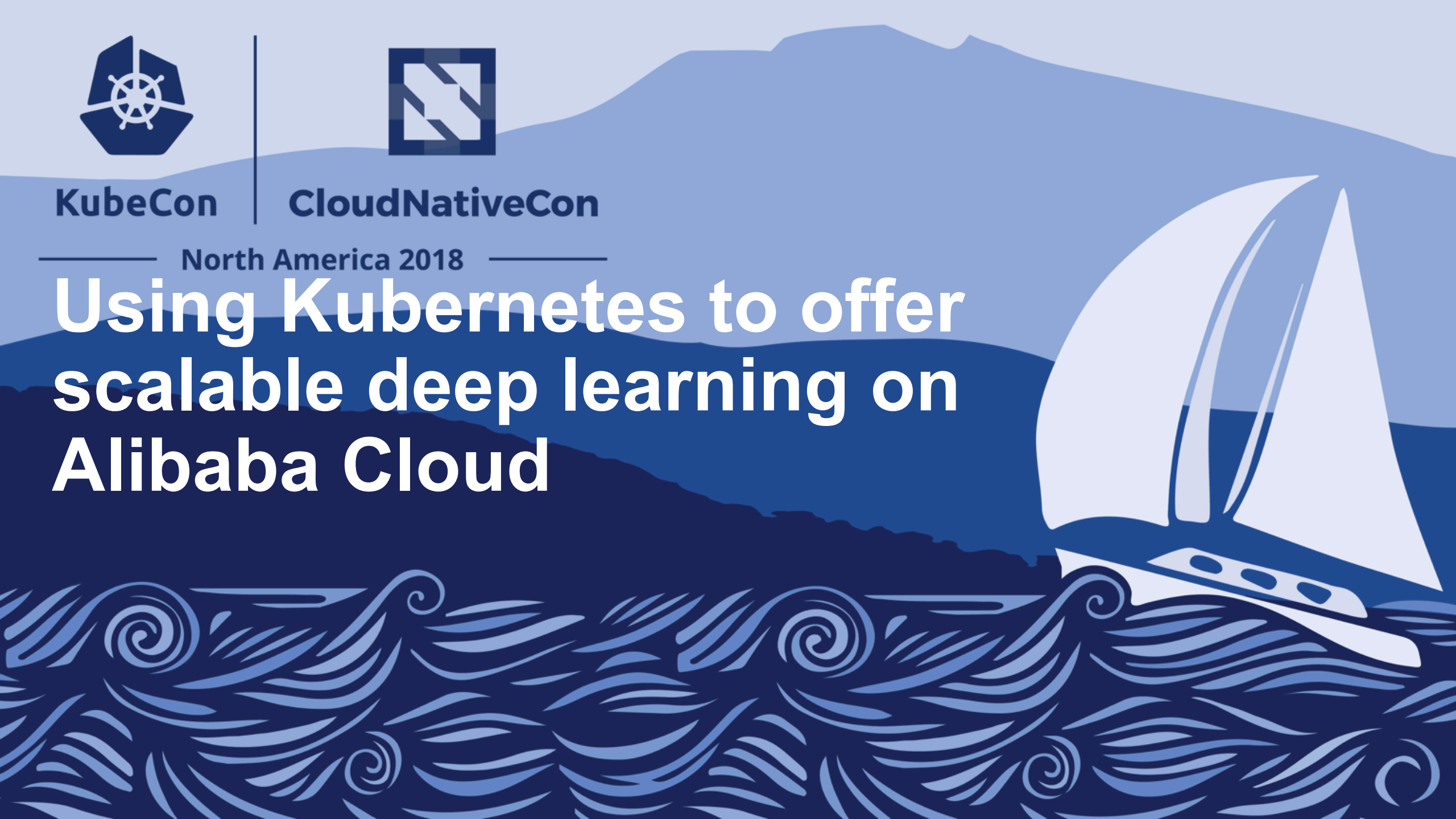
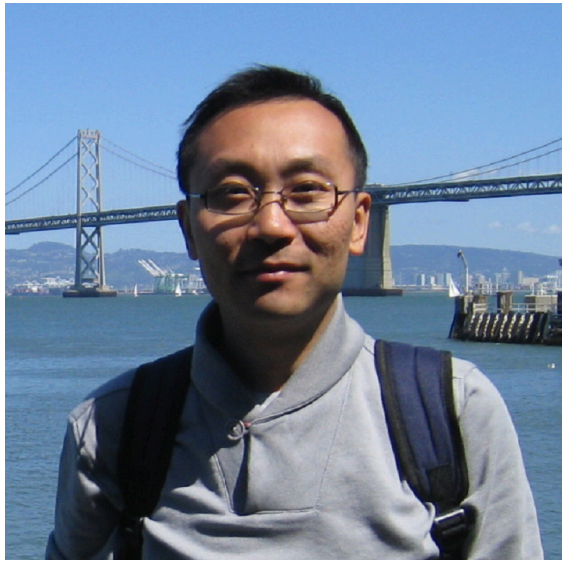
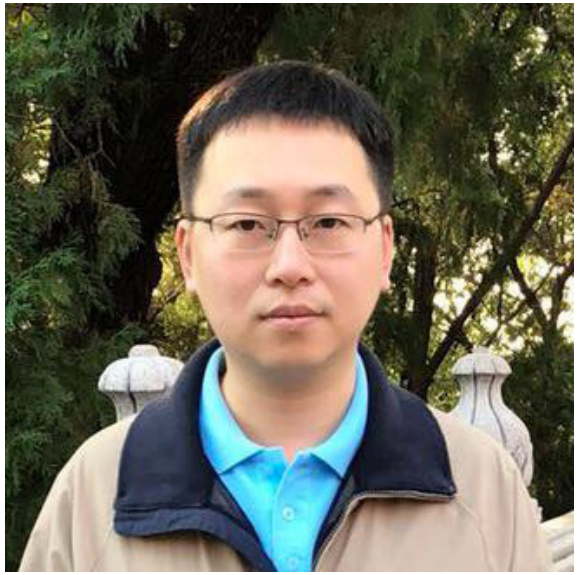


Using Kubernetes to offer scalable deep learning on Alibaba Cloud

Who are we?



