



Kubeflow

Natural Language Code Search With Kubeflow

2018/12/12

Hamel Husain(hamelsmu@github.com) & Jeremy Lewi

(jlewi@google.com)

http://bit.ly/kubecon_code_search_2018

Agenda

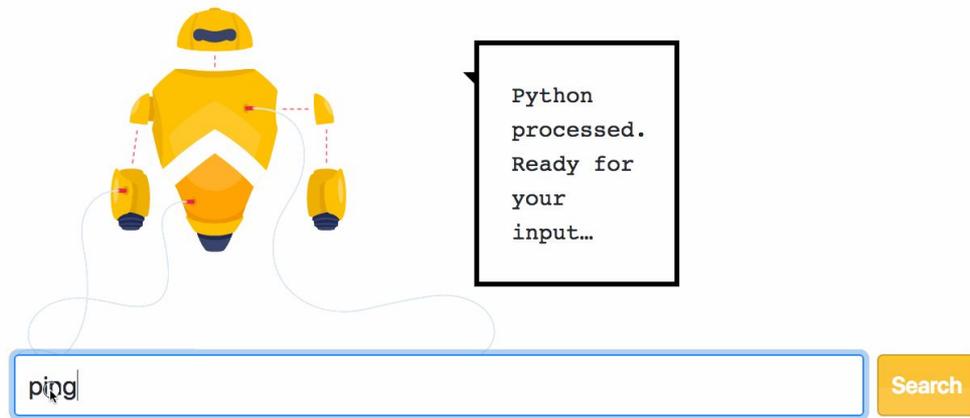
Summary: Kubeflow makes it easy to build and deploy ML products.

- Intro: Building Natural Language Code Search At GitHub
- Why Kubeflow?
 - Productionizing ML takes too long
- What is Kubeflow?
 - A Kubernetes native platform for ML
- Walk through how Kubeflow can accelerate turning experiments into production
- Demo
- Summary



Idea: Semantic Code Search

- Build a way to search code using natural language.
- Query: Natural language describing what code does
- Result: relevant code matching the query
- [Code](#) (kubeflow/examples)
- [Blog Post](#) by [Hamel Husain](#)



- <https://experiments.github.com/semantic-code-search>



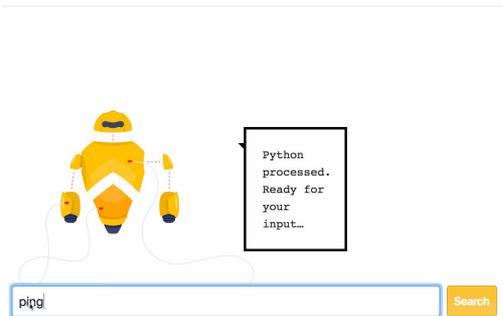
Going from an idea to a product

Prototype MVP With Demo In Jupyter Notebook: **2 Weeks**



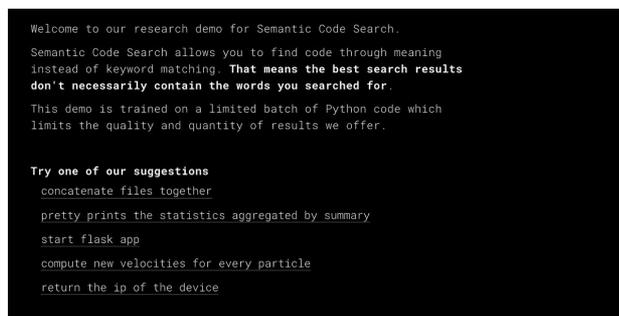
https://github.com/hamelsmu/code_search

Demo with front-end mockup with blog post: **+3 Days**



<https://towardsdatascience.com/semantic-code-search-3cd6d244a39c>

Experiments.Github.Com: **+3 Months**



<https://experiments.github.com/>

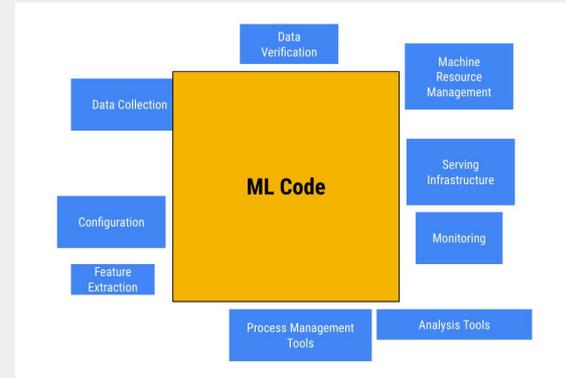


Perception: ML Products are mostly about ML

- “Someone else” will take care of production
- Most time & effort is spent building the model
- Data Science teams often given headcount and resourced that reflect this understanding of the world
- There isn't as big of a data scientist shortage as you think!



Team structure reflects perception



GitHub Circa August 2018:

- **10 Data Scientists**
- **1 Front End, 1 Data Engineer**
- No ML Infra. DIY



Datascientists

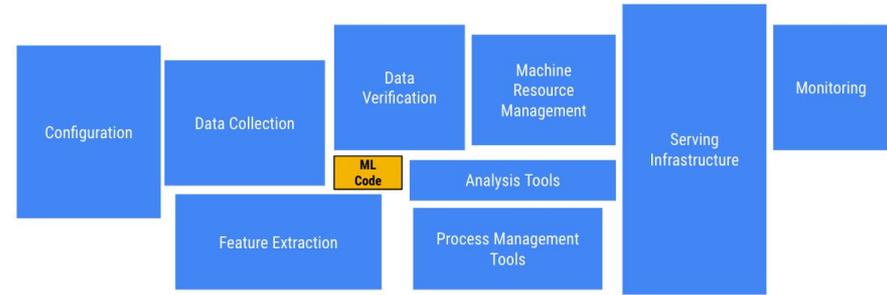
Devops

Reality:

- Infrastructure/DevOps team want to know a model is useful before investing in productinization
- Datascientists often don't know if a model is really useful without launching it
- Building the model is often the least costly step of productionization
 - 2 weeks to build model in notebook
 - 3 months to launch on experiments.com



ML Requires DevOps; lots of it



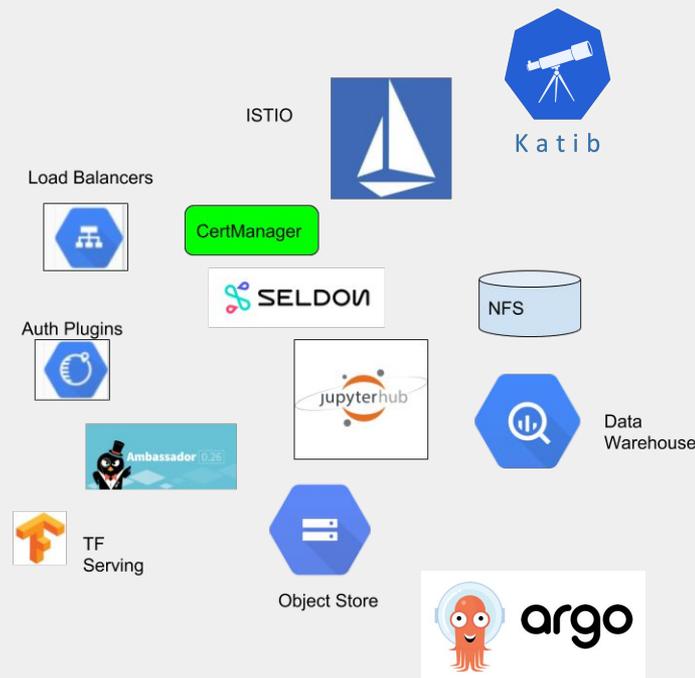
Kubeflow: A platform for building ML products

