BLL Scale

BL Scale

Meet the Team

Fatima Zaidouni

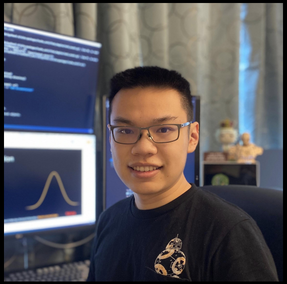
University of Rochester

Physics and Astronomy

Peter Ma

University of Toronto

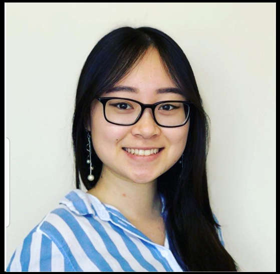
Math and Physics Specialist Program



Yuhong Chen

UC Berkeley

Computer Science



Shirley Wang

UC Berkeley

Data Science and Business Administration

Rachel Zhong

Georgia Tech

Computer Science and Math

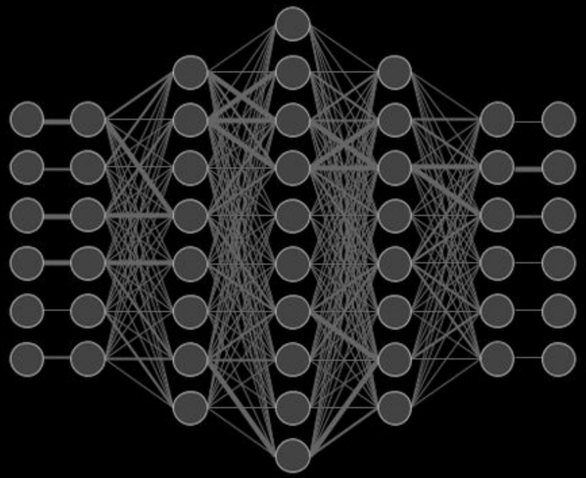
BL Scale

Table of Contents

Overview

We want to develop infrastructure to scale algorithms to deal with petascale data volumes.

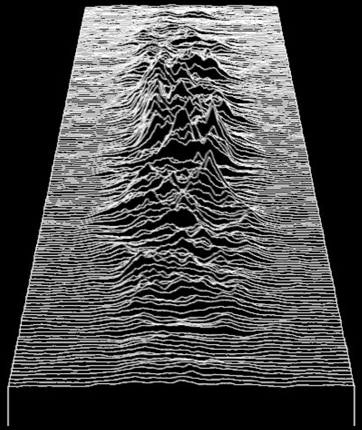
We want a variety of algorithms to search for a wider range of possible SETI signals.



BL@Scale.

Is a **reservoir** of search algorithms borrowing signal processing and ML techniques from **industry**.

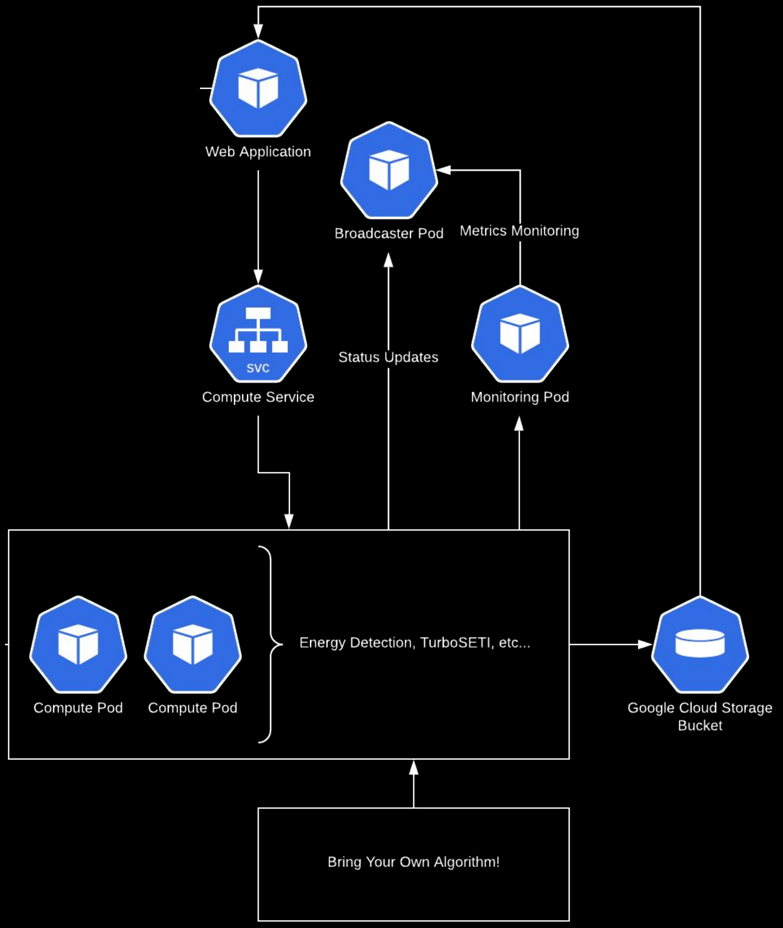
We developed a **cloud-based infrastructure** to seamlessly **scale** computing demand and algorithms across large datastores.

