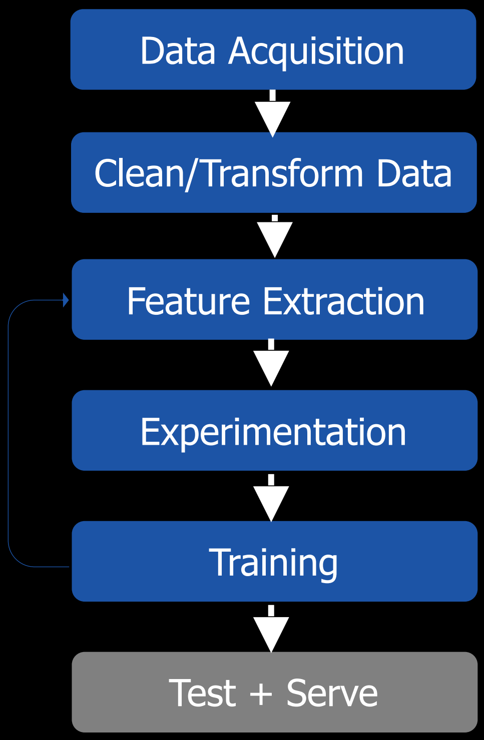
Add Tensorboard Support




```
def distributed_training():
  from hops import tensorboard
  model_dir = tensorboard.logdir()
  def input_fn(): # return dataset
  model = …
  optimizer = …
  model.compile(…)
  rc = tf.estimator.RunConfig('CollectiveAllReduceStrategy')
  keras_estimator = keras.model_to_estimator(model_dir)
  tf.estimator.train_and_evaluate(keras_estimator, input_fn)

experiment.allreduce(distributed_training)
```

GPU Device Awareness

```
def distributed_training():
  from hops import devices
  def input_fn(): # return dataset
  model = …
  optimizer = …
  model.compile(…)
  est.RunConfig(num_gpus_per_worker=devices.get_num_gpus())
  keras_estimator = keras.model_to_estimator(…)
  tf.estimator.train_and_evaluate(keras_estimator, input_fn)

experiment.allreduce(distributed_training)
```