- **Leverage containers and Kubernetes to solve the challenges of building ML products**
 - Reduce the time and effort to get models launched
- **Why Kubernetes**
 - Kubernetes has won
 - Kubernetes runs everywhere
 - Enterprises can adopt shared infrastructure and patterns for ML and non ML services
 - Knowledge transfer across the organization
- **Kubeflow is open**
 - No lock in
 - 120+ Members
 - 20+ [Organizations](#)
 - Stats available @ <http://devstats.kubeflow.org>



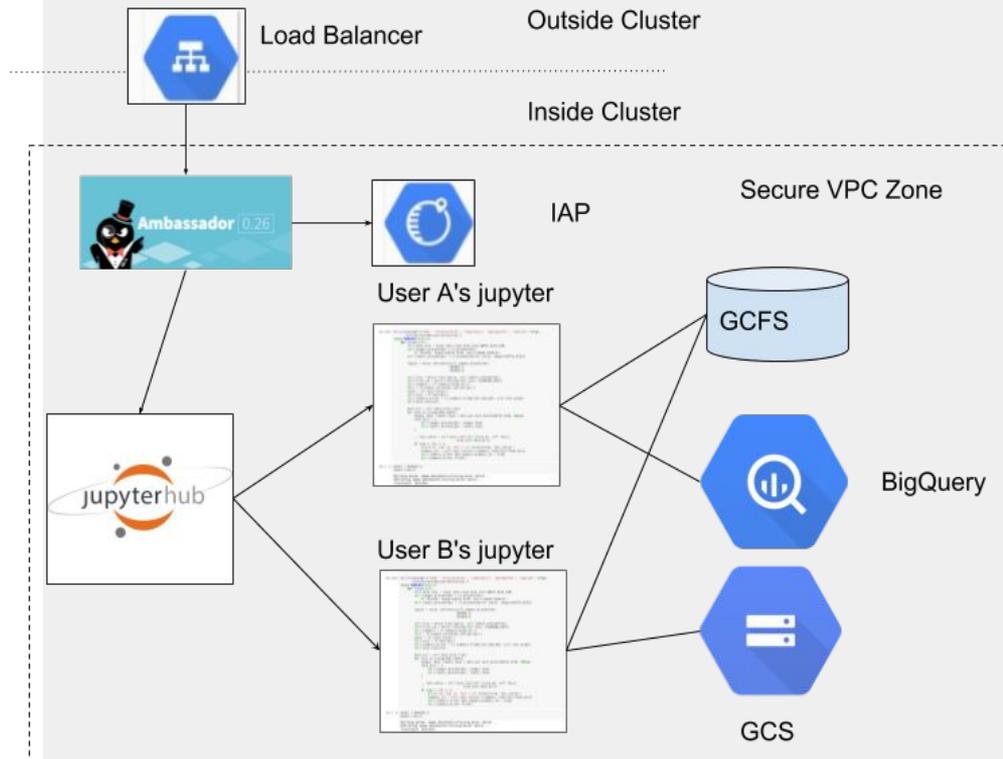
ML Components

- Goal: components for every stage of ML
- Examples:
 - Experimentation / Data Exploration
 - Jupyter/JupyterHub
 - Training
 - K8s CRDs for distributed training for PyTorch & TFJob
 - Katib - For HP Tuning
 - Inference
 - Beam transforms for batch inference
 - Workflows:
 - Pipelines
 - Feature Store
 - Feast (from GOJEK)



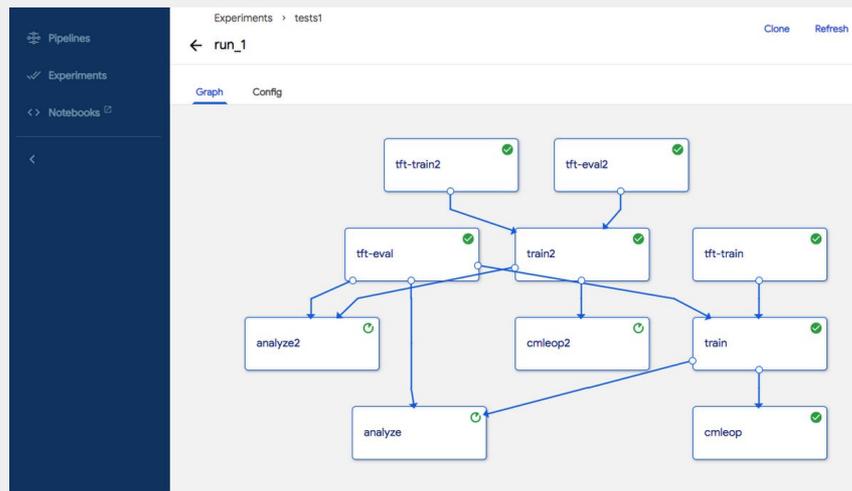
Deployment

- Make it easy to deploy components as a cohesive platform
 - CLI kfctl
 - Web UI
- Web UI currently GCP only
 - Ask your vendor to add support
 - See current members at
 - https://github.com/kubeflow/community/blob/master/member_organizations.yaml



Pipelines

- Define complex ML workflows
 - in Python
- Run pipelines regularly
- Visualize all runs and track all the results
- [Example](#)
- Powered by Argo



```
tfteval = dsl.ContainerOp(  
    name = 'tft-eval',  
    image = 'gcr.io/google-samples/taxi',  
    arguments = [ "--input_handle", ... ]  
)  
tfttrain = dsl.ContainerOp(...)  
tfttrain.after(tfteval)
```



Argo CD

- Declarative Continuous Delivery for Kubernetes
- Keeps resources in a cluster in sync with manifests in a git repository



The screenshot shows the Argo CD web interface for an application named 'code-search'. The interface includes a sidebar with navigation icons and a main content area. The top of the main area displays 'APPLICATION DETAILS' and 'code-search'. Below this, there are two summary boxes: '3 DAYS ACTIVE' and '0 DAYS SINCE LAST SYNCHRONIZED'. The main content area shows a tree view of the application's resources. The 'code-search' application is expanded, showing a 'deployment' resource for 'query-embed-server' and 'search-index-server', and 'service' resources for the same. The 'query-embed-server' deployment is in a 'containerincreasing' state, and the 'search-index-server' deployment is in a '0/1 containers ready' state. The 'query-embed-server' service is in a 'pod' state, and the 'search-index-server' service is in a 'pod' state.