Kai Zhang
Staff engineer of Alibaba Cloud



Yang Che
Senior engineer of Alibaba Cloud

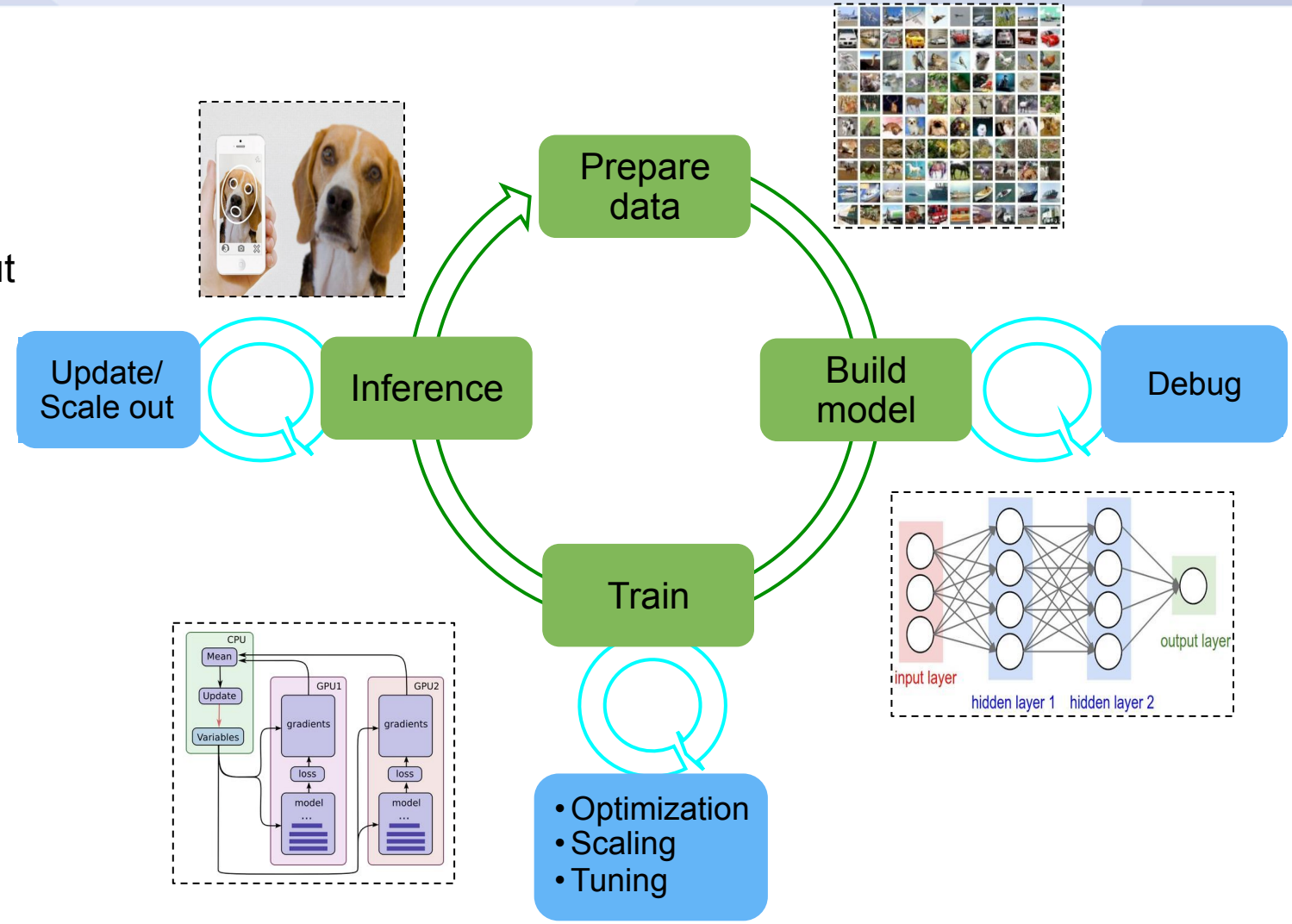
Container service, Kubernetes, Deep learning platform

Agenda

- Challenges of running large scale deep learning
- Container based solution on Alibaba cloud
- Kubeflow and Arena
- Key requirements in real life
- User cases
- Future works

How does data scientists run deep learning?

- ✓ **End to end** - Data in, executable out
- ✓ **Long time** - hours/days/weeks
- ✓ **Iterative optimization** - gradient descent, hyper parameters tuning
- ✓ **Massive data, massive computation**

