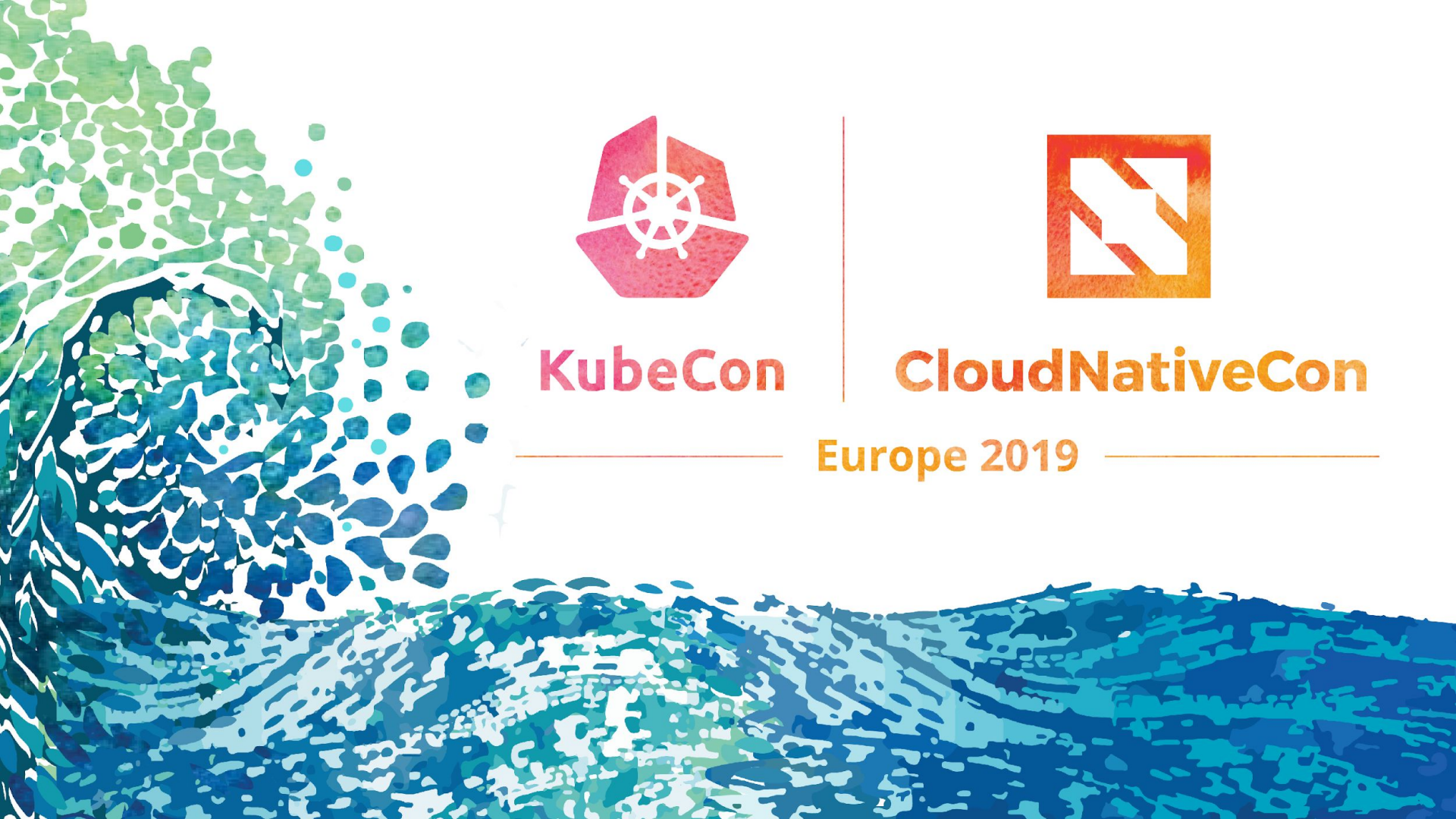
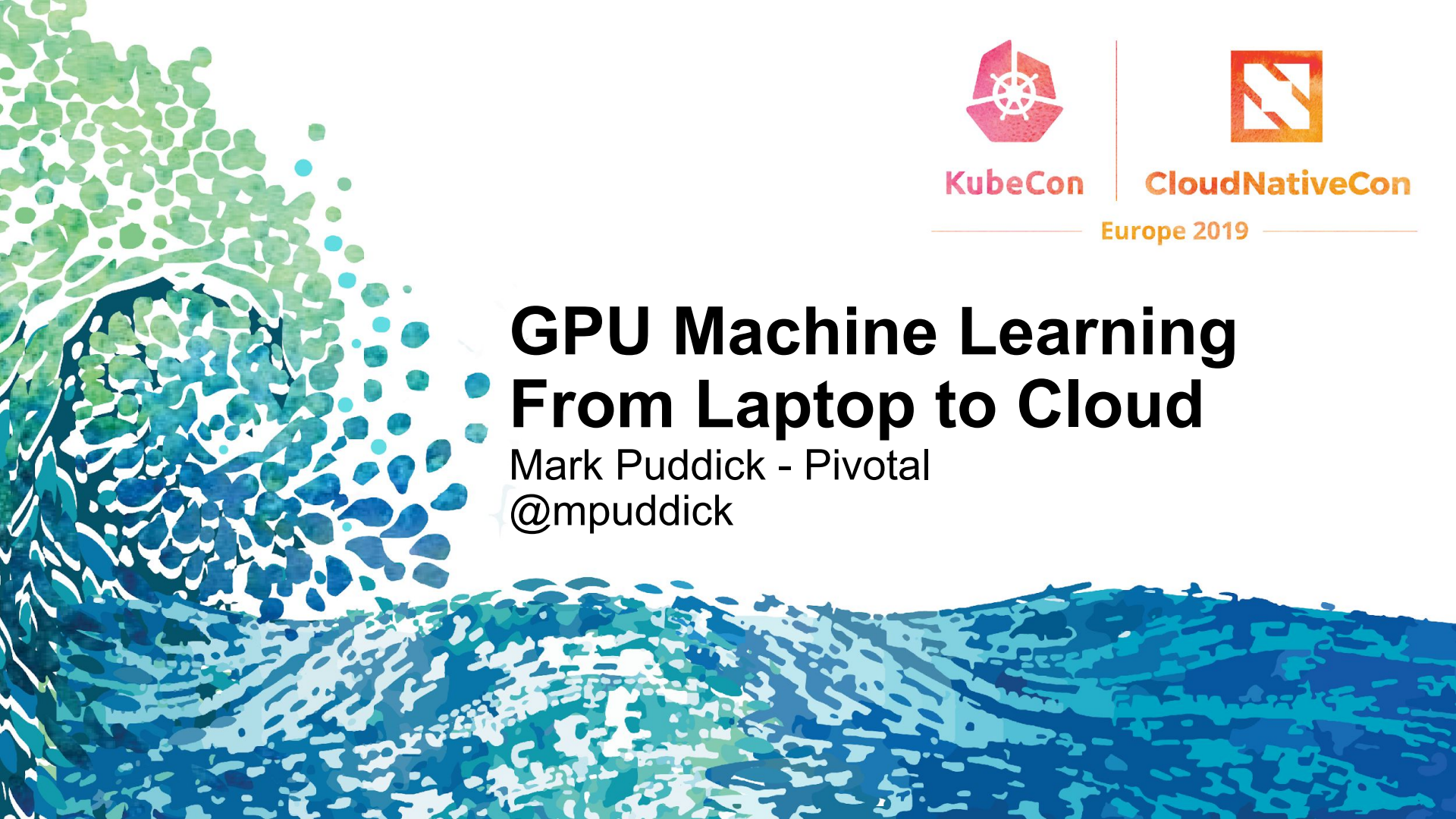