Going from an idea to production

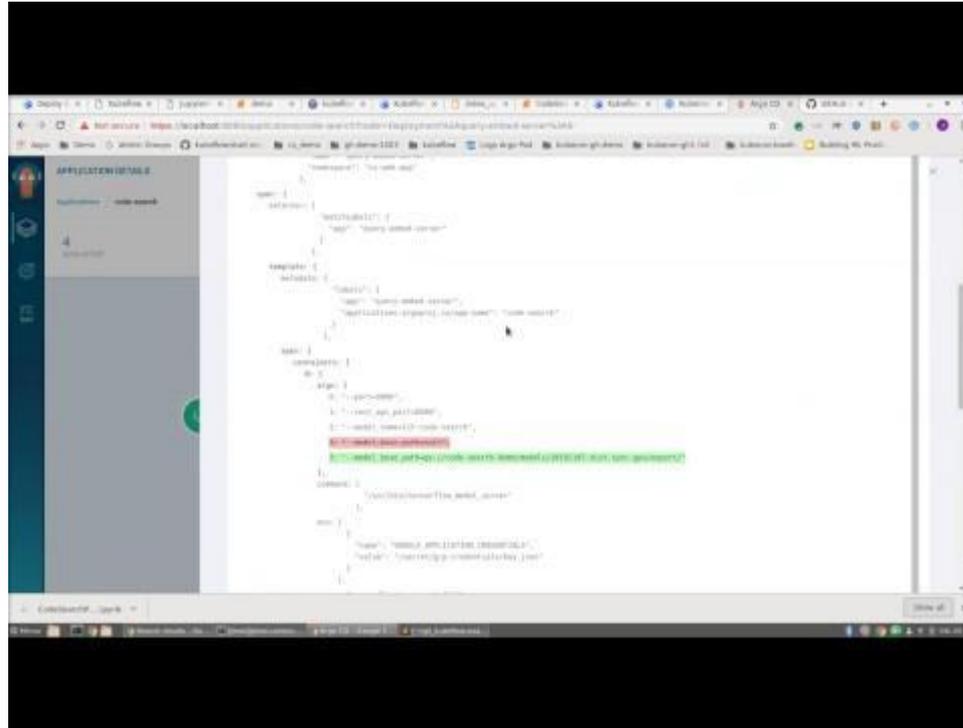
- Multiple steps involved in building the model
- Each step has different resource requirements
 - Resource management is a big problem
- Application consists of multiple microservices



Demo



Demo Video



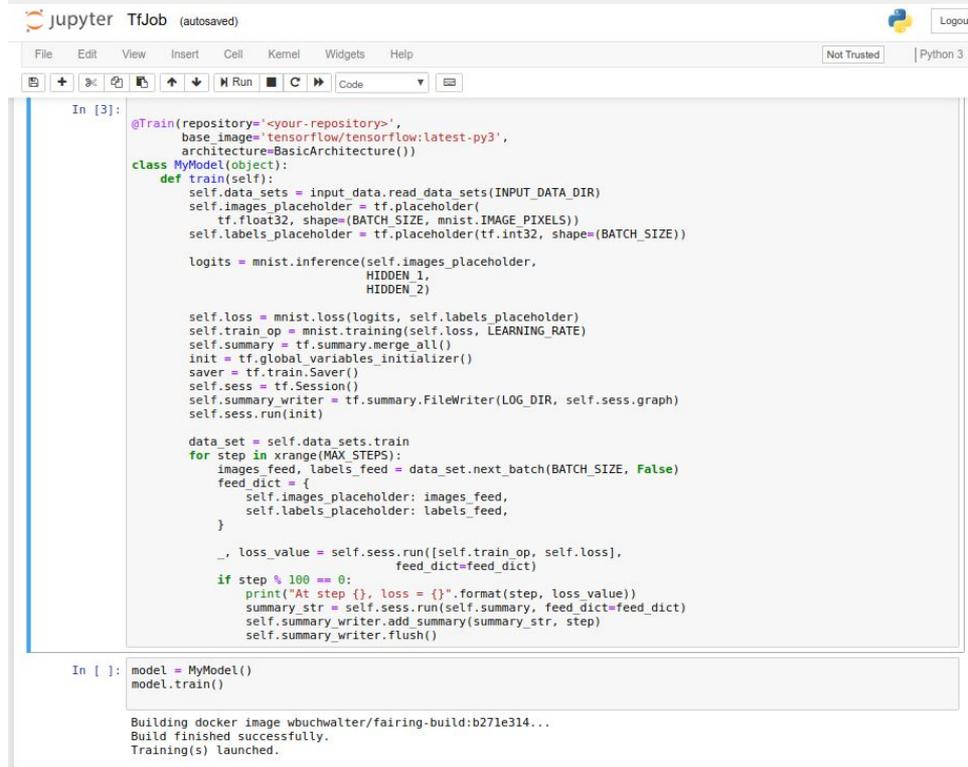
The screenshot shows a web browser window with a code editor. On the left, there is a sidebar with a tree view titled "APPLICATION STATE". The main area displays a JSON-like structure with several nested objects. One line of code is highlighted in green, indicating it is selected or active. The code appears to be a configuration or state object for an application.

```
...  
  "name": "my-app",  
  "version": "1.0.0",  
  "description": "A sample application",  
  "author": "John Doe",  
  "license": "MIT",  
  "dependencies": {  
    "react": "17.0.2",  
    "react-dom": "17.0.2",  
    "react-scripts": "5.0.1",  
    "typescript": "4.5.5",  
    "web-vitals": "2.1.4"  }  
  },  
  "devDependencies": {  
    "@testing-library/jest-dom": "5.16.5",  
    "@testing-library/react": "12.1.5",  
    "@testing-library/user-event": "13.5.0",  
    "jest": "27.5.1",  
    "react-test-renderer": "17.0.2"  }  
  },  
  "scripts": {  
    "start": "react-scripts start",  
    "build": "react-scripts build",  
    "test": "react-scripts test",  
    "eject": "react-scripts eject"  }  
  }  
}
```



Notebooks & Kubeflow

- Easily launch notebooks via UI
- Fairing is a library that makes it easy to use K8s from a notebook
 - Notebook -> container -> TFJob
- **Datascientists can leverage K8s without knowing K8s**
- Started by Microsoft
- Inspired by Lyft Learn
- Kudos Arrikto for the new JupyterHub spawner UI



```
jupyter TfJob (autosaved)
File Edit View Insert Cell Kernel Widgets Help
+ -> Run Code
In [3]:
@Train(repository='<your-repository>',
        base_image='tensorflow/tensorflow:latest-py3',
        architecture=BasicArchitecture())
class MyModel(object):
    def train(self):
        self.data_sets = input_data.read_data_sets(INPUT_DATA_DIR)
        self.images_placeholder = tf.placeholder(
            tf.float32, shape=(BATCH_SIZE, mnist.IMAGE_PIXELS))
        self.labels_placeholder = tf.placeholder(tf.int32, shape=(BATCH_SIZE))

        logits = mnist.inference(self.images_placeholder,
                                  HIDDEN_1,
                                  HIDDEN_2)

        self.loss = mnist.loss(logits, self.labels_placeholder)
        self.train_op = mnist.training(self.loss, LEARNING_RATE)
        self.summary = tf.summary.merge_all()
        init = tf.global_variables_initializer()
        saver = tf.train.Saver()
        self.sess = tf.Session()
        self.summary_writer = tf.summary.FileWriter(LOG_DIR, self.sess.graph)
        self.sess.run(init)

        data_set = self.data_sets.train
        for step in xrange(MAX_STEPS):
            images_feed, labels_feed = data_set.next_batch(BATCH_SIZE, False)
            feed_dict = {
                self.images_placeholder: images_feed,
                self.labels_placeholder: labels_feed,
            }

            _, loss_value = self.sess.run([self.train_op, self.loss],
                                          feed_dict=feed_dict)

            if step % 100 == 0:
                print("At step {}, loss = {}".format(step, loss_value))
                summary_str = self.sess.run(self.summary, feed_dict=feed_dict)
                self.summary_writer.add_summary(summary_str, step)
                self.summary_writer.flush()

In [ ]: model = MyModel()
        model.train()

Building docker image wbuchwalter/fairing-build:b271e314...
Build finished successfully.
Training(s) launched.
```