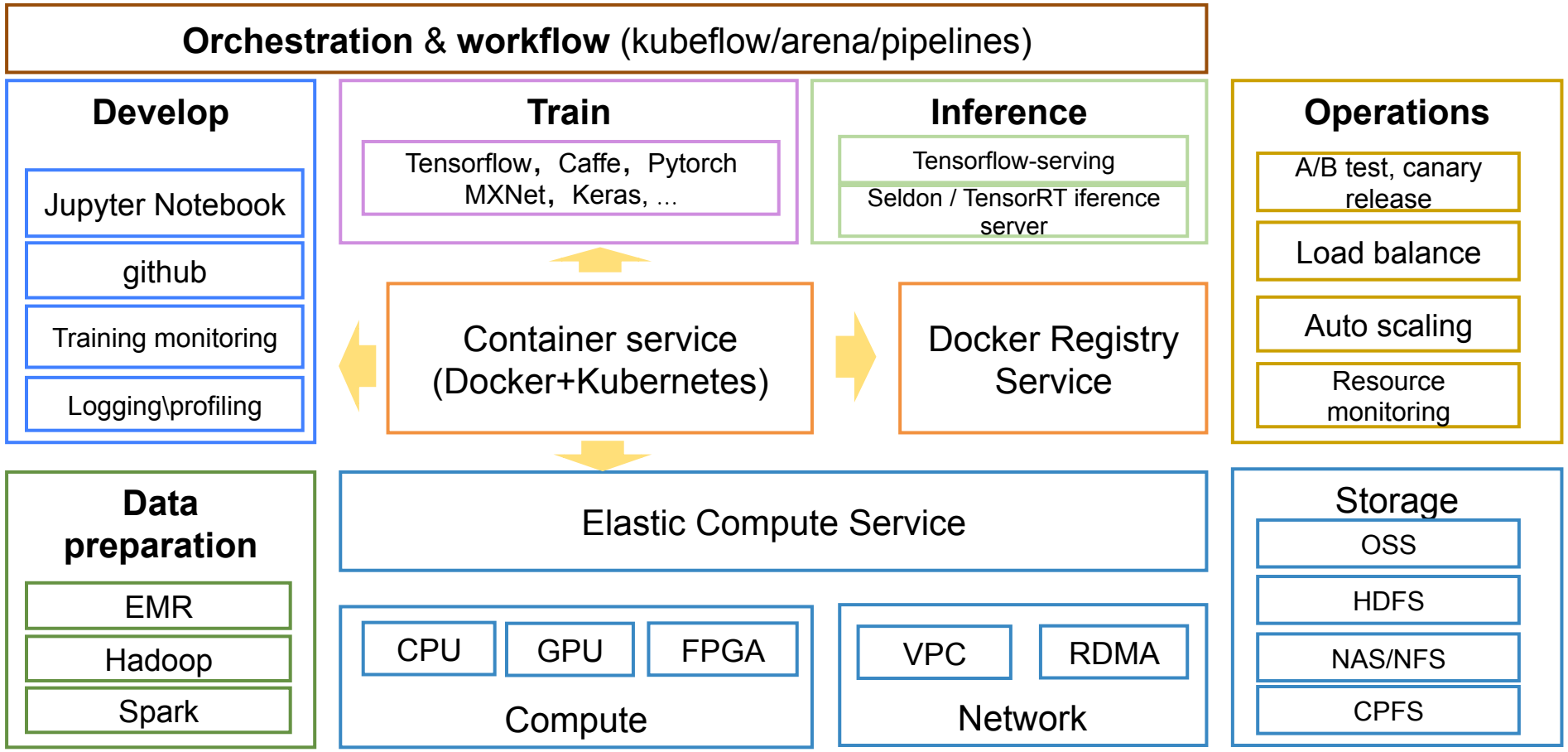


Challenges of deep learning at scale

1. Heterogeneous computing resources management
   - CPU, GPU, (X)PU, FPGA, RDMA
2. End to end support for deep learning experiments
   - Prepare data -> build/train/evaluate model -> release model, and repeat !!!
3. Continuously train and serve models at large scale
   - Cost effective scaling on demand

Deep learning solution on Alibaba Cloud container service

Reference architecture

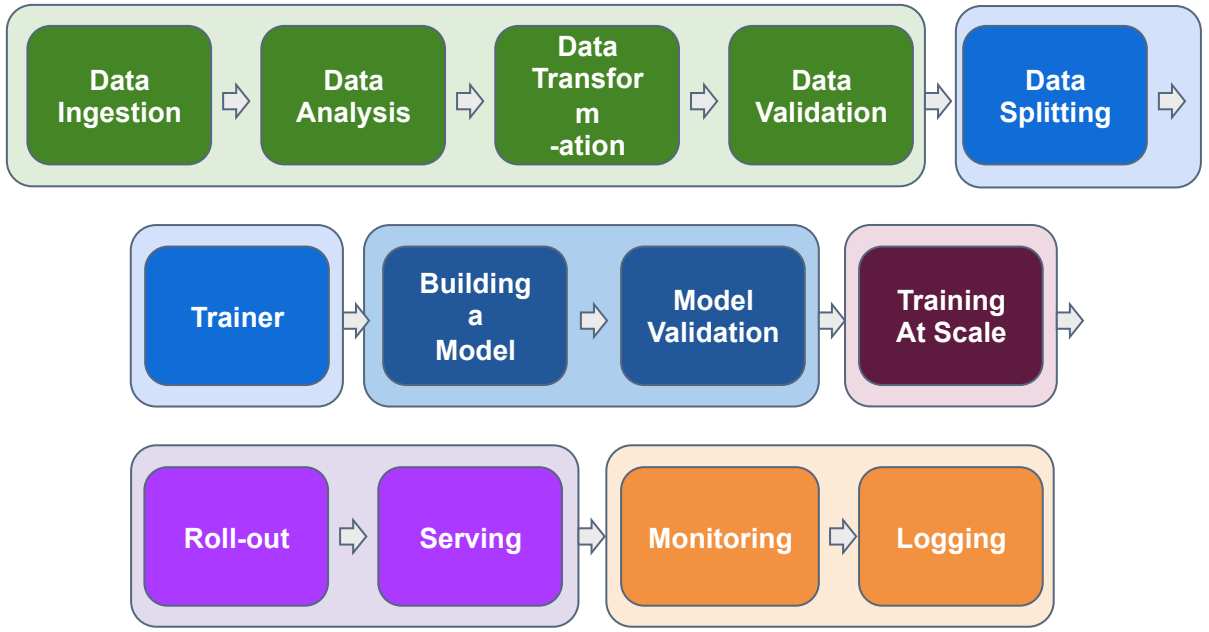


Agenda

- Challenges of running large scale deep learning
- Container based solution on Alibaba cloud
- Kubeflow and Arena
- Key requirements in real life
- User cases
- Future works

How to accelerate deep learning

- **Composability**
  - Composable workflow
  - Continuous training pipelines
  - Auto hyper-parameter tuning
  - adaptive job scheduling
- **Scalability**
  - Scale out training job to hundreds of nodes
- **Portability**
  - Support diverse accelerators like GPU/TPU/FPGA/RDMA
  - Immutable environment covers different frameworks, library, dependencies across on promise and cloud



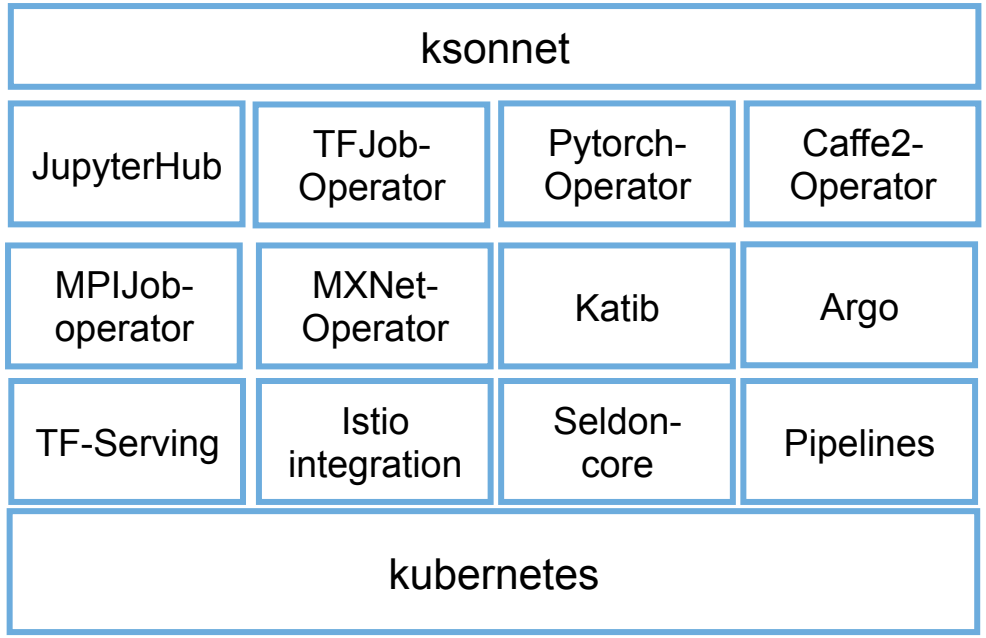
* Thanks kubeflow community for the wonderful flow diagram