KubeCon
CloudNativeCon
Europe 2019



GPU Machine Learning From Laptop to Cloud

Mark Puddick - Pivotal
@mpuddick

- What and Why
- Running on Laptop
- Moving to cloud challenges
- Running on Cloud
- Conclusion/Question

- Get started on laptop

- Keras, Tensorflow and Python
- Dockerising you Keras app
- Minikube

- Jupyter notebooks?

Laptop - Install

- Nvidia drivers - Latest Proprietary vs Open Source
- Cuda Toolkit and cuDNN SDK - Linux packages
  - Tensorflow GPU Install guide
  - Secure Boot
- Tensorflow GPU - pip

- Versions

- Now lets run some training!

Laptop - Docker

Docerkfile

```
FROM
nvidia/cuda:10.0-cudnn7-devel-ubuntu18.04

RUN apt-get update
RUN apt-get install -y python3.6  python3-pip

COPY requirements.txt /
RUN pip3 install -r /requirements.txt
```

REQUIREMENTS.txt

```
tensorflow-gpu
keras
pandas
numpy
s3fs
requests
```

Laptop - Docker

Dockerfile

```
FROM gcr.io/ml-gpu/base-gpu:v1

COPY . /app

CMD [ "python3", "/app/legos.py" ]
```

Laptop - Enabling GPU in Docker


```
docker run --runtime=nvidia --rm nvidia/cuda nvidia-smi
sudo apt-get install nvidia-docker2
```

/etc/docker/daemon.json

```
{
    "runtimes": {
        "nvidia": {
            "path": "nvidia-container-runtime",
            "runtimeArgs": []
        }
    },
    "default-runtime" :"nvidia"
}
```

Laptop - Enabling GPU in Docker

- Mount local directory as volume

```
docker run --runtime=nvidia  --mount
type=bind,source="/home/markpudd/mldemo/data/testml",target=/app/data testml
```

Laptop - Enabling GPU in Minikube

- Driver kvm2 or None
- Use none to use underlying docker

```
minikube start --vm-driver=none --apiserver-ips 127.0.0.1 --apiserver-name
localhost

kubectl create -f
https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/v1.10/nvidia-dev
ice-plugin.yml
```

Laptop - Enabling GPU in Minikube


```
apiVersion: batch/v1
kind: Job
metadata:
  name: ml-job
spec:
  template:
    metadata:
      labels:
        app: testml
    spec:
      containers:
      - name: testml
        image: testml
        resources:
          limits:
            nvidia.com/gpu: 1
        imagePullPolicy: Never
        volumeMounts:
        - mountPath: /app/data
          name: test-volume
      restartPolicy: Never
```

