Machine Learning with Kubernetes

Yaniv Donenfeld
AI/ML Solutions
Container Services, AWS

Jiaxin Shan
Software Development Engineer
Container Services, AWS
“Cloud has removed so many of the barriers to experimenting and innovating with AI that even risk-adverse businesses are making it part of their strategies.”

- Yaniv Donenfeld, just now.

40% of digital transformation initiatives supported by AI in 2019 —IDC 2018
Our mission at AWS

Put machine learning in the hands of every developer
The AWS ML Stack

Broadest and deepest set of capabilities

## AI Services

<table>
<thead>
<tr>
<th>VISION</th>
<th>SPEECH</th>
<th>LANGUAGE</th>
<th>CHATBOTS</th>
<th>FORECASTING</th>
<th>RECOMMENDATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>REKOGNITION IMAGE</td>
<td>REKOGNITION VIDEO</td>
<td>TEXITRACT</td>
<td>POLLY</td>
<td>TRANSCRIBE</td>
<td>TRANSLATE &amp; COMPREHEND MEDICAL</td>
</tr>
</tbody>
</table>

## ML Services

- **Amazon SageMaker**
  - Ground Truth
  - Notebooks
  - Algorithms + Marketplace
  - Reinforcement Learning
  - Training
  - Optimization
  - Deployment
  - Hosting

## ML Frameworks + Infrastructure

### FRAMEWORKS
- TensorFlow
- mxnet
- PYTORCH

### INTERFACES
- GLUON
- Keras

### INFRASTRUCTURE
- EC2 P3 & P3DN
- EC2 G4 & G4Cs
- FPGA
- DL CONTAINERS & AMIs
- ELASTIC CONTAINER SERVICE
- ELASTIC KUBERNETES SERVICE
- GREENGRASS
- ELASTIC INFERENCE
- INFERENTIA
Why Machine Learning on Kubernetes?

Composability

Portability

Scalability

http://www.shutterstock.com/gallery-635827p1.html
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
Use Case #1: Large Scale ML
Autonomous Vehicles Workloads
Typical Autonomous Vehicle Development Workflow

1. Data Acquisition
2. Data Ingestion
3. Data Pre-processing
4. Labeling
5. Model Training
6. Model Simulation (SIL/HIL)
7. Evaluation & Validation
8. Model Deployment and CI/CD
Typical Autonomous Vehicle Development Workflow

1. Data Acquisition
2. Data Ingestion
3. Data Pre-processing
4. Labeling
5. Model Training
6. Model Simulation (SIL/HIL)
7. Evaluation & Validation
8. Model Deployment and CI/CD
• Distributed Training Challenges

• Single GPU code ➔ multiple

• Dataset Copying time

• Dataset Sharing and Reuse

Horovod + MPIJob (or TFJob)

Use FSx Lustre / EFS

Built-in CSI driver with S3 integration
Want to Run Distributed Training on EKS?

Distributed TensorFlow training using Kubeflow on Amazon EKS

Ajay Vohra
Principal SA - Vision/AI/ML
Typical Autonomous Vehicle Development Workflow

1. Data Acquisition
2. Data Ingestion
3. Data Pre-processing
4. Labeling
5. Model Training
6. Model Simulation (SIL/HIL)
7. Evaluation & Validation
8. Model Deployment and CI/CD
Can you run my workload?
Simulations Architecture

Availability Zone 1
- VPC
- AWS Batch
- Compute Environment
- Spot instance
- Amazon FSx for Lustre

Availability Zone 2
- Spot instance
- Compute Environment
- Spot instance
- Amazon FSx for Lustre

Amazon Simple Storage Service (S3)
Simulations Architecture

Availability Zone 1
- VPC
- Amazon EKS
  - Compute Environment
    - Spot instance
- Amazon FSx for Lustre

Availability Zone 2
- Amazon EKS
  - Compute Environment
    - Spot instance
- Spot instance
- Amazon FSx for Lustre