Kubeflow – Build portable machine learning solutions using Kubernetes

- Kubeflow's goal is to make scaling machine learning models and deploying them to production as simple as possible

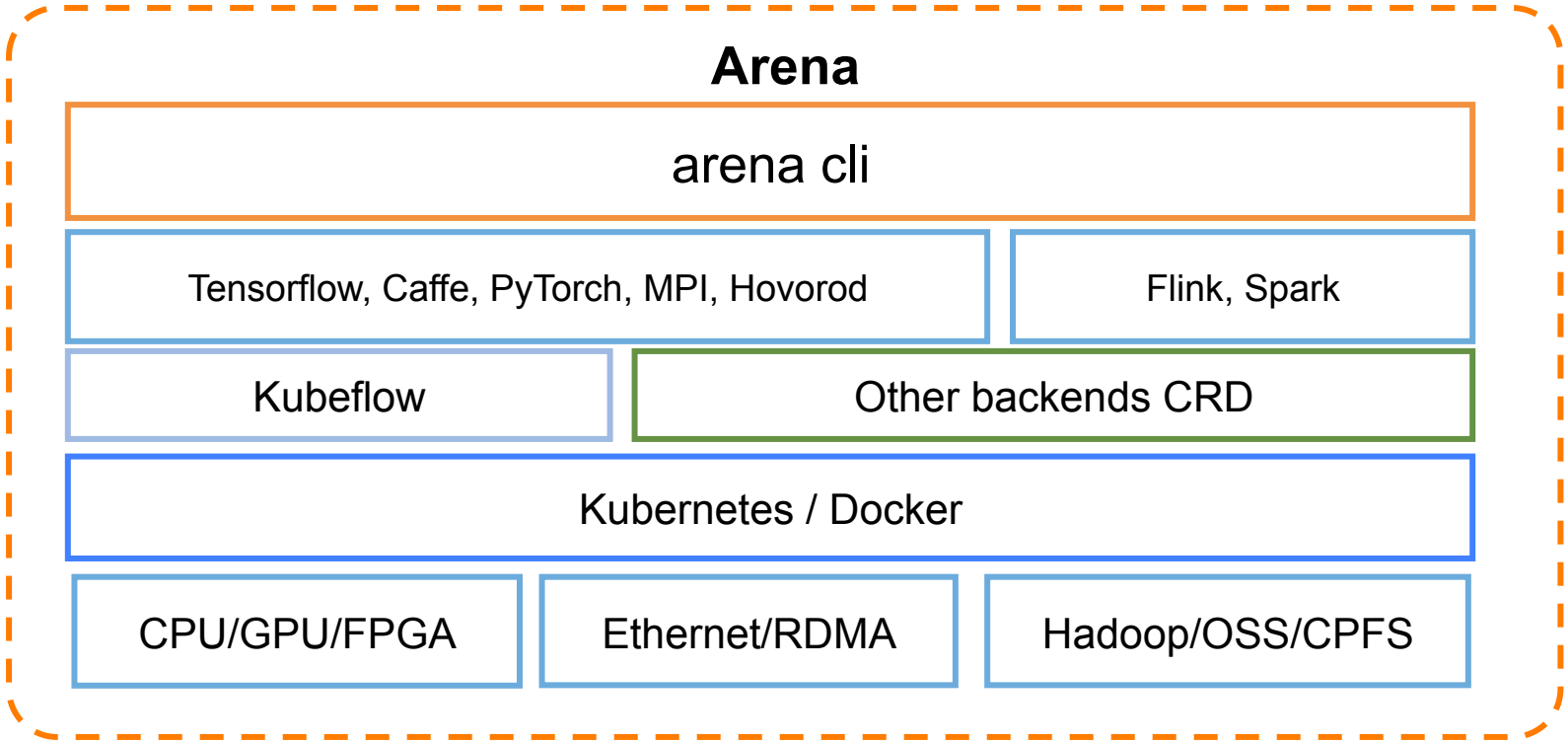
- It's a K8s native platform for machine learning, providing
  - K8s custom resources for managing tasks (distributed training, orchestration, model deployment etc...)
  - microservices for ML (data registries, model databases, hyperparameter tuning, etc...)
  - ksonnect packages to manage K8s infrastructure declaratively

https://github.com/kubeflow



Arena

Arena is open sourced by Alibaba Cloud container service team for accelerating deep learning workloads running on Kubernetes cluster, and making data scientists' life easier.