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0001
spec:
  accessModes:
    - ReadWriteOnce
  capacity:
    storage: 5Gi
  hostPath:
    path: /data/pv0001/
```

Running on Laptop

- Close browser windows
- nvidia -smi
- Decrease Batch size so we get a bit more memory

Moving to cloud challenges

So now we can just drop our container onto our Kubernetes cluster and run it?

Moving to cloud challenges

Not quite….

So all of the the previous demo loads from disk and send to stdout and displays graph using pyplot……

Need to be a bit more cloud native

And it would be nice if we could spin up multiple instance with different hyper parameters

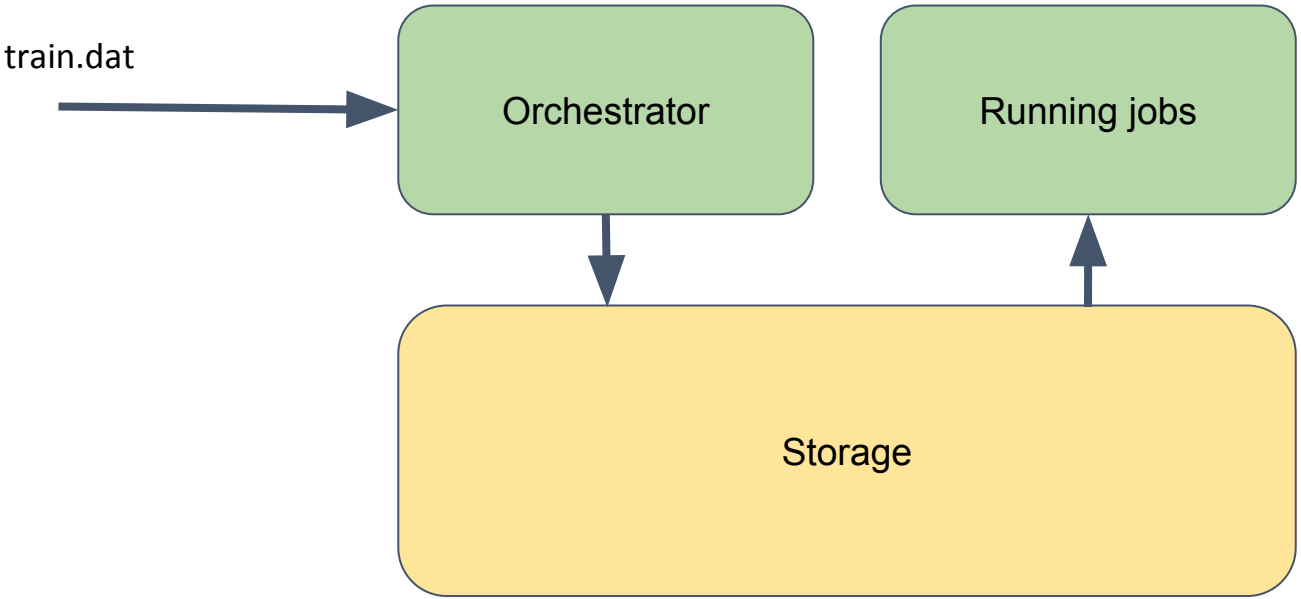
Where do we get our data?

- Object Store
- Database
- Disk

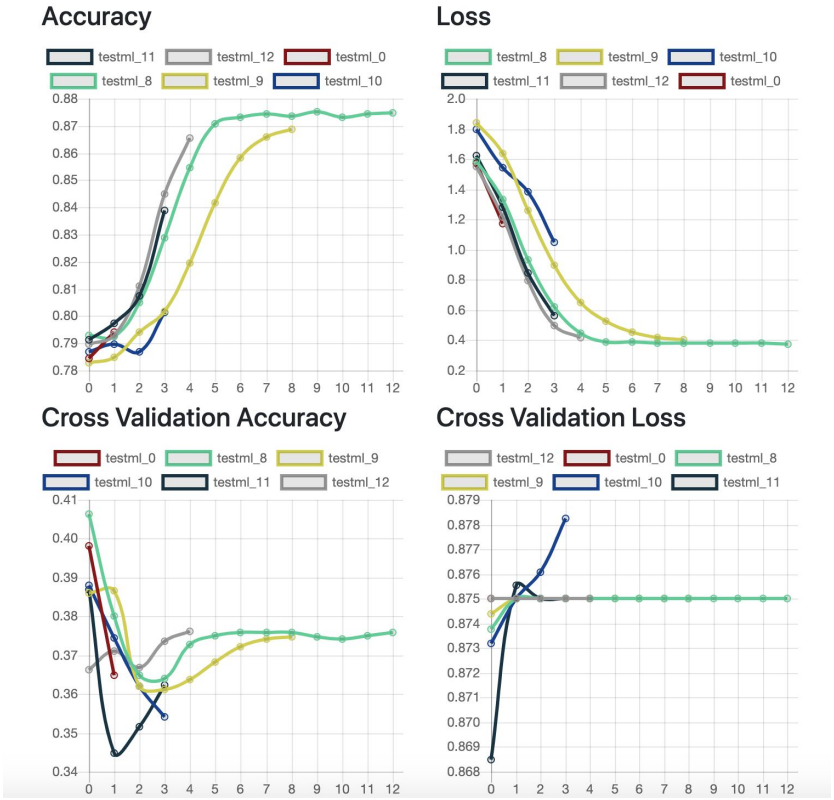
Where do we get our data?

- PersistentVolume
- One pod is going to copy data and effectively cache
- Other pods will load data from claim, so we don’t need to change our application
- Other option is pods just pull the data when they need it….

Flow



Moving to cloud challenges



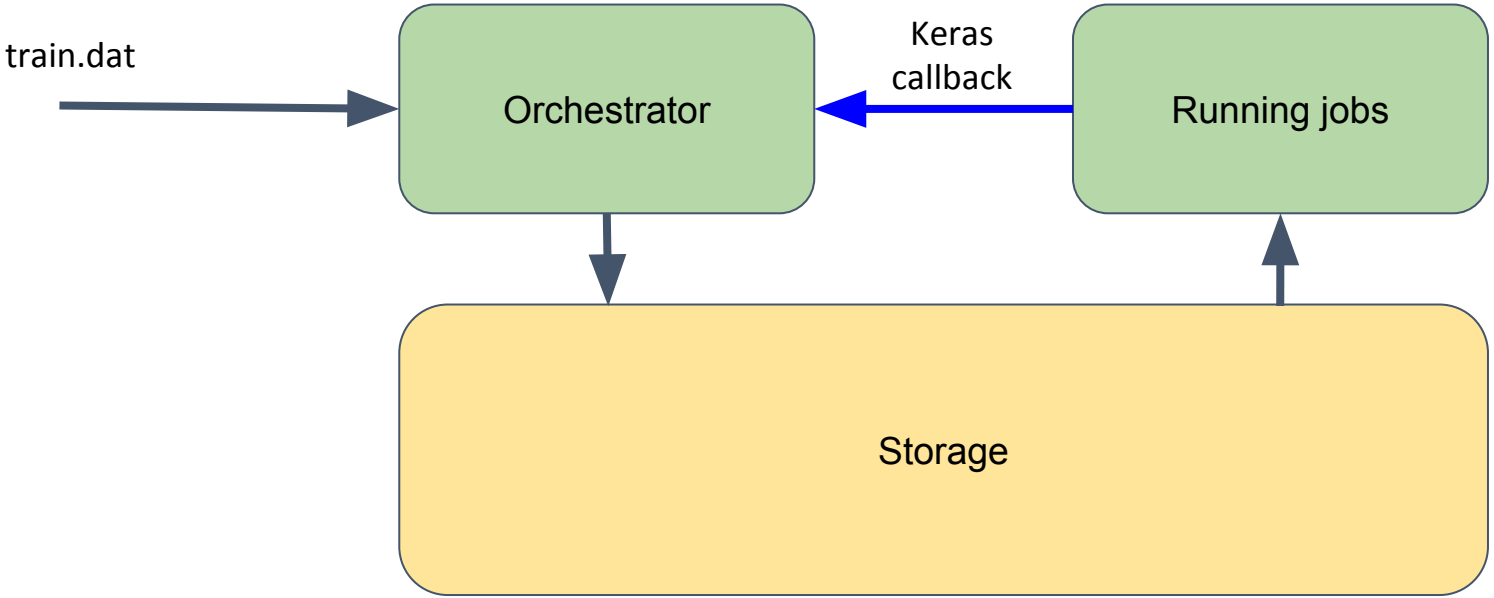
Moving to cloud challenges

Lets use a keras callback to send the data some were….