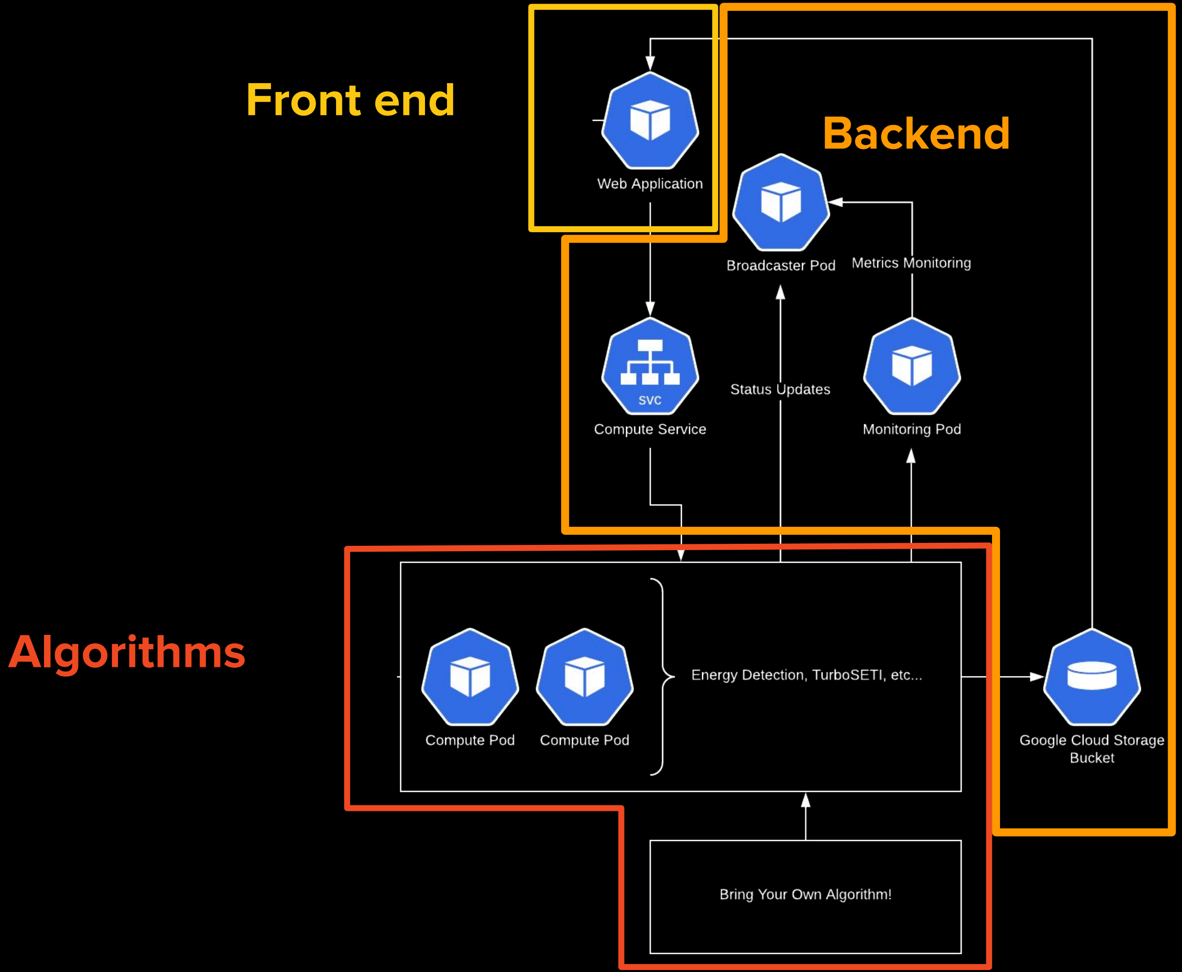


Our Platform.