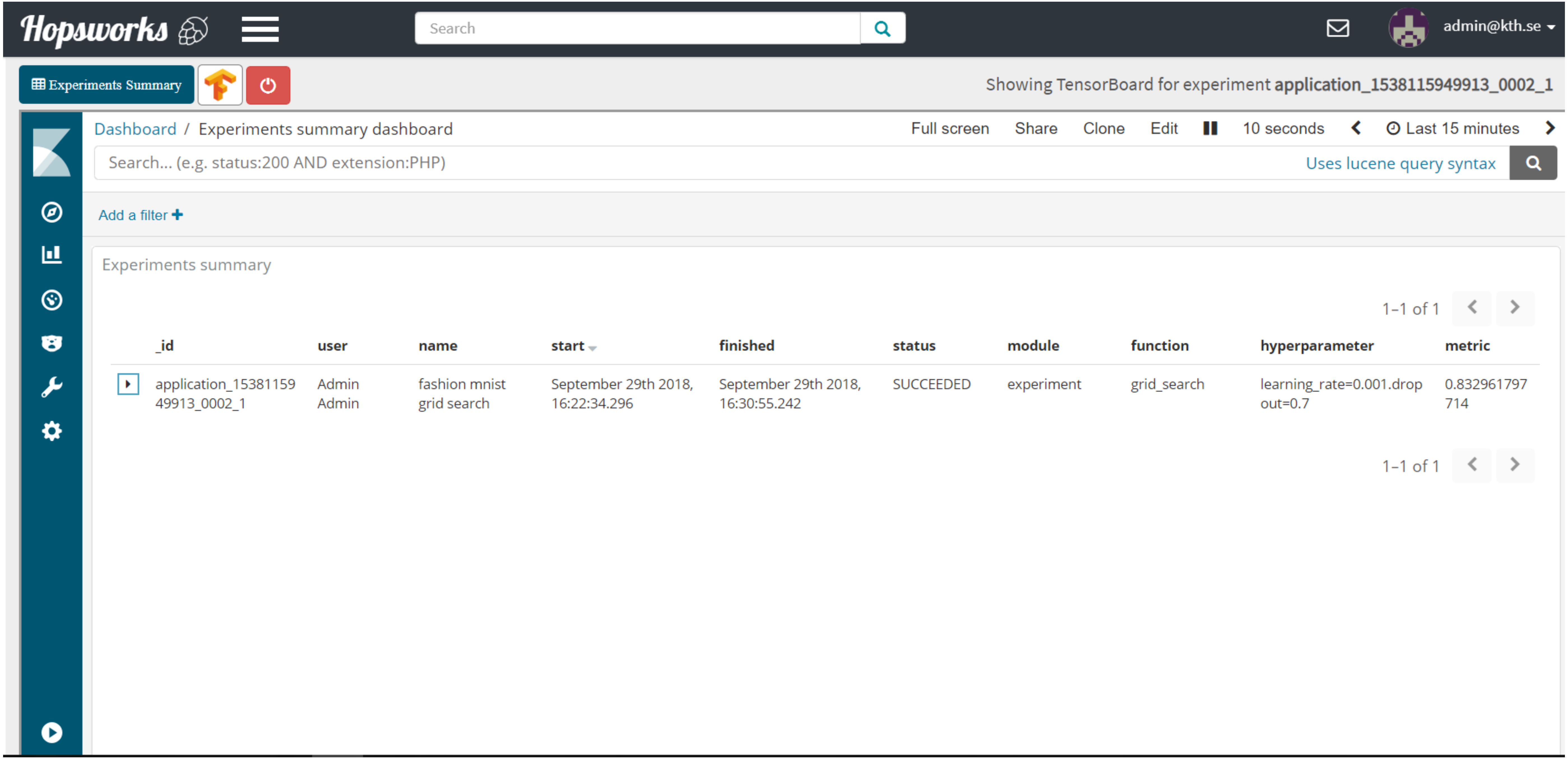


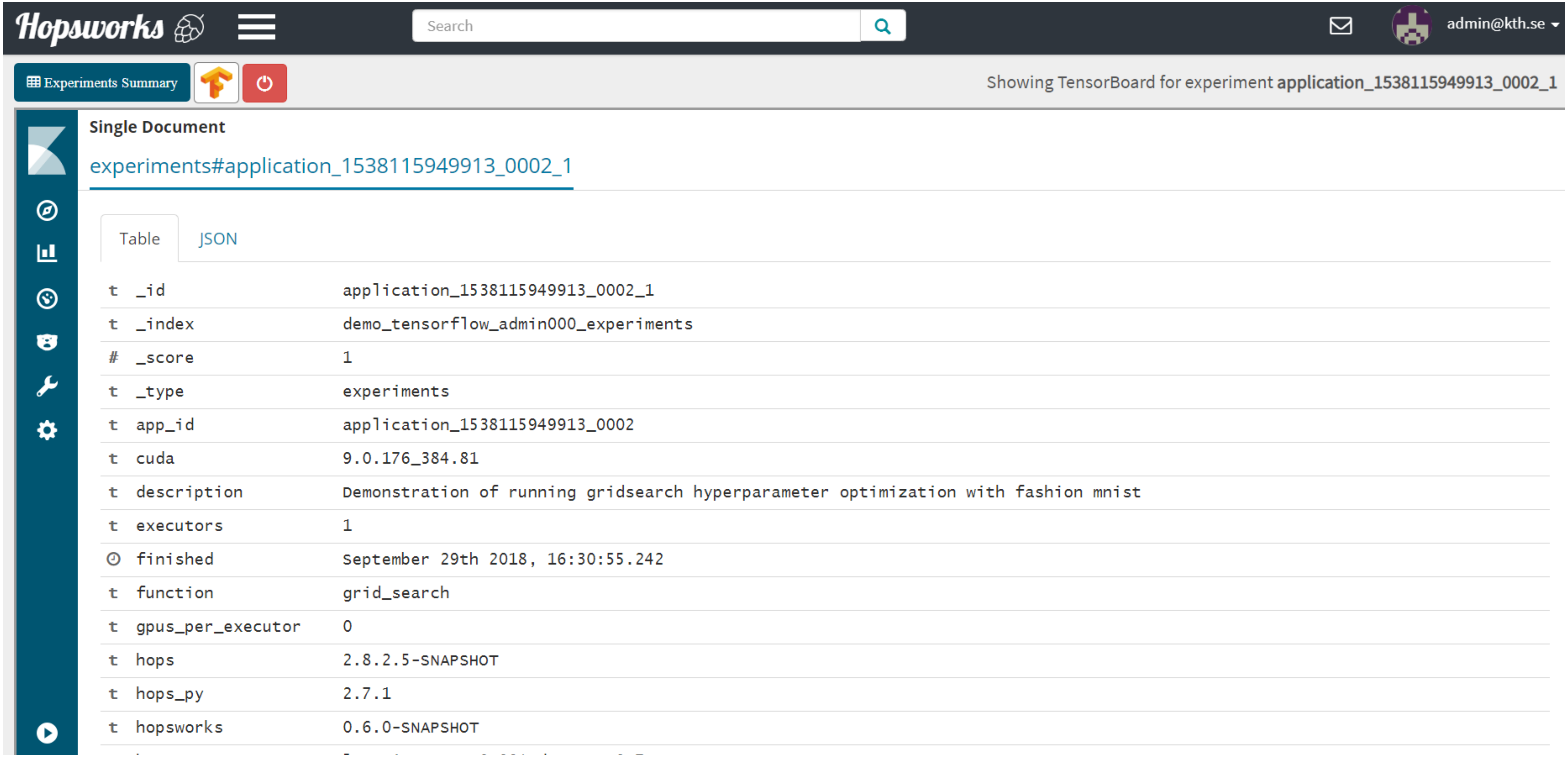
LOGICAL CLOCKS

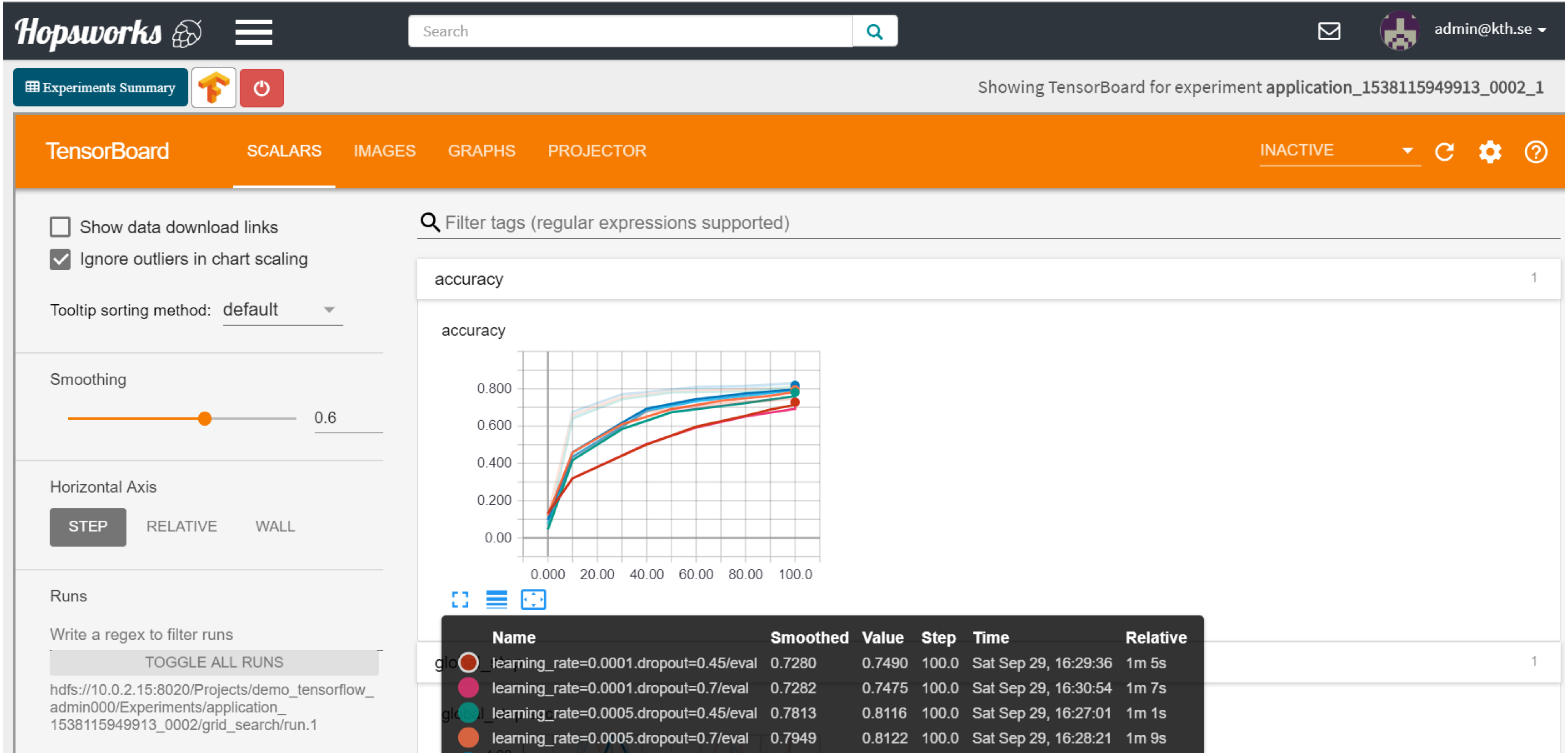
Experiment Versioning (.ipynb, conda, results)

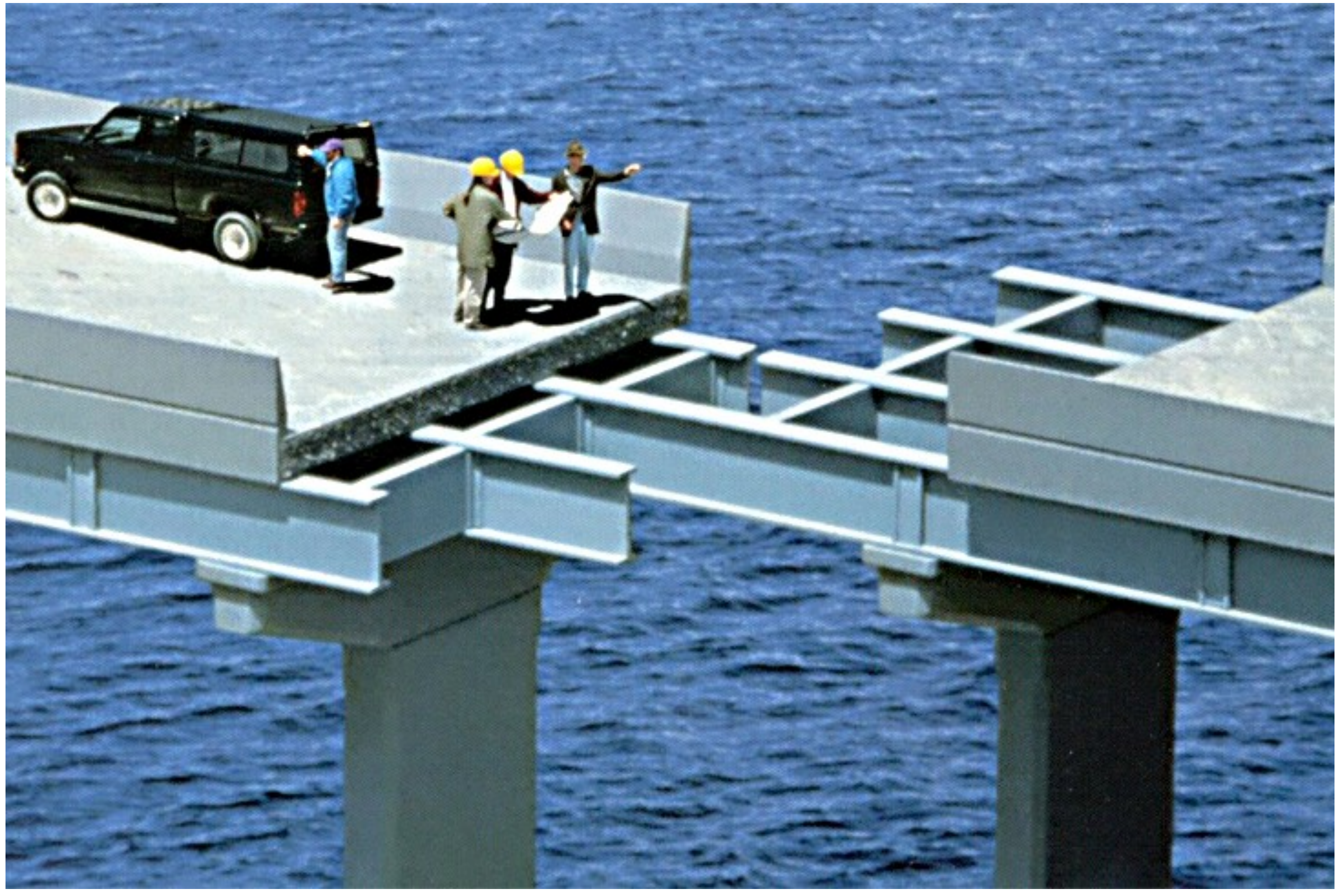
```
def distributed_training():
  def input_fn(): # return dataset
  model = …
  optimizer = …
  model.compile(…)
  rc = tf.estimator.RunConfig('CollectiveAllReduceStrategy')
  keras_estimator = keras.model_to_estimator(…)
  tf.estimator.train_and_evaluate(keras_estimator, input_fn)

notebook = hdfs.project_path()+'/Jupyter/Experiment/inc.ipynb'
experiment.allreduce(distributed_training, name='inception',
   description='A inception example with hidden layers',
   versioned_resources=[notebook])
```