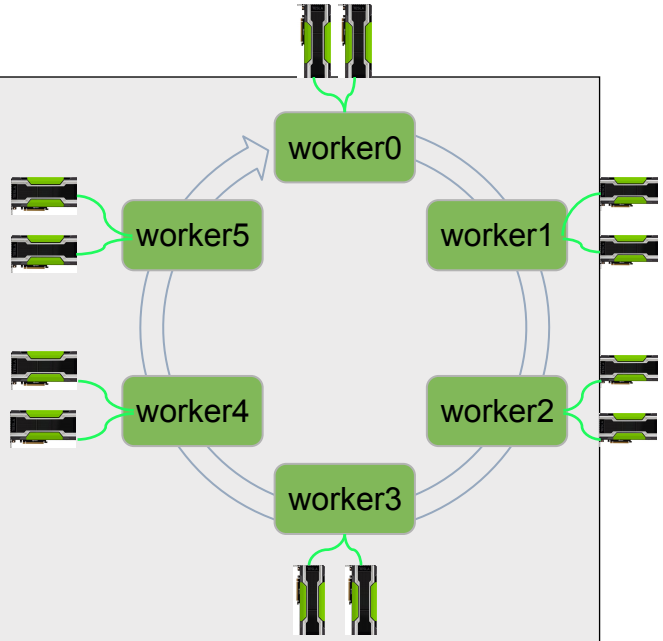
https://github.com/kubeflow/arena

demo

Arena demo – Submit a distributed training job

arena submit mpijob

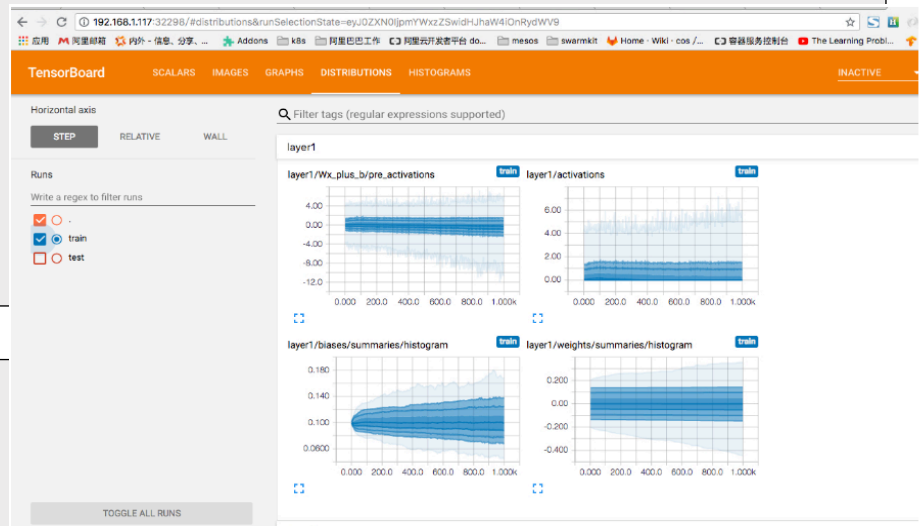
```
        --name=myhvd \
        --workers=6 \
        --gpus=2 \
        --sshPort=33 \
        --syncMode=git \
        --syncSource=https://github.com/xxx/tensorflow-sample-code.git \
        --data=tfdata:/data_dir
        --env=num_batch=100 \
        --env=batch_size=80 \
        --image=registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/ali-perseus:gpu-tf-1.6.0 \
        "/root/hvd-distribute.sh 12 2”
```



Arena demo – check job status


```
# check job list
arena list

NAME            STATUS      TRAINER  AGE    NODE
caffe-1080ti-1  RUNNING     MPIJOB   3d     192.168.1.118
tf-dist-data    SUCCEEDED   TFJOB    3d     N/A
```




```
# check job deatils
arena get tf-dist-data

NAME         STATUS      TRAINER  AGE  INSTANCE                    NODE
tf-dist-data  RUNNING     tfjob    3d   tf-dist-data-tfjob-ps-0     192.168.1.120
tf-dist-data  SUCCEEDED   tfjob    3d   tf-dist-data-tfjob-worker-0  N/A
tf-dist-data  SUCCEEDED   tfjob    3d   tf-dist-data-tfjob-worker-1  N/A

Your tensorboard will be available on:
192.168.1.117:32594
```

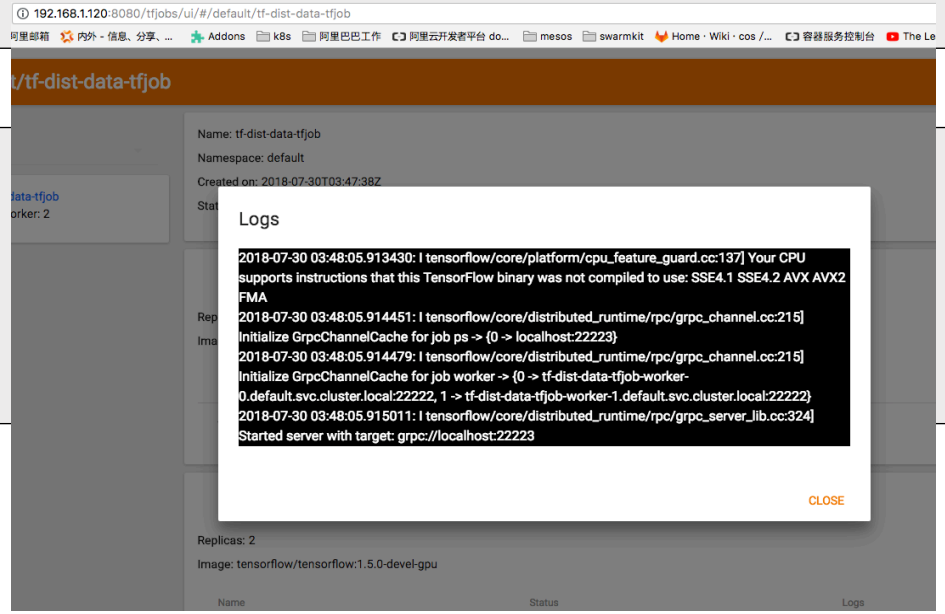
Arena demo – check job log

check real time log
arena logs tf-dist-data

2018-07-30T03:47:49.881380632Z 2018-07-30 03:47:49.881141: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA

check full log
arena logviewer tf-dist-data

Your LogViewer will be available on:
192.168.1.120:8080/tfjobs/ui/#/default/tf-dist-data-tfjob



Arena demo – check GPU status

arena top job style-transfer

INSTANCE NAME	GPU(Device Index)	GPU(Duty Cycle)	GPU(Memory MiB)	STATUS	NODE
style-transfer-tfjob-ps-0	N/A	N/A	N/A	Running	192.168.0.117
style-transfer-tfjob-worker-0	6	98%	15641.0MiB / 16276.2MiB	Running	192.168.0.118
	7	96%	15481.0MiB / 16276.2MiB	Running	192.168.0.118
style-transfer-tfjob-worker-1	3	98%	15641.0MiB / 16276.2MiB	Running	192.168.0.195
	4	95%	15481.0MiB / 16276.2MiB	Running	192.168.0.195

arena top node

NAME	IPADDRESS	ROLE	GPU(Total)	GPU(Allocated)
cn-shanghai.i-ufxxxxxxt76c4lm	192.168.168.124	worker	2	2
cn-shanghai.i-ufxxxxxxt76c4ln	192.168.168.123	worker	2	1
cn-shanghai.i-ufxxxxxxt76c4lo	192.168.168.125	worker	2	1
cn-shanghai.i-ufxxxxxxvggce0x	192.168.168.133	worker	2	2

Allocated/Total GPUs In Cluster:
6/8 (75%)

Arena demo – more commands

SEE ALSO

- arena data - manage data.
- arena delete - delete a training job and its associated pods
- arena get - display details of a training job
- arena list - list all the training jobs
- arena logs - print the logs for a task of the training job
- arena logviewer - display Log Viewer URL of a training job
- arena serve - Serve a job.
- arena submit - Submit a job.
- arena top - Display Resource (GPU) usage.
- arena version - Print version information