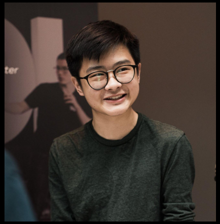
Containerized compute pods host algorithms

Networking, scaling, installation abstracted away

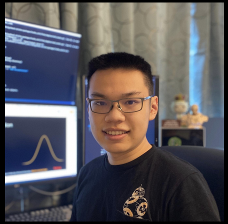



Front end
Backend
Algorithms
Web Application
Broadcaster Pod
Metrics Monitoring
SVC
Compute Service
Status Updates
Monitoring Pod
Compute Pod
Compute Pod
Energy Detection, TurboSETI, etc...
Google Cloud Storage Bucket
Bring Your Own Algorithm!

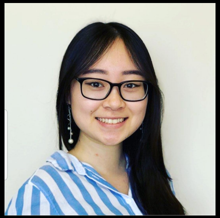
Frontend & Web Application



Peter Ma

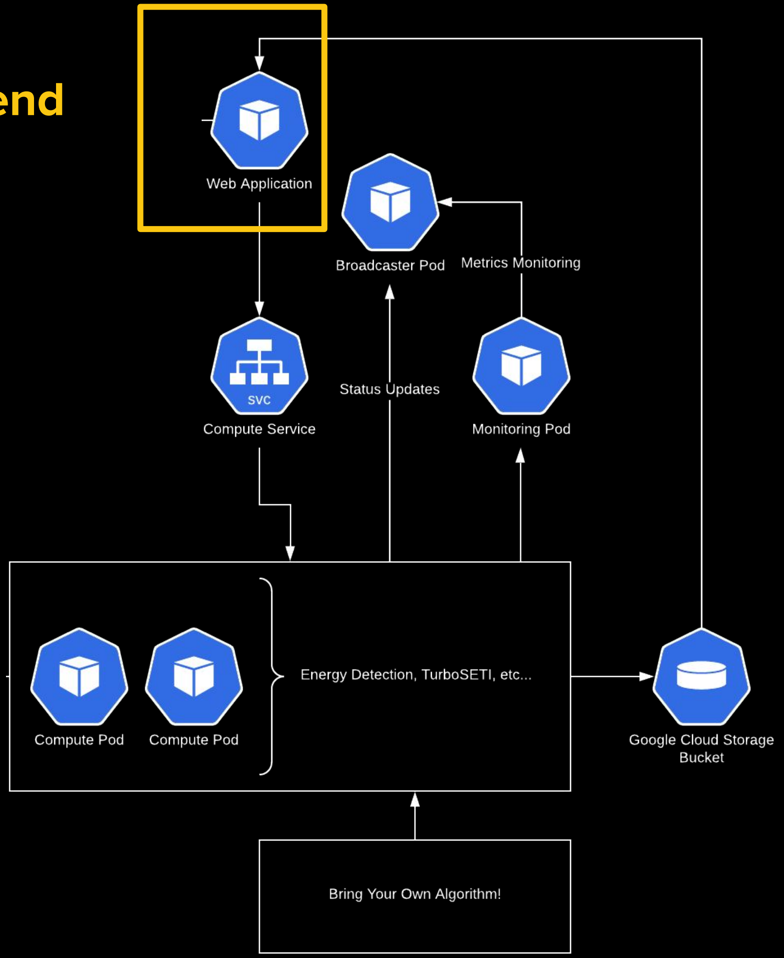


Yuhong Chen

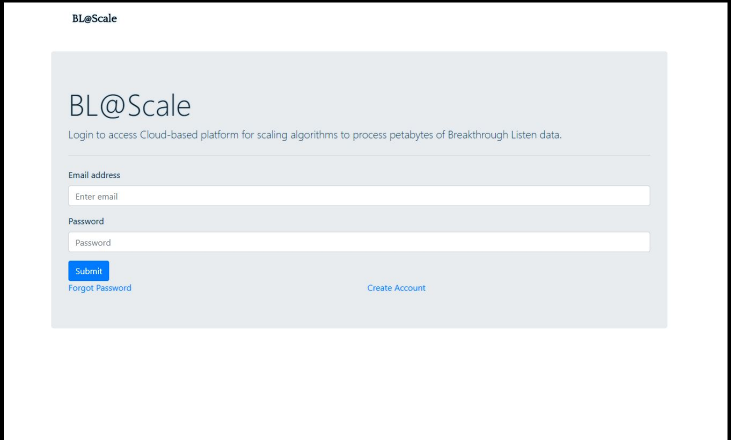


Shirley Wang

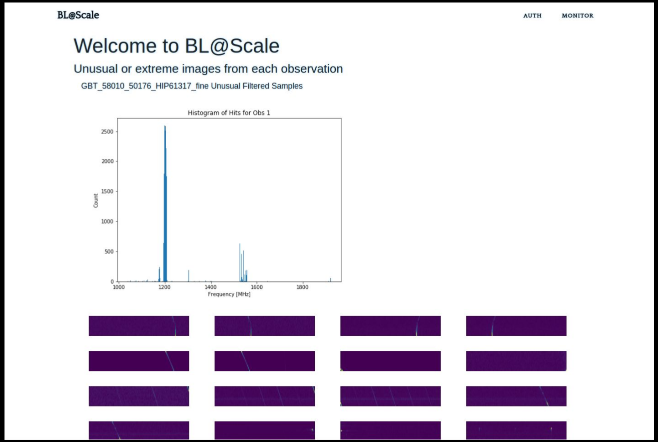
Front end


Web Application
Broadcaster Pod
Metrics Monitoring
SVC
Compute Service
Status Updates
Monitoring Pod
Compute Pod