Experiments/Versioning in Hopsworks


Hopsworks
Search
admin@kth.se
Experiments Summary
Showing TensorBoard for experiment application_1538115949913_0002_1
Dashboard / Experiments summary dashboard
Full screen
Share
Clone
Edit
10 seconds
Last 15 minutes
Search... (e.g. status:200 AND extension:PHP)
Uses lucene query syntax
Add a filter
Experiments summary
1–1 of 1
_id
user
name
start
finished
status
module
function
hyperparameter
metric
application_1538115949913_0002_1
Admin Admin
fashion mnist grid search
September 29th 2018, 16:22:34.296
September 29th 2018, 16:30:55.242
SUCCEEDED
experiment
grid_search
learning_rate=0.001.dropout=0.7
0.832961797714
1–1 of 1

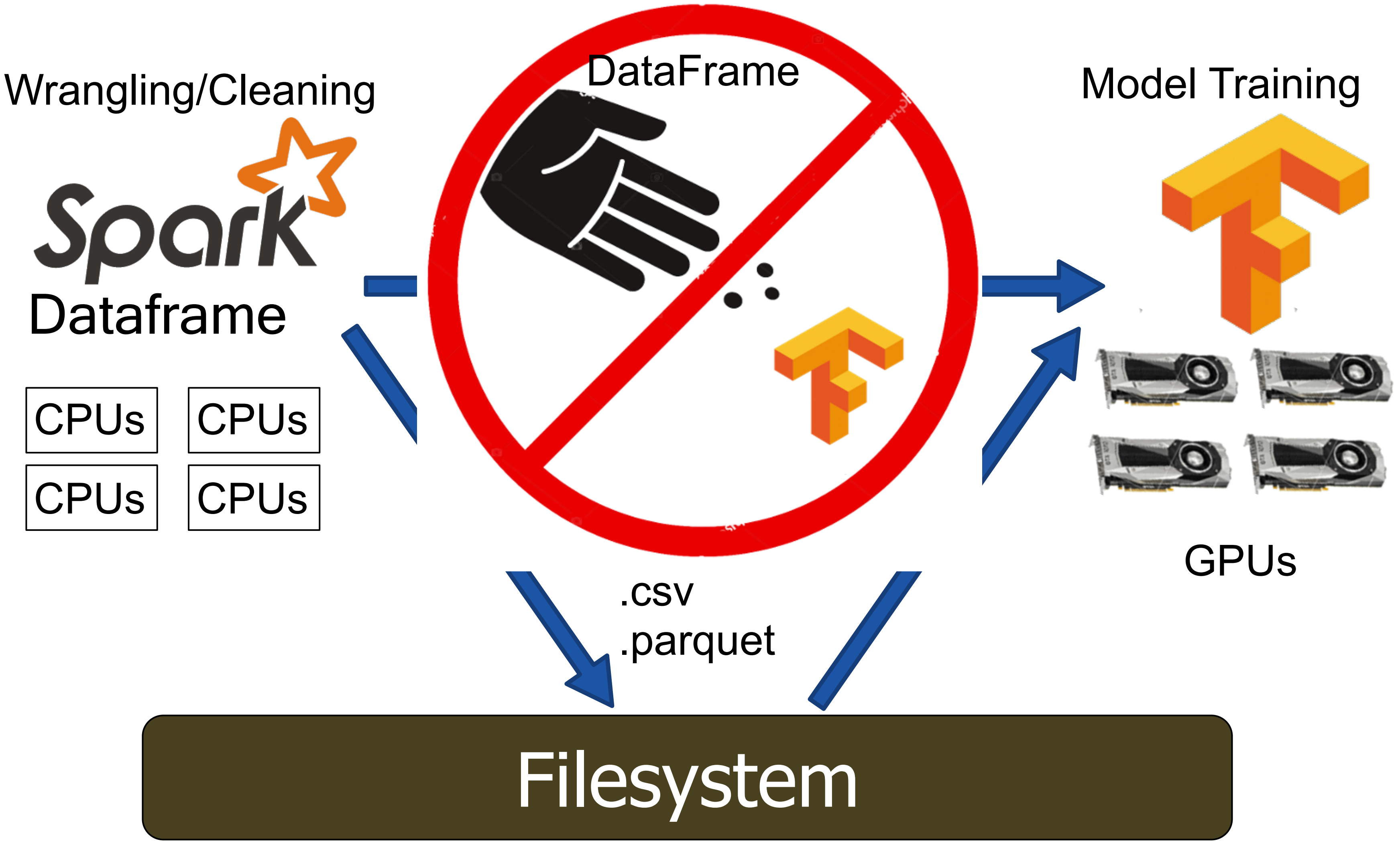

Hopsworks
Search
admin@kth.se
Experiments Summary
Showing TensorBoard for experiment application_1538115949913_0002_1
Single Document
experiments#application_1538115949913_0002_1
Table
JSON
t _id application_1538115949913_0002_1
t _index demo_tensorflow_admin000_experiments
_score 1
t _type experiments
t app_id application_1538115949913_0002
t cuda 9.0.176_384.81
t description Demonstration of running gridsearch hyperparameter optimization with fashion mnist
t executors 1
finished September 29th 2018, 16:30:55.242
t function grid_search
t gpus_per_executor 0
t hops 2.8.2.5-SNAPSHOT
t hops_py 2.7.1
t hopsworks 0.6.0-SNAPSHOT


Hopsworks
Search
admin@kth.se
Experiments Summary
Showing TensorBoard for experiment application_1538115949913_0002_1
TensorBoard
SCALARS
IMAGES
GRAPHS
PROJECTOR
INACTIVE
Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
hdfs://10.0.2.15:8020/Projects/demo_tensorflow_admin000/Experiments/application_1538115949913_0002/grid_search/run.1
Filter tags (regular expressions supported)
accuracy
1
accuracy
0.800
0.600
0.400
0.200
0.00
0.000
20.00
40.00
60.00
80.00
100.0
Name
Smoothed
Value
Step
Time
Relative
learning_rate=0.0001.dropout=0.45/eval 0.7280 0.7490 100.0 Sat Sep 29, 16:29:36 1m 5s
learning_rate=0.0001.dropout=0.7/eval 0.7282 0.7475 100.0 Sat Sep 29, 16:30:54 1m 7s
learning_rate=0.0005.dropout=0.45/eval 0.7813 0.8116 100.0 Sat Sep 29, 16:27:01 1m 1s
learning_rate=0.0005.dropout=0.7/eval 0.7949 0.8122 100.0 Sat Sep 29, 16:28:21 1m 9s



The Data Layer (Foundations)

Feeding Data to TensorFlow



Project Hydrogen: Barrier Execution mode in Spark: JIRA: SPARK-24374, SPARK-24723, SPARK-24579

Existing Filesystems are not good enough?

Uber on Petastorm:

"[Using files] is hard to implement at large scale, especially using modern distributed file systems such as HDFS and S3 (these systems are typically optimized for fast reads of large chunks of data)."

https://eng.uber.com/petastorm/

PetaStorm: Read Parquet directly into TensorFlow

```
with Reader('hdfs://myhadoop/dataset.parquet') as reader:
    dataset = make_petastorm_dataset(reader)
    iterator = dataset.make_one_shot_iterator()
    tensor = iterator.get_next()
    with tf.Session() as sess:
        sample = sess.run(tensor)
        print(sample.id)
```

NVMe Disks – Game Changer

- HDFS (and S3) are designed around large blocks (optimized to overcome slow random I/O on disks), while new **NVMe hardware supports orders of magnitude faster random disk I/O**.

- Can we support faster random disk I/O with HDFS?
  - Yes with HopsFS.

Small files on NVMe disks?*

- At Spotify's HDFS:
  - 33% of files < 64KB in size
  - 42% of operations are on files < 16KB in size