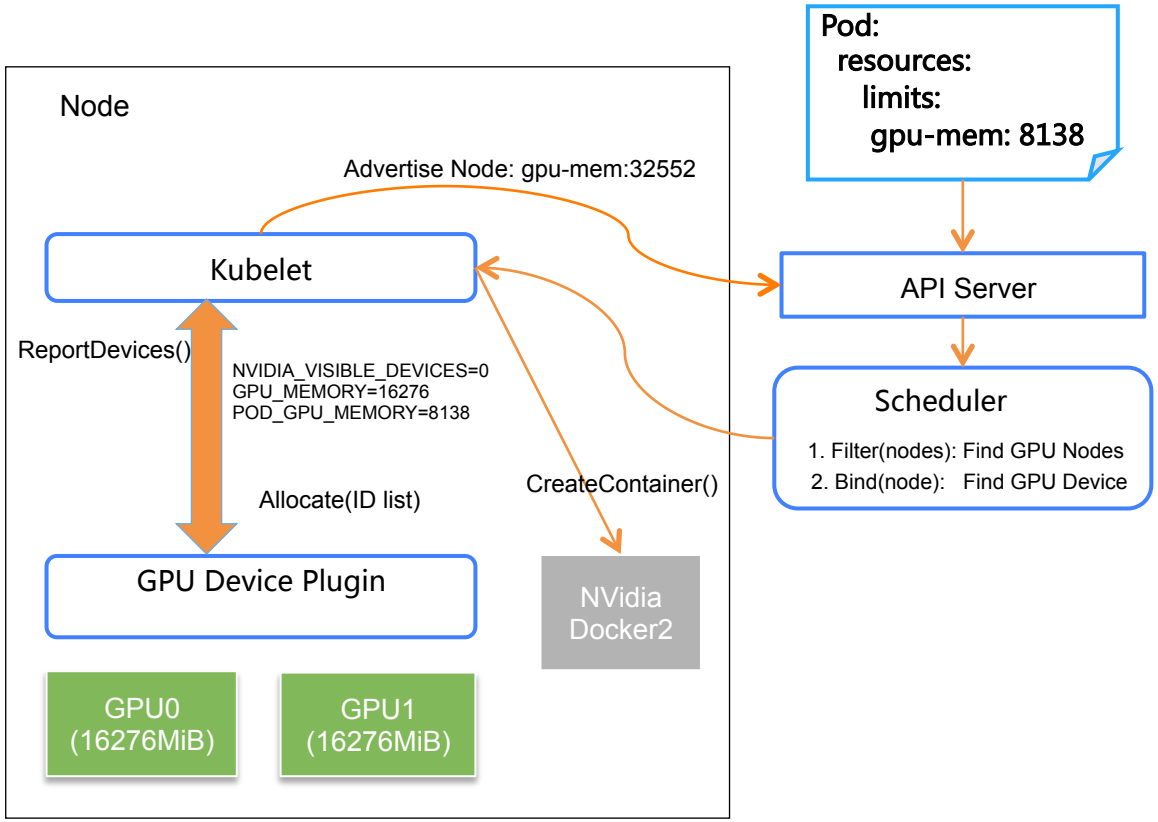
https://github.com/kubeflow/arena

Agenda

- Challenges of running large scale deep learning
- Container based solution on Alibaba cloud
- Kubeflow and Arena
- Key requirements in real life
- User cases
- Future works

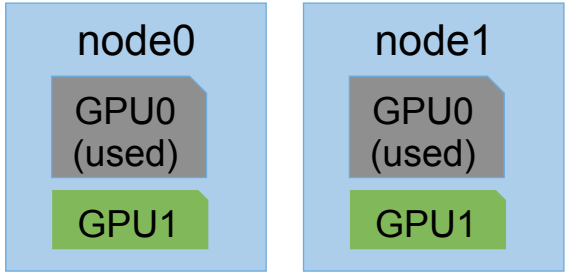
Key requirements – GPU sharing

- Share NVIDIA GPUs among multiple containers to increase utilization for model inference
- The Challenge:
  - Schedule
    - Kubernetes current scheduler enforces exclusive GPU assignment, can't be shared
    - Device Plugin and Scheduler make decision independently
  - Isolation
    - MPS is only for Volta, and not production ready
    - NVIDIA Grid is only for the virtual machine now
- Our solution will be open sourced soon