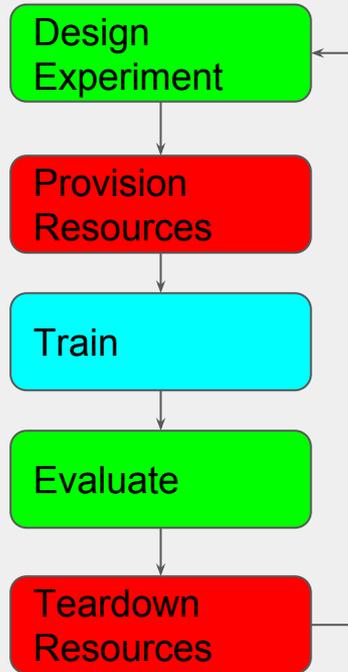


Training

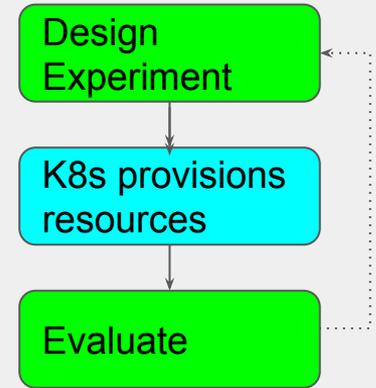
- Scale horizontally
 - Multiple experiments
 - Different vocab sizes, loss functions
 - TFJob for Distributed Training
- Scale vertically
 - Multiple GPUs (K80, P100, V100, T4)
- Multiple users
- **Kubernetes takes care of resource management so datascientists can focus on experiment design & analysis**



Without
Kubernetes

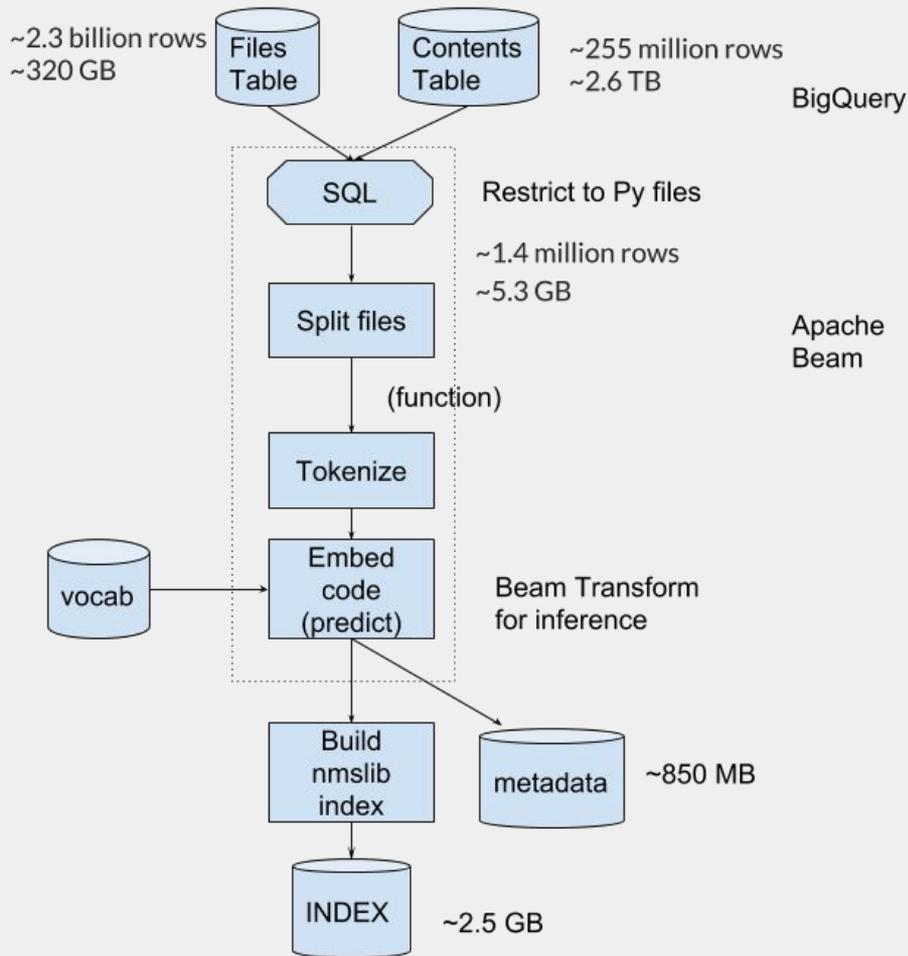


With Kubernetes



Offline Inference

- Embeddings for all python functions (possible search results) are precomputed
- Embeddings are indexed using nmslib to enable fast lookup in response to a query
- Scale out & up
 - Beam Job = 347 vCPU hr
 - Build nmslib index ~28 GB Ram



Online Inference

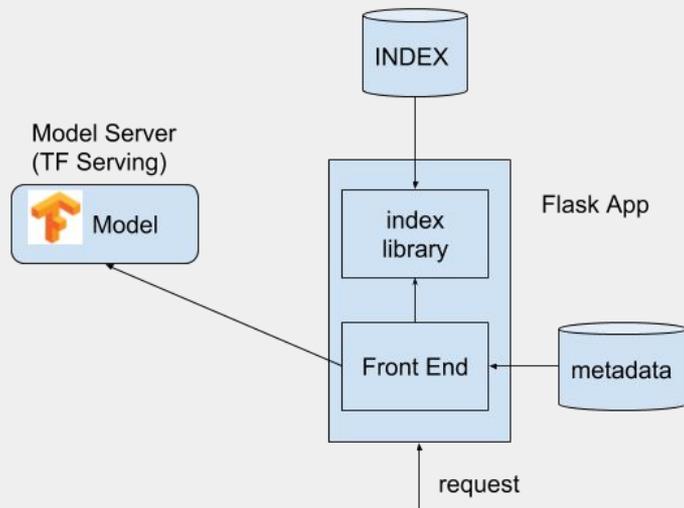
- TFServing to compute query embeddings
- Index server provides fast lookup
- Launching a public experiment is costly
 - ~ 3 months
 - No devops support
 - Security concerns
- So much room for improvement!
 - GitHub building our Kubeflow infra.



```
Welcome to our research demo for Semantic Code Search.  
Semantic Code Search allows you to find code through meaning  
instead of keyword matching. That means the best search results  
don't necessarily contain the words you searched for.  
This demo is trained on a limited batch of Python code which  
limits the quality and quantity of results we offer.
```

```
Try one of our suggestions  
concatenate files together  
pretty prints the statistics aggregated by summary  
start flask app  
compute new velocities for every particle  
return the ip of the device
```