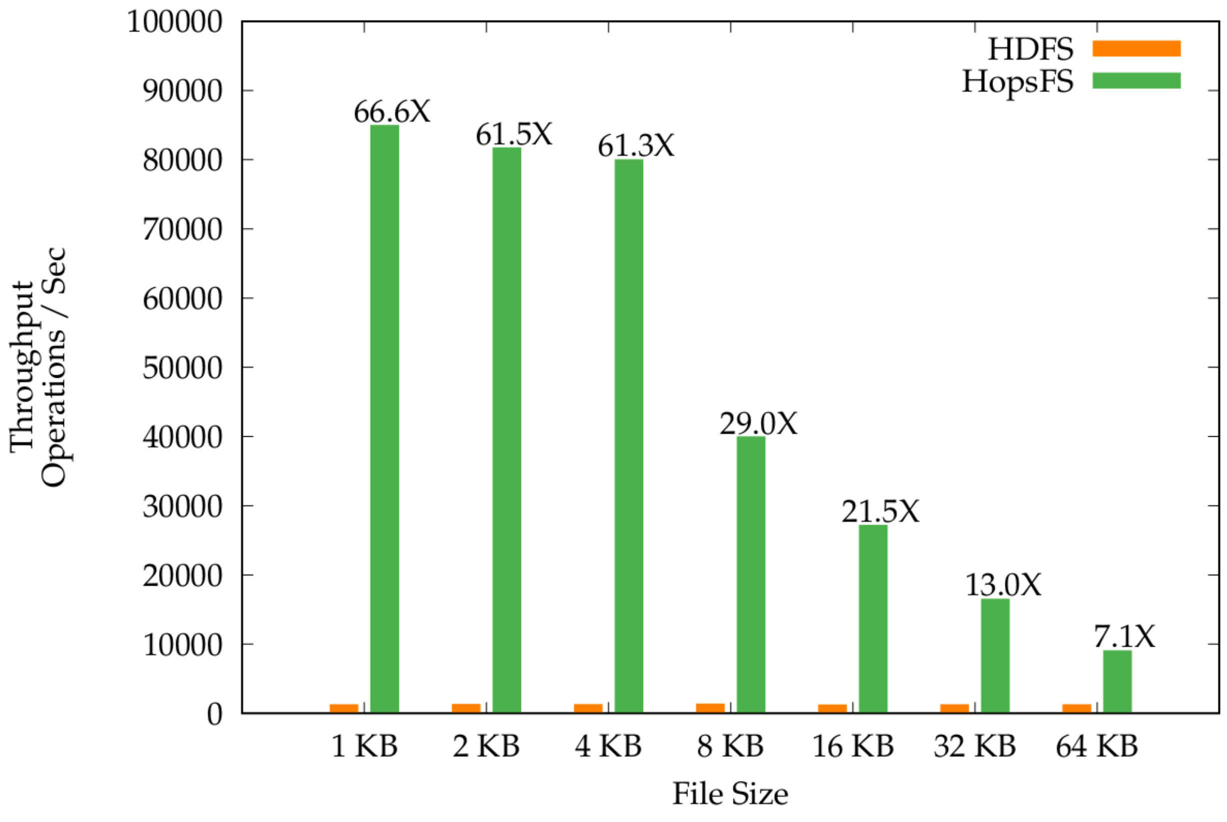
- Similar statistics from Yahoo!'s Hadoop clusters.

- **Solution**: Keep the same large block size, but store the small files in HopsFS' metdata layer.

*Size Matters: Improving the Performance of Small Files in Hadoop, Middleware 2018. Niazi et al

HopsFS – NVMe Performance for Small Files*

- HopsFS is HDFS with Distributed Metadata
- Small files stored replicated in the metadata layer on NVMe disks*
  - Read 10s of 1000s of images/second from HopsFS