demo

Key requirements – Gang scheduling

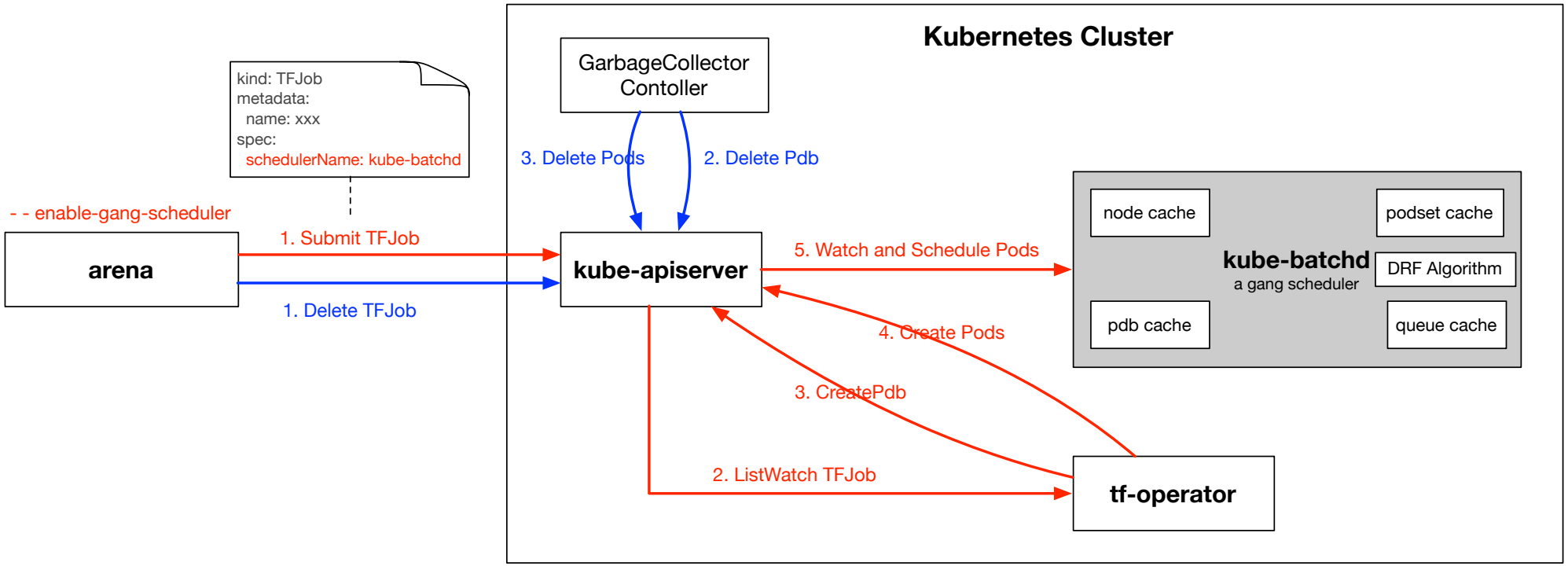


Job1 has 3 pods, each wants 1 GPU

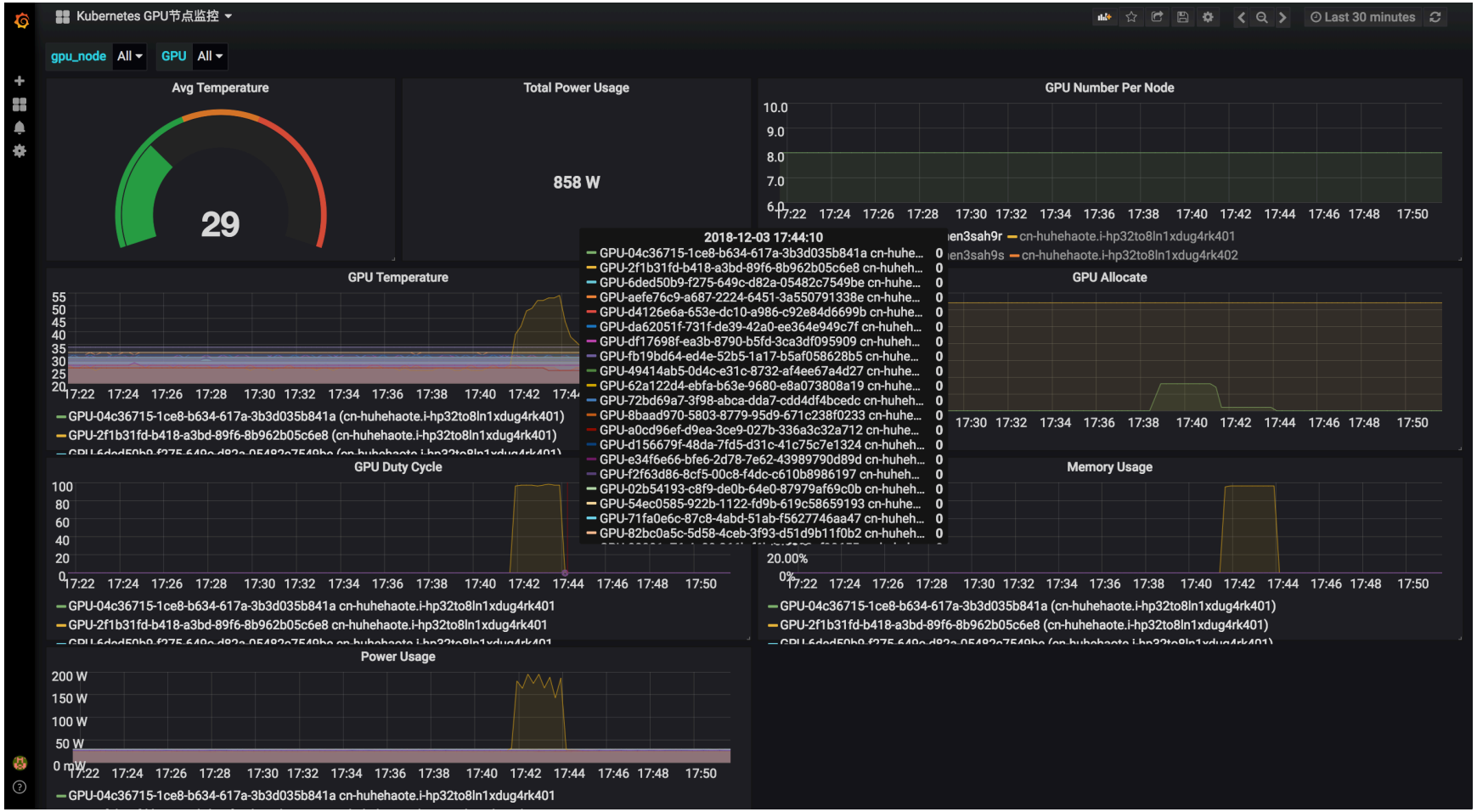
Job1 pending, no pod should get any GPU

Job2 has 1 pod, wants 1 GPU

All or Nothing: Only bind nodes to a job if all of its task pods can be allocated with enough resources



Key requirements – GPU monitoring



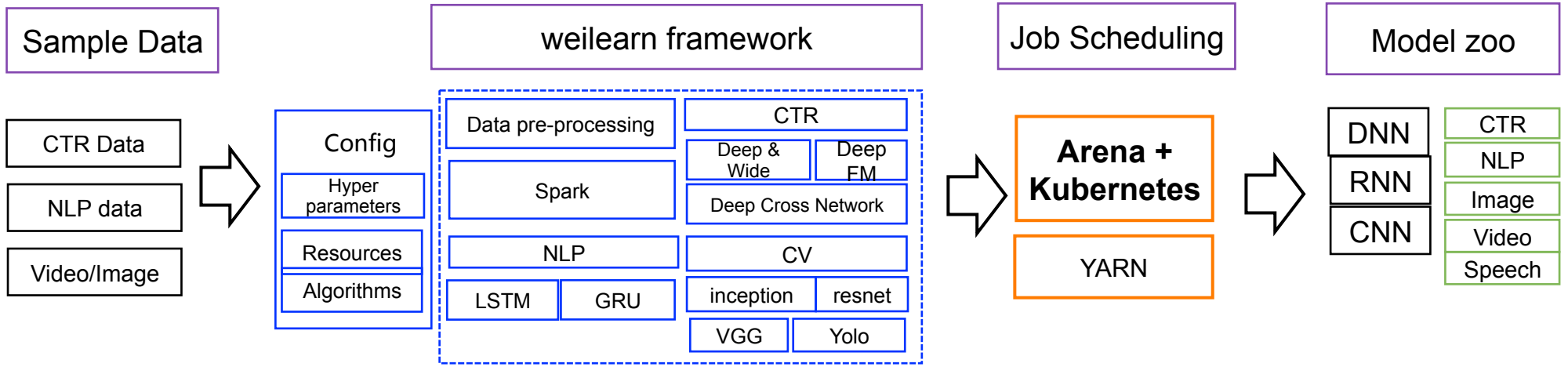
Node level：

- GPU duty cycle
- GPU memory usage
- GPU Temperature
- Power usage
- Total/allocated GPU