<https://experiments.github.com/semantic-code-search>

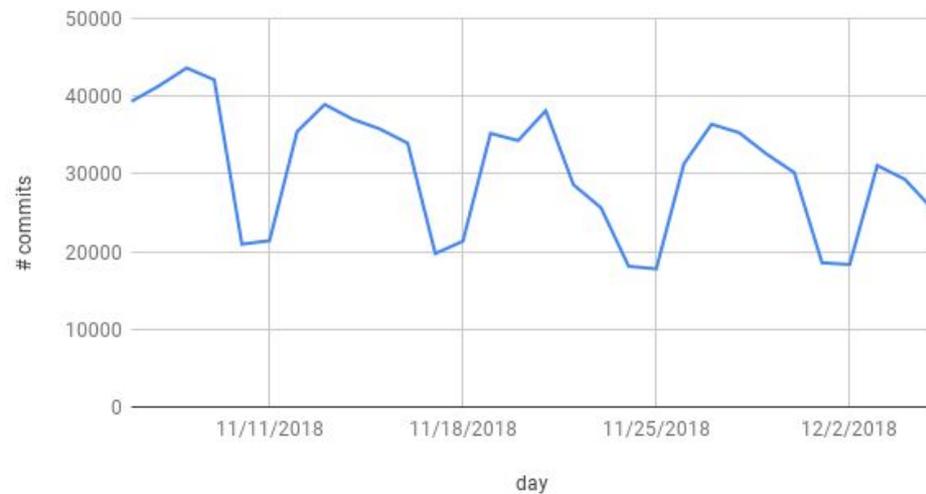


Keeping the index fresh

- Code is constantly changing in GitHub
- If we don't update the index frequently quality of the results will deteriorate
- Common problem in ML products



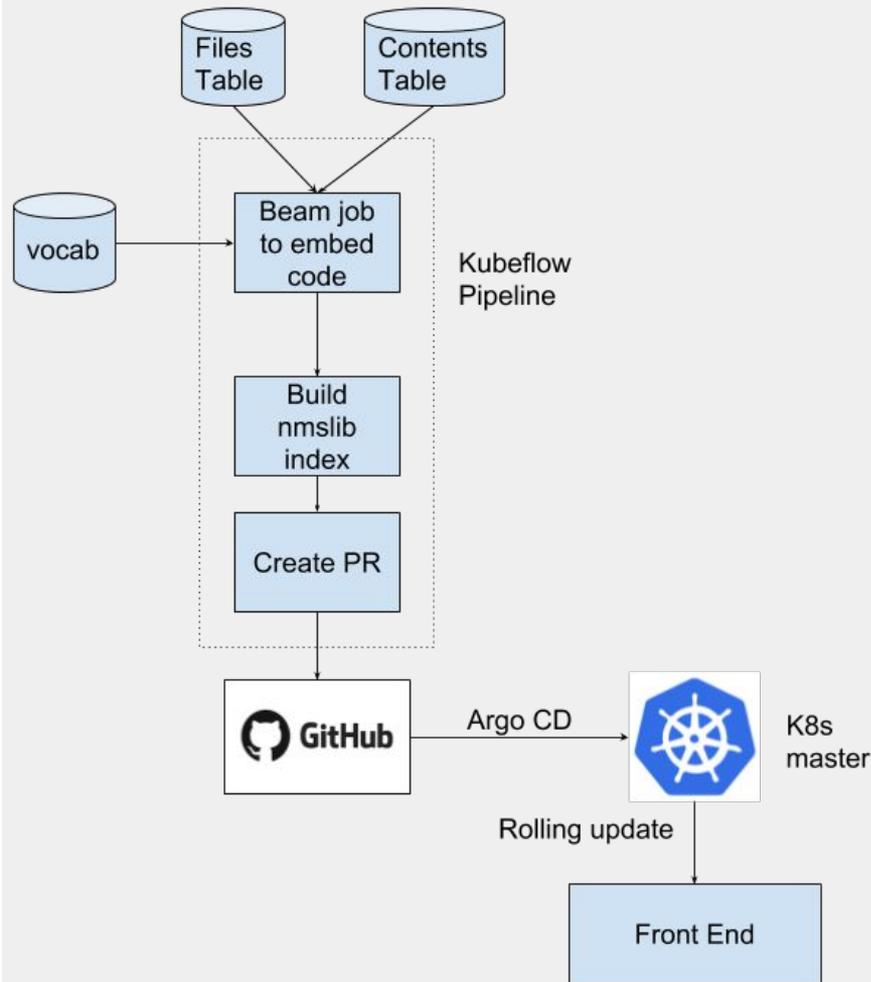
GitHub commits per day



From BigQuery table
`bigquery-public-data.github_repos.commits`

Updating the Index

- Use Kubeflow pipelines to periodically run the steps to compute the index
- Pipeline creates a PR updating the index served by the front end
- Argo CD synchronizes the deployed infrastructure
- Pipeline code:
https://github.com/kubeflow/examples/tree/master/code_search/pipeline

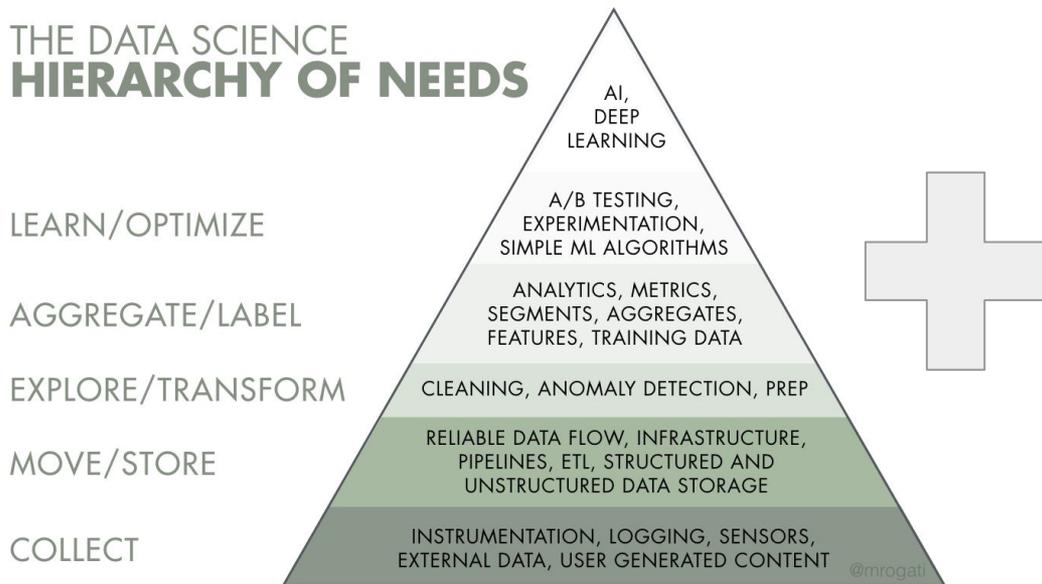


State of ML in Industry

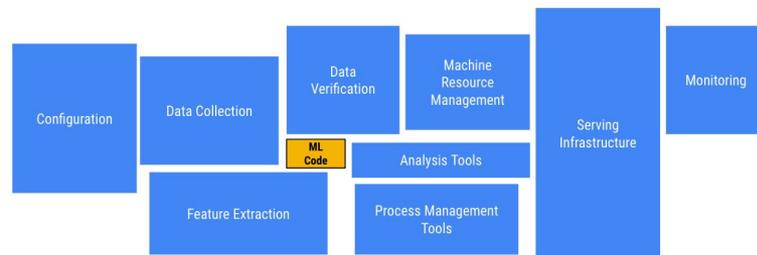


You Probably Don't Need To Hire Another ML-Data Scientist

THE DATA SCIENCE HIERARCHY OF NEEDS



Source: [Monica Rogatti's Hierarchy of Needs](#)