```
rm = callbacks.RemoteMonitor(root=server_root, path=endpoint_path,
field='data', headers=None, send_as_json=True)

tm =model.fit(X_train, Y_train, epochs = 75, batch_size = 64,
validation_data=(X_test, Y_test), callbacks=[rm])
```

Flow


train.dat
Orchestrator
Keras
callback
Running jobs
Storage

Where do we get our data?

- Job changes
  - Image
  - imagePullPolicy
- Still works on minikube

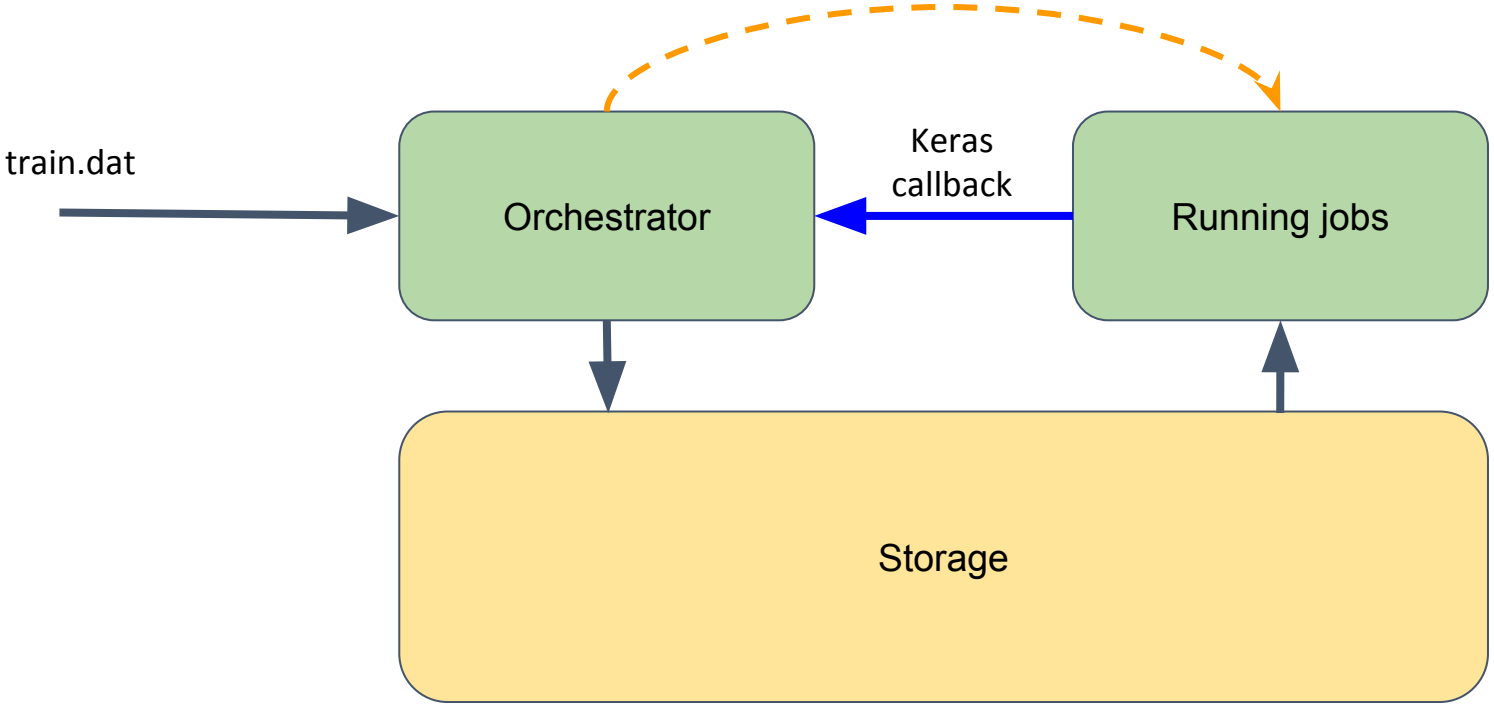
```
apiVersion: batch/v1
kind: Job
metadata:
  name: ml-job
spec:
  template:
    metadata:
      labels:
        app: testml
    spec:
      containers:
      - name: testml
        image: gcr.io/karttech/testml:v1
        resources:
          limits:
            nvidia.com/gpu: 1
        imagePullPolicy: Never
        volumeMounts:
        - mountPath: /app/data
          name: test-volume
      restartPolicy: Never
```

Moving to cloud challenges

- Tuning hyperparameters
- Hyperparameters set as environment