Pod level：

- GPU duty cycle
- GPU memory usage
- Allocated GPU

Customer case 1 - weibo's Deep Learning Training Platform



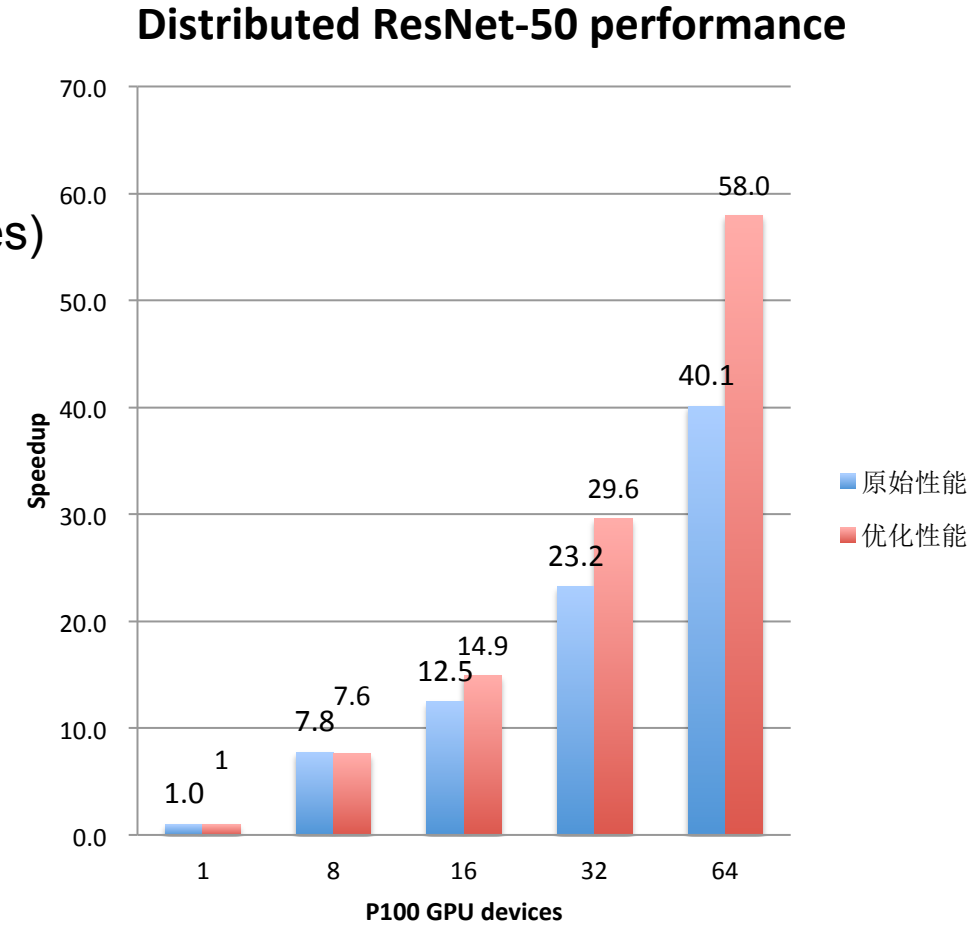
10 millions of feature, 1 billions of sample data

- 200+ GPU nodes cluster
- Unified supports for Tensorflow, Horovod, Caffe
- GPU monitoring and auto scaling
- Real time training visualization and logging
- Create cluster in 10 minutes, start deep learning job in 1 minute

weibo + Alibaba Cloud

Customer case 2 – Accelerate distributed training

- Scenario
  - image classification
- Dataset
  - Imangenet (ILSVRC2012, 1.28million images, 1K classes)
- Model - ResNet-50
- Resources
  - 8xP100 GPU/node, 56 vCPU, 480GB, 25Gb eth
- Framework – Perseus vs. Tensorflow
- Performance optimization
  - MPI + ring-allreduce + FP16
  - Overlap communication and computation
- Results
  - Use 64 GPU get 90% speed up
  - 45% better than native TF



Future works

- Training with serverless Kubernetes and spot instance
  - Cost effective
  - Don’t care about cluster
- Model hub
  - Pre-trained models
  - Reproducible training workflows
- Data/model management
  - Versioning and security
- Comprehensive inference service
  - Framework agnostic
  - Built in support for A/B test, release policy, smart routing and auto scaling

Welcome to try and fix it ! https://github.com/kubeflow/arena

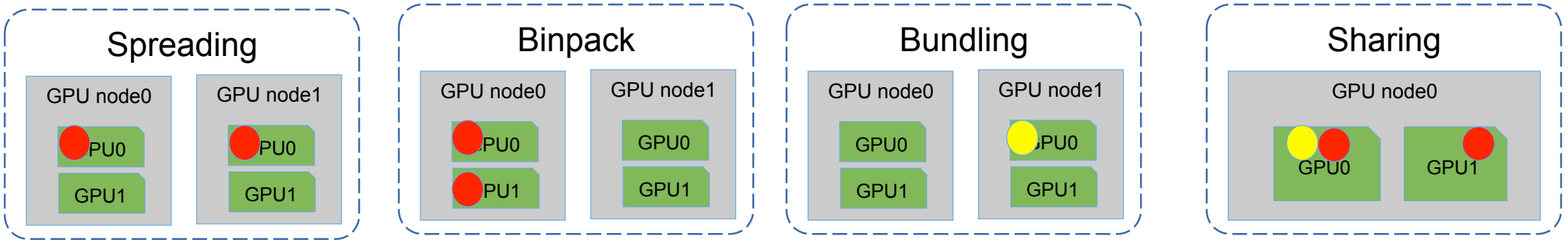
Thank you!

&

Questions?


KubeCon
CloudNativeCon
North America 2018

GPU placement



*Job2 specify it must run on GPU0-node1

Job#1 need 2 GPUs

Job#2 need 1 GPU