Source: [Sculley et al.: Hidden Technical Debt in Machine Learning Systems](#)



Special Thanks

Sanyam Kapoor - Google Intern Ported model to Kubeflow

Yang Pan - Googler who build the pipeline

Chris Beitel - Director ML Research UCSF - Helped with training training the model

Lukasz Kaiser and Ryan Sepassi - Googlers behind Tensor2Tensor; helped with model design with T2T

Jesse Suen and Danny Thomson - Intuit engineers working on Argo CD and integration with Kubeflow



ML and Kubeflow Related Talks

12/11 11:40 - 12:15 [Using Kubernetes to Offer Scalable Deep Learning on Alibaba Cloud](#) Kai Zhang & Yang Che

12/11 13:45 - 15:10 [Tutorial: Kubeflow End-to-End: GitHub Issue Summarization](#) chasm@ amyruh@

12/11 15:40 - 16:15 [Machine Learning as Code: and Kubernetes with Kubeflow](#) jaysmith@ aronchick@

12/11 16:30 - 17:05 [Why Data Scientists Love Kubernetes](#) Sophie Watson & William Benton

12/12 10:50 - 11:25 [Natural Language Code Search for GitHub Using Kubeflow](#) jlewi@ & Hamel Husein

12/12 11:40 - 12:15 [Nezha: A Kubernetes Native Big Data Accelerator For Machine Learning](#) Huamin Chen & Yuan Zhou

12/12 14:35 - 15:10 [Eco-Friendly ML: How the Kubeflow Ecosystem Bootstrapped Itself](#) Peter MacKinnon

12/12 15:40 - 16:15 [Deep Dive: Kubeflow BoF](#)



Kubeflow Related Booths

- Agile Stacks (Booth S/E 22)
 - See a demo of automated deployment of Kubeflow and ML pipelines on AWS and on-prem bare metal, tightly integrated with infrastructure services for scheduling, monitoring, logging, data management, storage, and security.
- Arrikto (Booth S/E 43)
 - Come see a multi-cloud ML workflow (ingress of multi-GB data, pre-processing, distributed training, and inference in distinct locations) with Kubeflow + Arrikto!
- One Convergence (Booth S/E49)
 - OnPrem Deep Learning as a Service



More Info

- Kubeflow Docs - <https://www.kubeflow.org/>
- Code - https://github.com/kubeflow/examples/tree/master/code_search
- GitHub Experiments - <https://experiments.github.com/>
- Argo CD- <https://github.com/argoproj/argo-cd>
- Tensor2Tensor - <https://github.com/tensorflow/tensor2tensor>





Kubeflow

Thank You

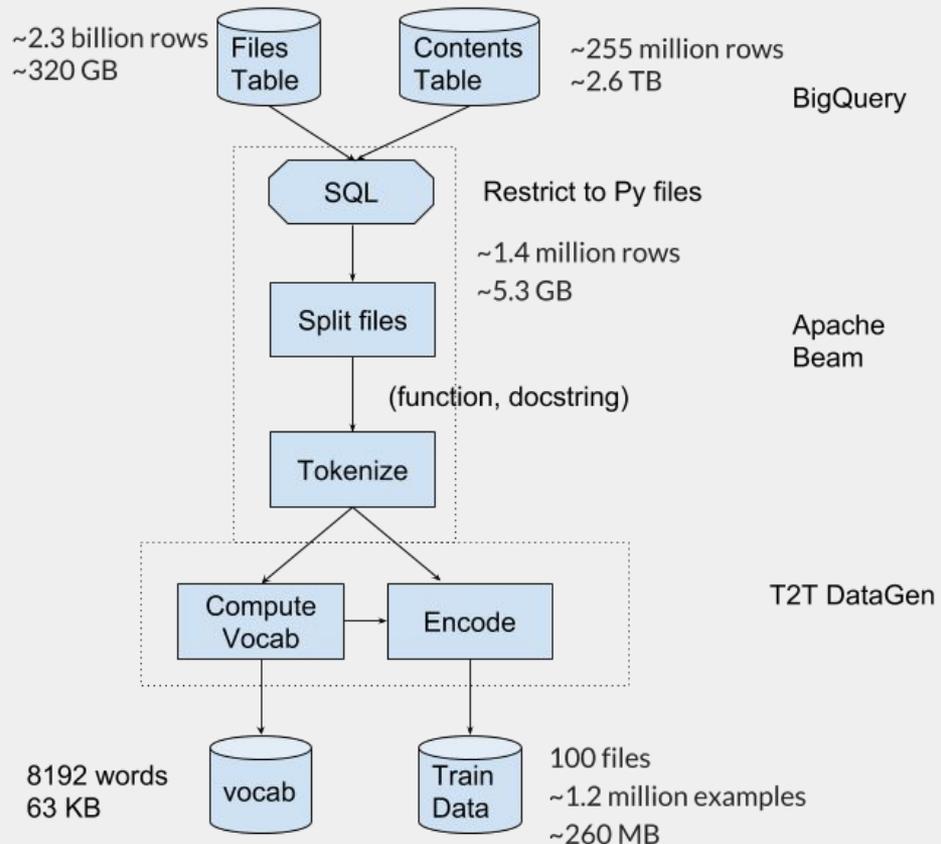


Appendix: Extra Slides



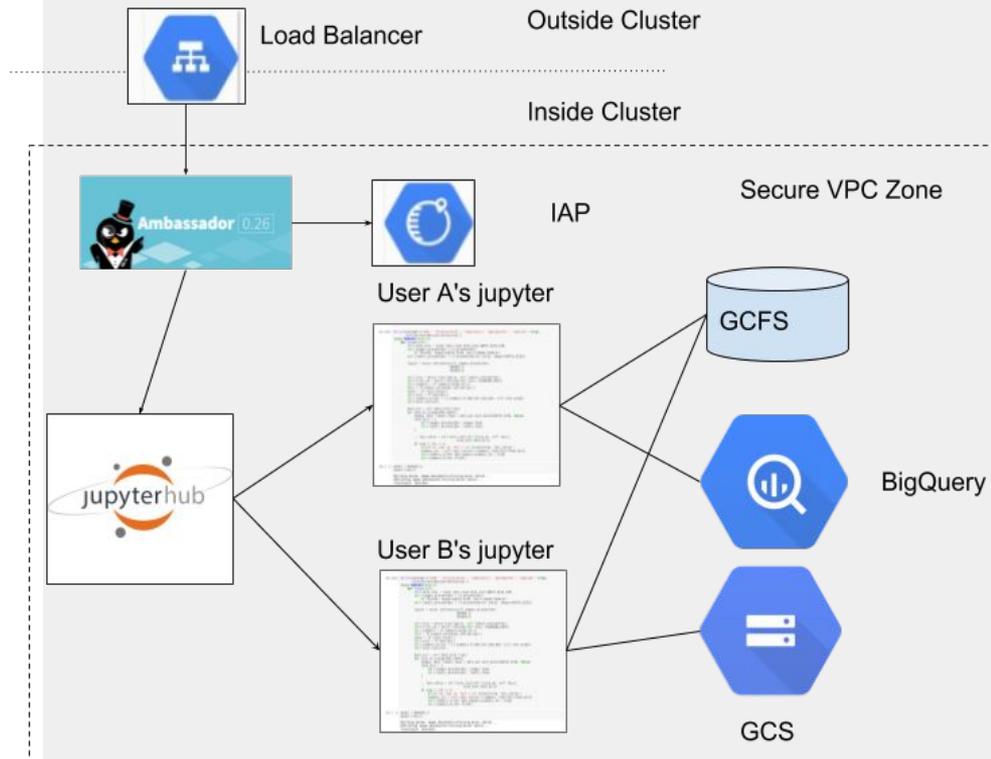
Data Preparation

- 3 Different compute engines involved
 - SQL(BigQuery)
 - Beam(Dataflow)
 - T2T Binary (K8s Job)
- **Beam Job ~108 vCPU hours**
 - Need to scale vertically and/or horizontally
 - Kubernetes enables both