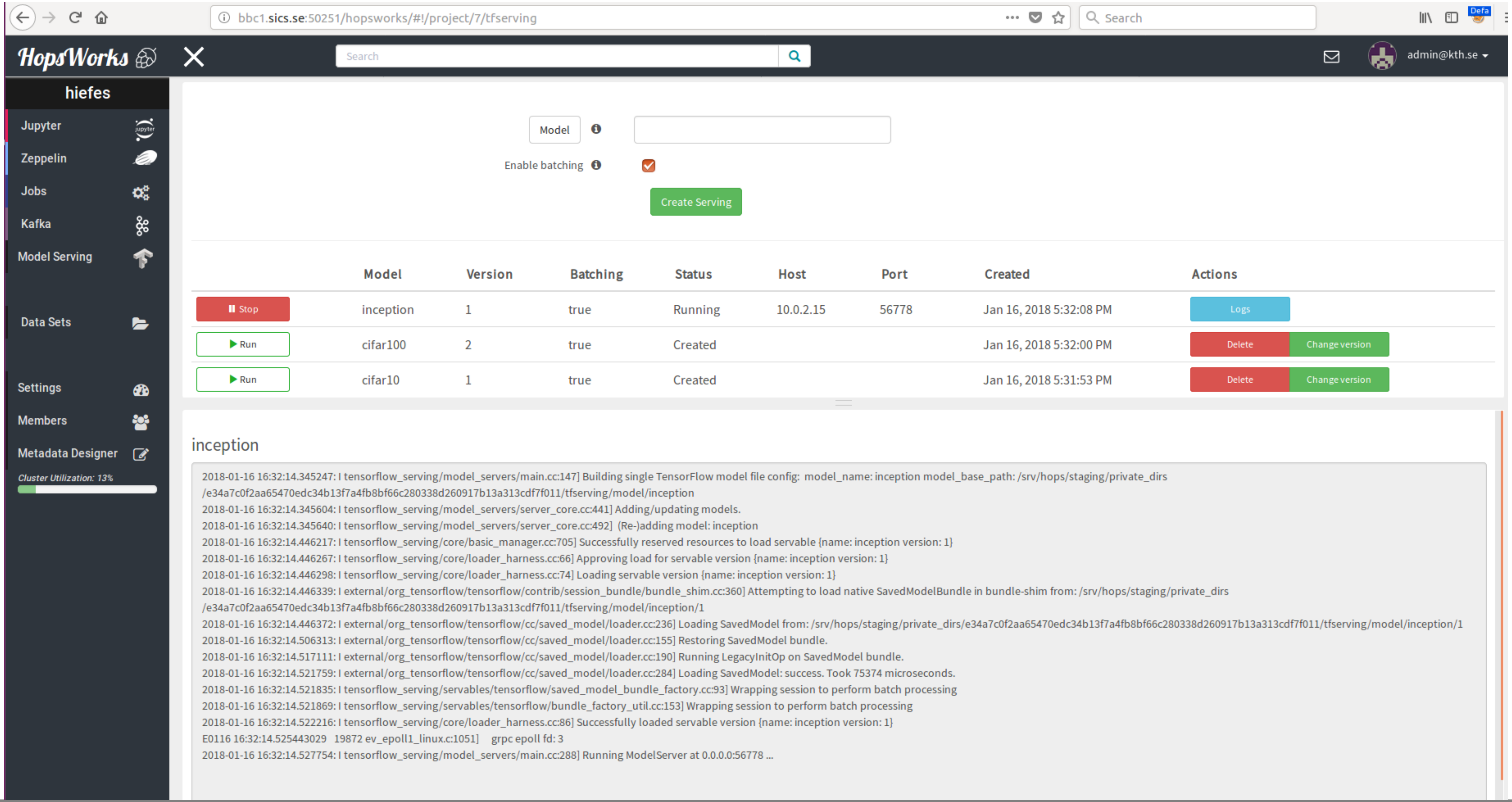


a. File Write Performance

*Size Matters: Improving the Performance of Small Files in Hadoop, Middleware 2018. Niazi et al

Model Serving

Model Serving on Kubernetes



Training-Serving Skew

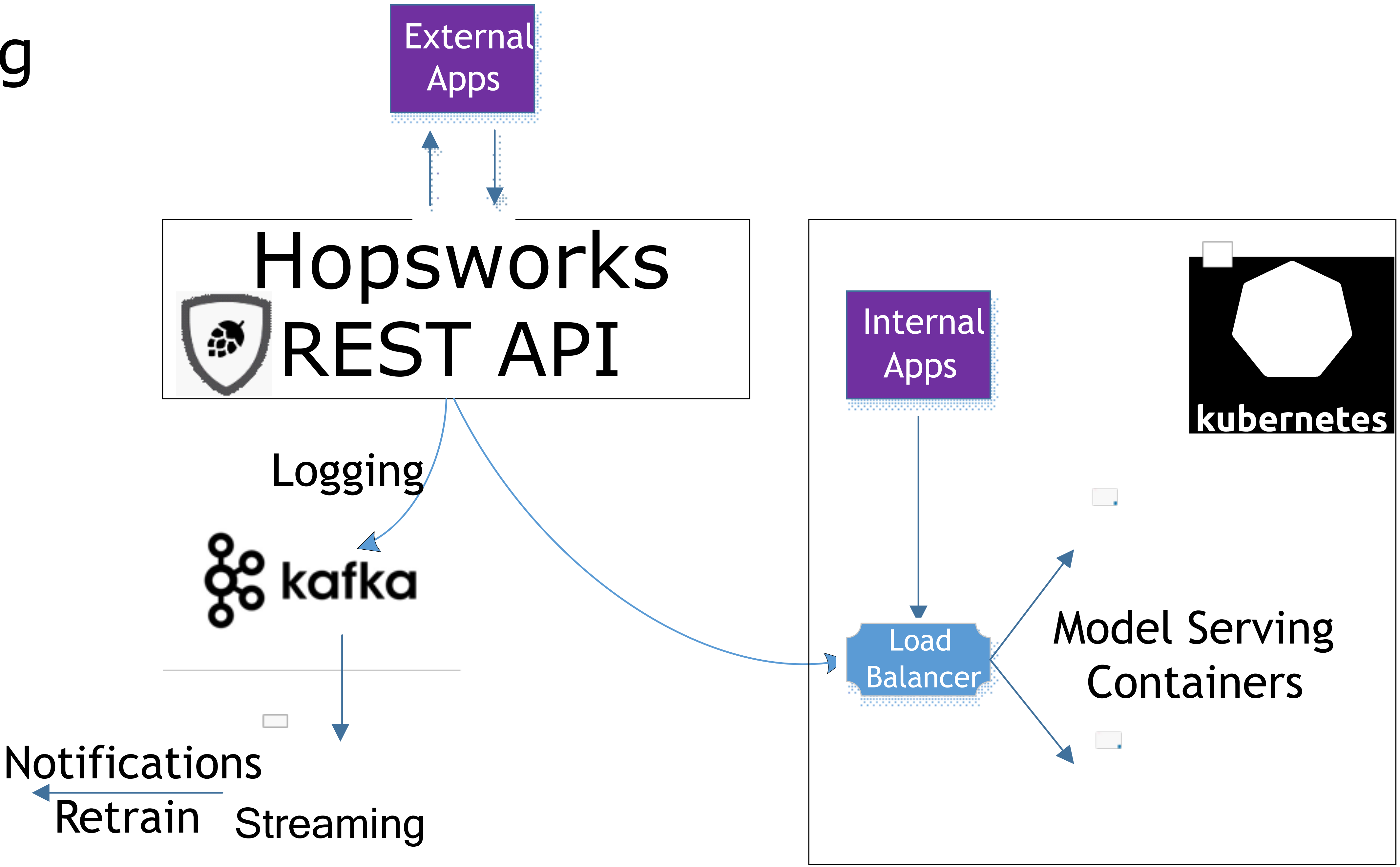
- Monitor differences between performance during training and performance during serving.
  - Differences in how you process data in training vs serving.
  - Differences in the training data and live data for serving.
  - A feedback loop between your model and your algorithm.