Amazon Simple Storage Service (S3)
TOP500 – Top 10 Supercomputers in June 2019

<table>
<thead>
<tr>
<th>Rank / Name</th>
<th>Rmax / Rpeak (Petaflops)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Summit</td>
<td>148.600 / 200.795</td>
</tr>
<tr>
<td>2. Sierra</td>
<td>94.640 / 125.712</td>
</tr>
<tr>
<td>3. Sunway Tahihu Light</td>
<td>93.015 / 125.436</td>
</tr>
<tr>
<td>4. Tianhe-2A</td>
<td>61.445 / 100.679</td>
</tr>
<tr>
<td>5. Frontera</td>
<td>23.516 / 38.746</td>
</tr>
<tr>
<td>7. Trinity</td>
<td>20.159 / 41.461</td>
</tr>
<tr>
<td>8. AI Bridging Cloud Infrastructure</td>
<td>19.880 / 32.577</td>
</tr>
<tr>
<td>10. Lassen</td>
<td>18.200 / 23.047</td>
</tr>
</tbody>
</table>
We’re helping our customers run at Supercomputer Scale, targeting the equivalent of one of the Top 10 largest supercomputers in the world.
Use Case #2: ML Development Platform
Jupyter Notebook / JupyterHub

- Build, deploy, and train ML models
- Live code, equations, visualizations, and narrative text
- 40+ programming languages
- Sharing and collaboration

Built-in AWS CLI and ECR support

EFS for reusing training data and results
Kubeflow KFServing

• Simple and pluggable platform for ML inference
• Intuitive and consistent experience
• Serving models on arbitrary frameworks
  e.g. TensorFlow, XGBoost, SciKitLearn
• Encapsulates GPU auto-scaling, canary rollouts

Credits @ellis-bigelow (Kubeflow slack)
Kubeflow KFServing
Pluggable Interface

apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "InferenceService"
metadata:
  name: "sklearn-iris"
spec:
  default:
    sklearn:
      storageUri: "gs://kfserving-samples/models/sklearn/iris"

apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "InferenceService"
metadata:
  name: "flowers-sample"
spec:
  default:
    tensorflow:
      storageUri: "gs://kfserving-samples/models/tensorflow/flowers"

apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "KFService"
metadata:
  name: "pytorch-cifar10"
spec:
  default:
    pytorch:
      storageUri: "gs://kfserving-samples/models/pytorch/cifar10"
      modelClassName: "Net"
Kubeflow Pipelines

- A user interface (UI) for managing and tracking experiments, jobs, and runs.
- An engine for scheduling multi-step ML workflows.
- An SDK for defining and manipulating pipelines and components.
Kubeflow Pipelines Component
Creating a pipeline

```python
@dsl.pipeline(
    name='Sample Trainer',
    description=''
)

def sample_train_pipeline(...):
    create_cluster_op = CreateClusterOp('create-cluster', ...)
    analyze_op = AnalyzeOp('analyze', ...)
    transform_op = TransformOp('transform', ...)
    train_op = TrainerOp('train', ...)
    predict_op = PredictOp('predict', ...)
    confusion_matrix_op = ConfusionMatrixOp('confusion-matrix', ...)
    roc_op = RocOp('roc', ...)
    kfp.compiler.Compiler().compile(sample_train_pipeline, 'my-pipeline.zip')
```
Kubeflow 1.0 Arriving January 2020

Core CUJ

Develop (Jupyter)

Deploy (KFServing)

Build (Fairing)

Train (TFJob and PyTorch)

Kubeflow 1.0 – Main components

- Graduating 1.0
  - kfctl for deployment and upgrades
  - TFJob and PyTorch for distributed training (already 1.0)
  - Jupyter notebook controller and web app
  - Profile controller and UI for multiuser management
- Beta
  - Katib for hyper-parameter tuning
  - Fairing SDK to facilitate use of notebooks for build-train-deploy
  - Metadata SDK, UI, and backend
  - KFServing for model deployment and inference
Kubeflow 1.0 – AWS Support

• Multi user support
  • Kubeflow pipelines
  • Managed contributors
• IAM Roles for Service Accounts integration with notebooks
Want to Dive Deeper on Kubeflow?

**Now**

2:30PM  Kubeflow Workshop  (Workshop Room Harborside)

**Later**

https://eksworkshop.com/kubeflow/
Join the kubeflow#aws Slack channel!