```
apiVersion: batch/v1
kind: Job
metadata:
  name: ml-job
spec:
  template:
    metadata:
      labels:
        app: testml
    spec:
      containers:
      - name: testml
        image: gcr.io/karttech/testml:v1
        env:
        - name: LEARNING_RATE
          value: "0.00008"
        resources:
          limits:
            nvidia.com/gpu: 1
        imagePullPolicy: Never
        volumeMounts:
        - mountPath: /app/data
          name: test-volume
      restartPolicy: Never
```

Flow

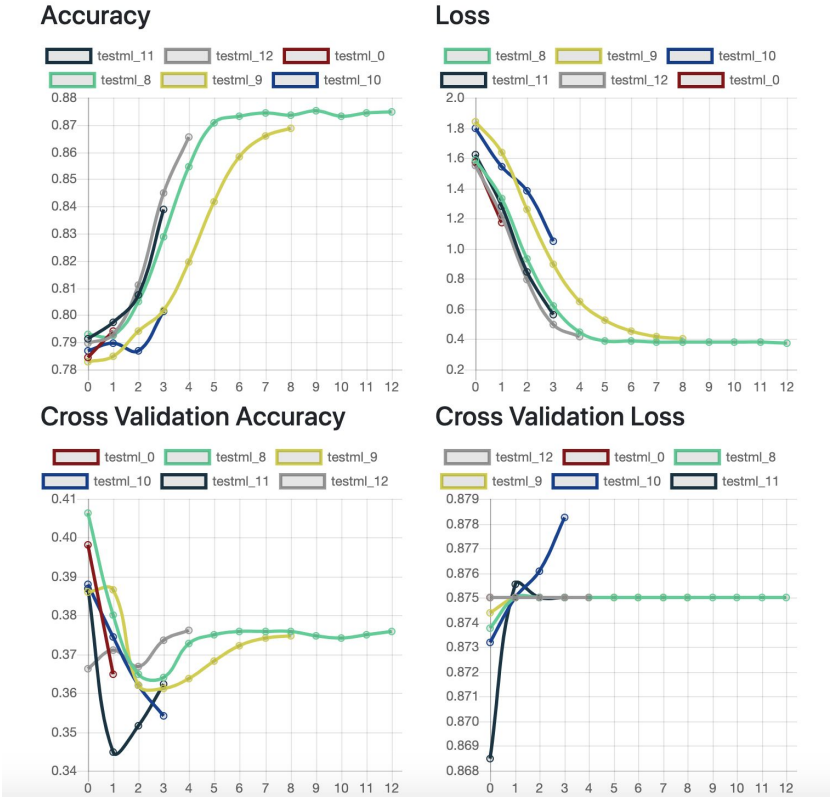

train.dat
Orchestrator
Keras
callback
Running jobs
Storage

Running on Cloud

- Running multiple versions of the
- We’ve done a few tricks in the go app so we’re only running one GPU pod per node

Running on Cloud

- Here is our result!
- Weights stored on disk



Conclusion

- Let point at some bigger data set
- Running parallel TF jobs
- Is it worth containerisation?

- Github - github.com/markpudd/mlorc

Thank you!

Mark Puddick - Pivotal
@mpuddick