- When to retrain?
  - If you look at the input data and use **covariant shift** to see when it deviates significantly from the data that was used to train the model on.

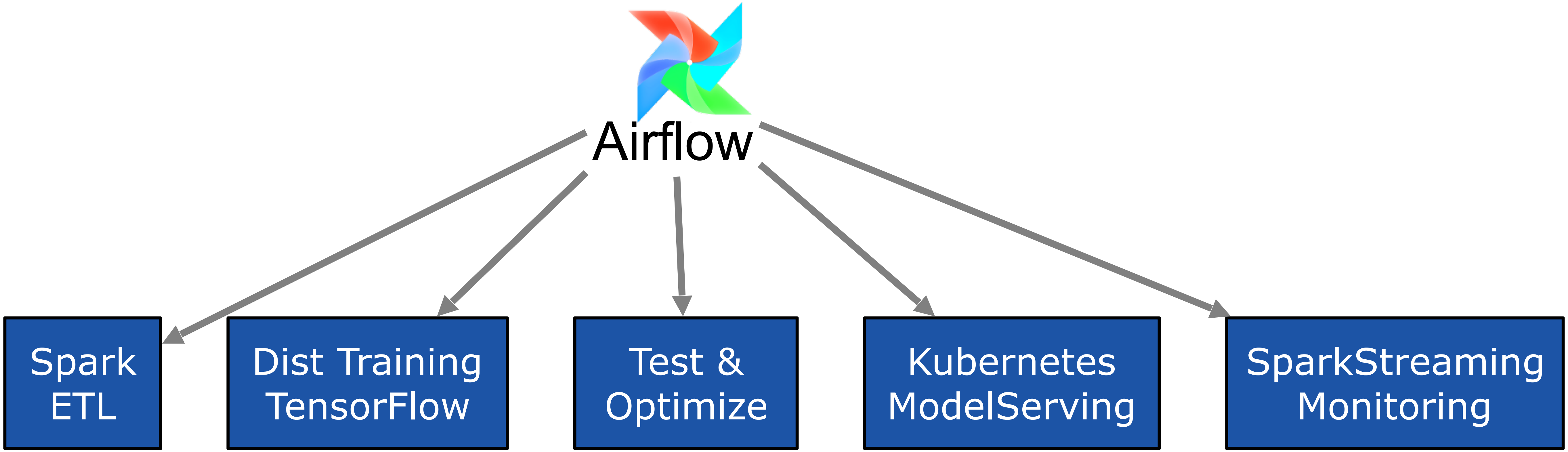
Model Serving

Features:
- Canary
- Multiple Models
- Scale-Out/In

Frameworks:
- ✓ TensorFlow Serving
- ✓ MLeap for Spark
- ✓ scikit-learn

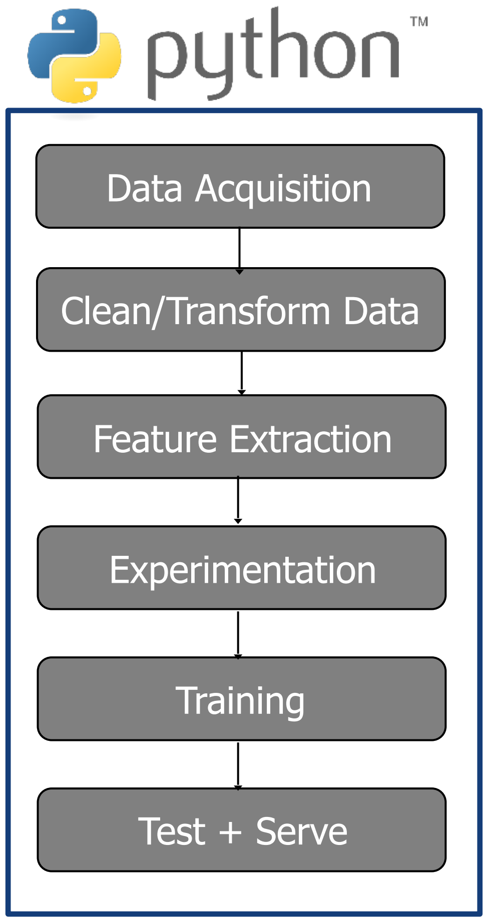


Orchestrating ML Pipelines with Airflow



HopsML

- Experiments
  - Dist. Hyperparameter Optimization
  - Versioning of Models/Code/Resources
  - Visualization with Tensorboard
  - Distributed Training with checkpointing
- Feature Store
- Model Serving and Monitoring



Want to try Hopsworks?

1. Register for an account at:

www.hops.site

Use an email address from a Swedish university or company.

Summary

- The future of Deep Learning is Distributed
  https://www.oreilly.com/ideas/distributed-tensorflow

- Hopsworks is a new Data Platform with first-class support for Python / Deep Learning / ML / Data Governance / GPUs

- Hopsworks is open-source

*https://twitter.com/karpathy/status/972701240017633281

The Team

Active:

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

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, Roshan Sedar.

www.hops.io

@hopshadoop