Compute Pod
Energy Detection, TurboSETI, etc...
Google Cloud Storage
Bucket
Bring Your Own Algorithm!

Where We Started...



Login



Results

Displaying Histograms and Images

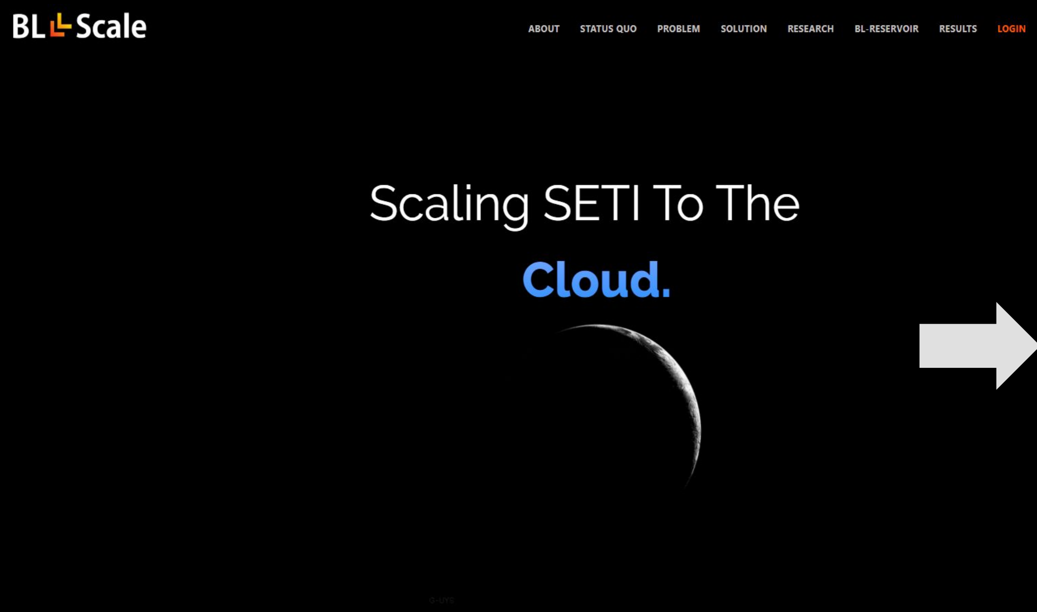
- Wrote a function that intakes a pandas dataframe from the GCP, create a histogram of the frequency distribution, and then convert it to a base64 string
- Originally wrote a function that pulled images straight from the GCP using url
    - Edited the energy detection file so that it would save all images and the 12 filtered images in numpy array with shape (pixel width, pixel length, number of images)
    - Added the basics of caching with firebase so decreases time for results page refresh

X

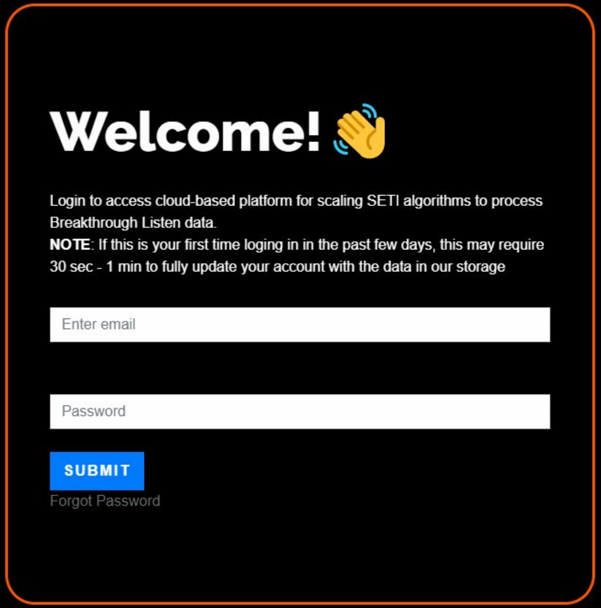
Scaling SETI To The Cloud.