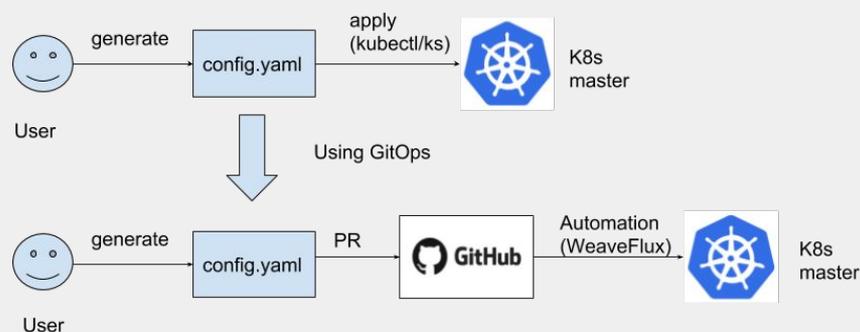
Deploying notebooks

- Easily launch notebooks
- Access notebooks securely:
<https://kubeflow.acme.com/jupyter>
- Shared storage (NFS) for collaboration
- Connect to data warehouse
- Enforce enterprise security policies



Enable GitOps for ML

- Fully declarative
- kubectl is a two step process
 - Create configs
 - Apply configs
- GitOps reduces the toil of managing infrastructure
 - Automation (e.g. WeaveFlux) keeps infrastructure up to date
 - Allows for automatic policy enforcement

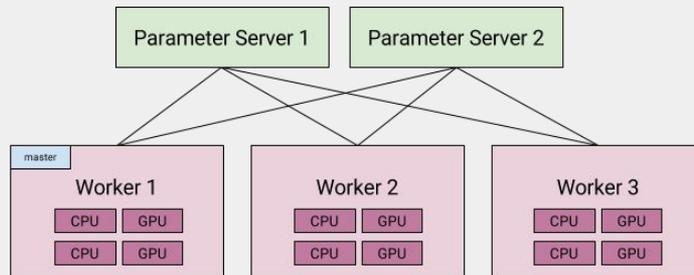


TFJob

- Integrates TF distributed training and estimator API with K8s
- Use K8s to scale out training and leverage accelerators
- TF specific controller takes care of managing all the K8s resources
 - K8s services
 - Pods
- Users benefit from K8s toolchain
 - kubectl for CLI
 - K8s dashboard for monitoring



```
apiVersion: kubeflow.org/v1alpha2
kind: TFJob
metadata:
  name: tf-job-simple
  namespace: kubeflow
spec:
  tfReplicaSpecs:
    Workers:
      replicas: 3
      template:
        spec:
          containers:
            - image: acme/myjob
```

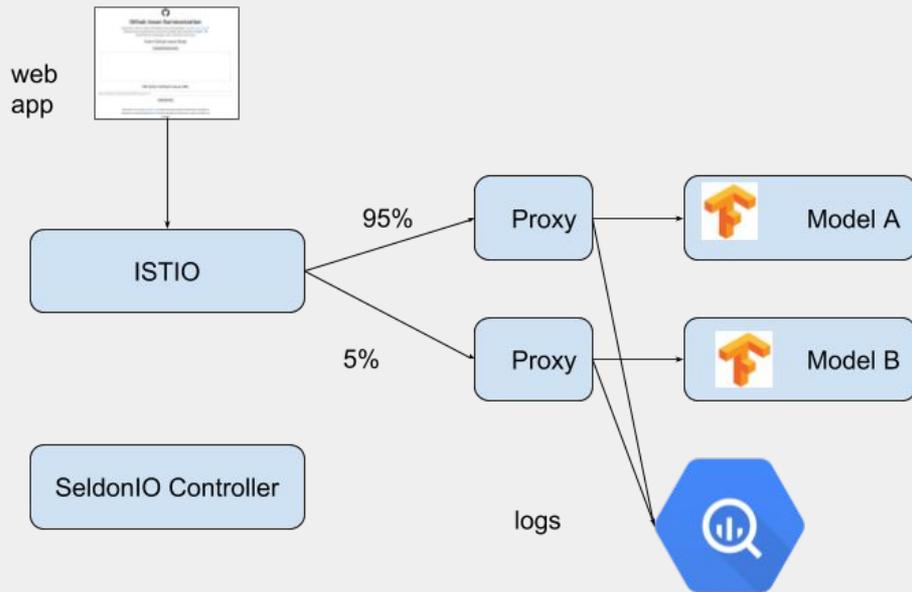


TF Serving

- Collaborating with TF to create a K8s native story for TFServing
- Adding prometheus exporter for metrics
- ISTIO for telemetry and traffic splitting
- **Opportunity to leverage K8s to simplify pushing models**
 - Need to measure model quality
 - Global rollout needed to uniformly sample traffic across



