A Day in the Life of a Data Scientist

Conquer Machine Learning Lifecycle on Kubernetes
Brian Redmond

- Cloud Architect @ Microsoft (18 years)
- Azure Global Black Belt Team
- Live in Pittsburgh, PA, USA
- Avid marathon runner and outdoors enthusiast
- World traveler
Rita Zhang

- Software engineer @ Microsoft, San Francisco
- Azure Cloud Native Compute team
- Kubernetes upstream features, Azure Kubernetes Service
OpenAI
Scaling Kubernetes to 2,500 Nodes

https://blog.openai.com/scaling-kubernetes-to-2500-nodes/
Agenda

- What is the typical ML workflow and some of their shortcomings
- Why DevOps?
- Why Containers, Kubernetes, and Helm?
- Intro to Kubeflow, Helm, Argo
- Demos
  - Image classification with Inception v3 and transfer learning
  - Automate repeatable ML experiments with containers
  - Deploy ML components to Kubernetes with Kubeflow
  - Scale and test ML experiments with Helm
  - Manage training jobs and pipelines with Argo
  - Serve trained models for inference with TF Serving
  - Rapid prototyping with self-service Jupyter notebook from JupyterHub
Simplified ML Workflow/Pipeline

- Keeping track of datasets is hard
- How to do automatic retraining when data changes?
- Storage and network bottlenecks

Model Development

- Slow sequential training
- Hard to explore hyperparameter space
- Distributed training is difficult to setup

Data Preparation

Serving

Classic App Dev Issues:
- “Works on my machine”
- Scalability
- Feedback from production
- Automation

Training
What is DevOps?

• “A cross-disciplinary community of practice dedicated to the study of building, evolving and operating rapidly-changing resilient systems at scale” (Jez Humble)

• Applying Agile practices to operations
  • Infrastructure as code
  • Ops teams embracing source control (git)
  • Automated testing
  • Repeatable/consistent
  • CI/CD

• This has worked well for App Dev. Now time for AI/ML
  • But, must ensure data scientist are not hindered by structure
Why Containers, Kubernetes & Helm?

• Container
  • Contains everything needed to run your application
  • Build once run anywhere
  • Starts in seconds: Great for scalability
  • Images are stored in a centralized place (Docker Hub, Azure Container Registry, gcr, ECR etc.)

• Container orchestration
  • Automating deployment, scaling, and management of containerized applications
  • Declarative
  • Can be a mix of GPU or CPU nodes

• Massive Scale
  • OpenAI dedicates up to 10k cores for a single experiment
  • Autoscaling capabilities: Pay for what you use, scale down when idle
  • Parallel training instead of sequential: huge time saver for large trainings
Kubeflow

- Machine Learning Toolkit for Kubernetes
  - To make ML workflows on Kubernetes simple, portable, and scalable
- Training controllers – simplify and manage the deployment of training jobs
  - TFJob – custom resource to handle drivers and config
  - Tensorflow, PyTorch, MXNet, Chainer, and more
- JupyterHub to create and manage interactive Jupyter notebooks
- Model serving – serve exported models with TF Serving or Seldon
- Additional components for storage, workflow, etc.
Artificial Intelligence solves critical life problems
Chinese star Fan Bingbing seen in disappearance

By Ben Westcott, CNN
Updated 6:28 AM ET, Wed October 17, 2018

Fan Bingbing outside the airport in Beijing.

Fan Bingbing spotted for first time in months, outside Beijing airport

October 17, 2018

Entertainment, Movies, People

By Agency
Demo: Find 范冰冰

Image classification with Inception v3 and transfer learning

- Generate dataset and labels for Fan Bingbing and not Fan Bingbing
- Using transfer learning, take a trained model Inception v3, retrain a new top layer for new classes of images

- Create end to end workflow with Argo
- Serve trained models for inference with TF Serving

- Run repeatable experiments using containers
- Visualize trainings with TensorBoard
- Scale and test experiments in parallel with Kubernetes and Helm
Demo: Run TensorFlow Training with Containers
Demo: Serving the Model with TF Serving

- Options for serving
  - Wrap model in a web framework (eg – Flask)
  - Tensorflow Serving
  - Seldon
Demo: Run TensorFlow Training with Kubeflow
Demo: Scale and Test Experiments in Parallel using Kubernetes, TFJob, and Helm

- Spin up pods for each variation of hyperparameters
- One centralized TensorBoard instance
- Autoscaling will create / remove VMs as needed to save cost
Demo:
Create End to End ML Pipelines with Argo
Demo: Rapid prototyping with self-service Jupyter notebook from JupyterHub
What’s Next?

• Pachyderm can version datasets and trigger new trainings when changes occur
  • Keeping track of datasets
  • How to do automatic retraining when data changes?
• Distributed File Systems
  • NFS
  • Storage and Network bottlenecks that slows training
  • ...

(one) Solution is Kubernetes:
• Slow Sequential Training
• Highly Scalable
• Hard to explore hyper-parameters space
• Easy to do distributed training
• Distributed training is hard to set up

But really, Data Scientists shouldn’t have to care about containers, kubernetes and all that stuff

Classic DevOps solutions:
• Containers
• CI/CD
• Autoscaling
• A/B testing and canary release of Models
• Comparing Production accuracy vs expected accuracy when possible
• Rolling-updates
• Etc.

Classic App dev. issues:
• Reproducibility (“it works on my machine”)
• Scalability
• Getting feedback from Production
Resources

- Source code for this talk:
  [https://github.com/ritazh/kubecon-ml](https://github.com/ritazh/kubecon-ml)
- Kubeflow labs for AKS:
  [https://github.com/Azure/kubeflow-labs](https://github.com/Azure/kubeflow-labs)
- Provision a Kubernetes cluster on Azure:
  [https://github.com/Azure/kubeflow-labs/tree/master/2-kubernetes#provisioning-a-kubernetes-cluster-on-azure](https://github.com/Azure/kubeflow-labs/tree/master/2-kubernetes#provisioning-a-kubernetes-cluster-on-azure)