model push \neq binary push



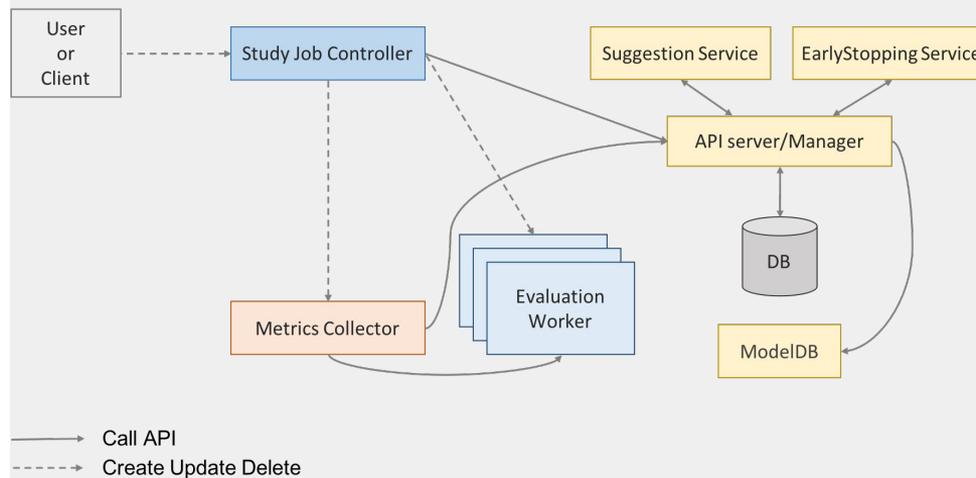
GPU Serving

- NVIDIA Inference Server - Optimized for GPUs
- Using TF Serving



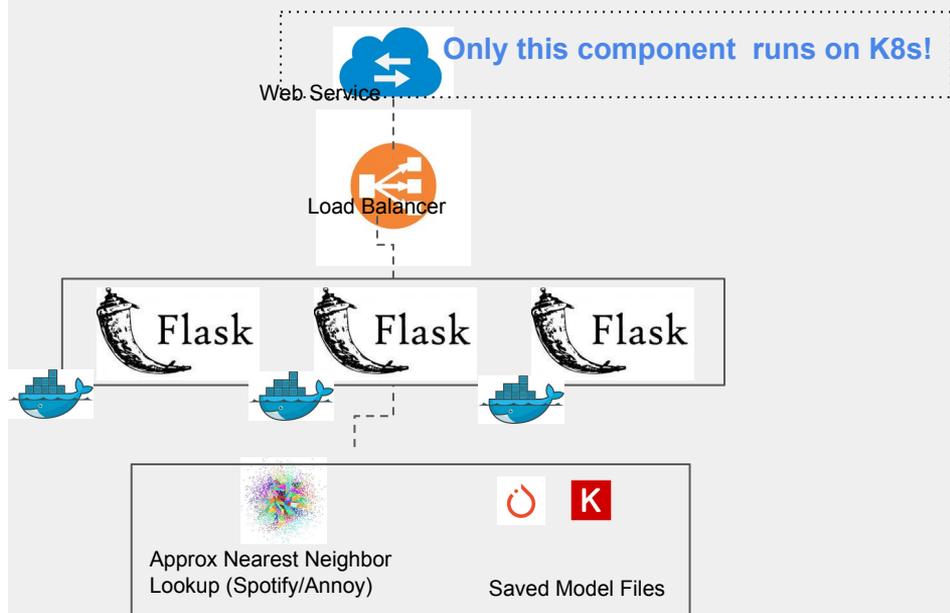
Katib(HP Tuner)

- Pluggable microservice architecture for HP tuning
 - Different optimization algorithms
 - Different frameworks
- StudyJob (K8s CR) ([example](#))
 - Hides complexity from user
 - No code needed to do HP tuning



Experiments.GitHub.com

- Launching a public experiment is costly
 - No devops support
 - Security concerns
- So much room for improvement!
 - Currently building our Kubeflow infra.
- ~ 3 months



```
Welcome to our research demo for Semantic Code Search.

Semantic Code Search allows you to find code through meaning
instead of keyword matching. That means the best search results
don't necessarily contain the words you searched for.

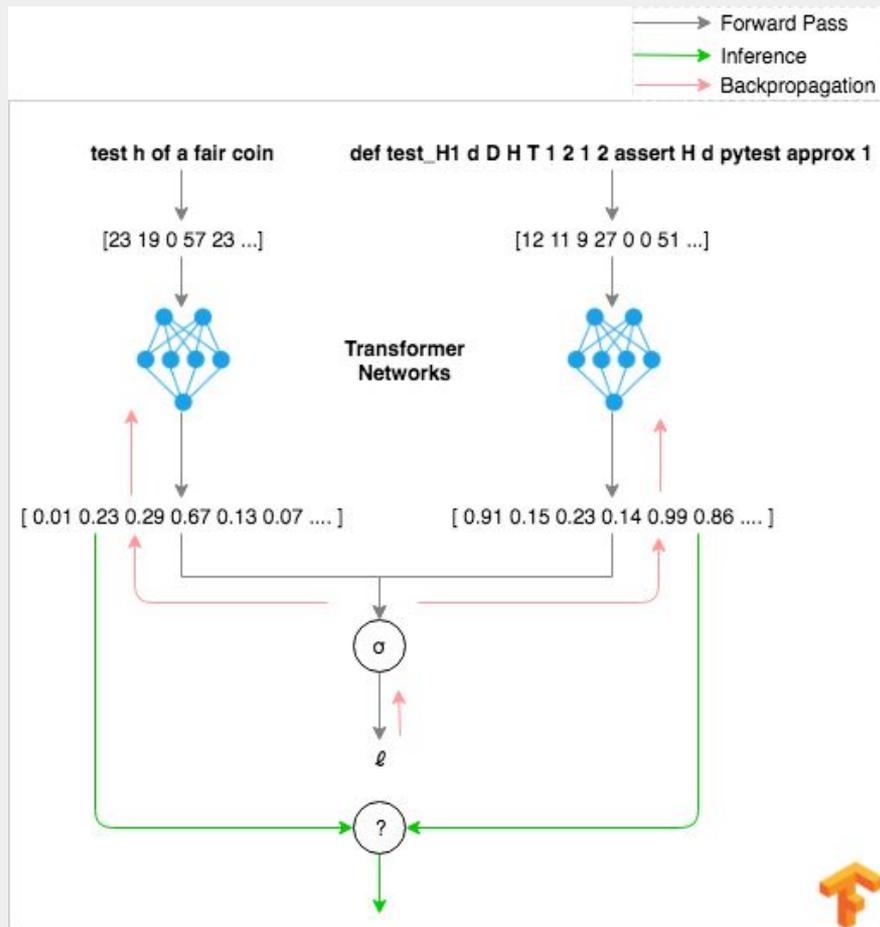
This demo is trained on a limited batch of Python code which
limits the quality and quantity of results we offer.

Try one of our suggestions
concatenate files together
pretty prints the statistics aggregated by summary
start flask app
compute new velocities for every particle
return the ip of the device
```



The model

- Model embeds search query and code in the same space
 - predicts how well the code matches the query
- Built on Tensor2Tensor library of models

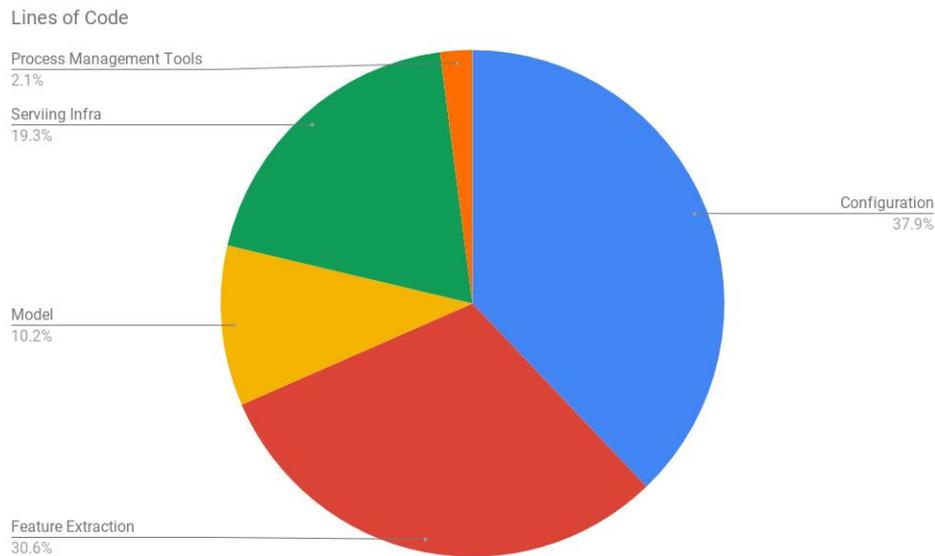


Code Breakdown

1. Building ML productions requires lots of devops
2. Kubeflow makes it easy to build ML products using Kubernetes



Most code & config is not related to the model



Big industry challenge



ginablaber

@ginablaber

Follow



The story of enterprise Machine Learning: “It took me 3 weeks to develop the model. It’s been >11 months, and it’s still not deployed.”

[@DineshNirmalIBM](#) [#StrataData](#) [#strataconf](#)

10:19 AM - 7 Mar 2018