Demo

http://seti.berkeley.edu/bl-scale


BL Scale
ABOUT
STATUS QUO
PROBLEM
SOLUTION
RESEARCH
BL-RESERVOIR
RESULTS
LOGIN
Scaling SETI To The
Cloud.

index


Welcome!
Login to access cloud-based platform for scaling SETI algorithms to process Breakthrough Listen data.
NOTE: If this is your first time loging in in the past few days, this may require 30 sec - 1 min to fully update your account with the data in our storage
Enter email
Password
SUBMIT
Forgot Password

Login

Hey! 👋

peterxiangyuanma@gmail.com

You are currently logged in as peterxiangyuanma@gmail.com. This is the admin page where you can control and oversee the scaling and the status of various triggers. Account activity will be tracked with each trigger to log the results.

NOTE: Please let, Yuhong, Peter or Shirley know about any issues.

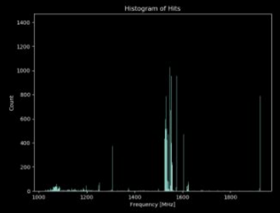
LOGOUT

Monitor 🌡️

Monitor the kubernetes cluster remotely!

MONITOR

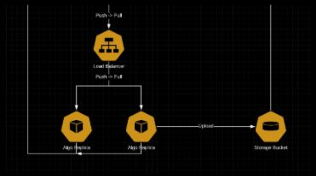
Results



Checkout the results from the latest searches.

RESULTS

Trigger 💥



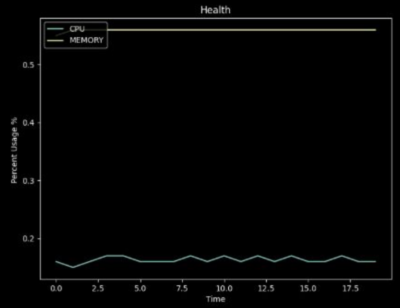
You can trigger the latest searches from this page.

TRIGGER

Dashboard

bl-scale-algo-56f69bf465-2z2p4

STATUS



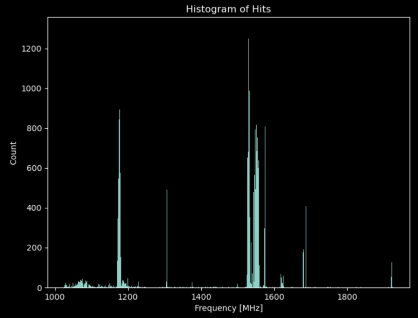
CPU 0.16%

RAM 0.56%

Monitor

Results

GBT_58064_82832_HIP91462_fine Unusual Filtered Samples



Algorithm Type: Energy-Detection

Timestamp: Fri Aug 7 21:32:09 2020

Processing Time: 33.96 minutes

Results

Algorithm Package

Algorithm Name

File URL

SUBMIT

Most Recent Triggers

HIP1368

Telescope:GBT
Algorithm: Energy-Detection

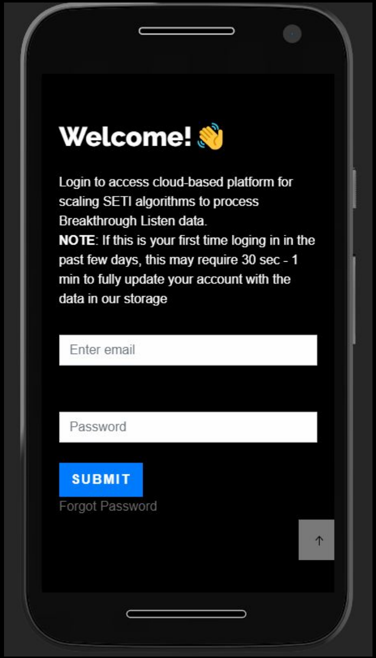
MJD:57803.75190
Time Triggered:
Mon Aug 10 18:06:03 2020

Resolution:mid.h5
Message:
energy_detection/energy_detection_mid.py finished in 68.88403534889221 seconds. Results uploaded to gs://bl-scale/GBT_57803_75190_HIP1368_mid

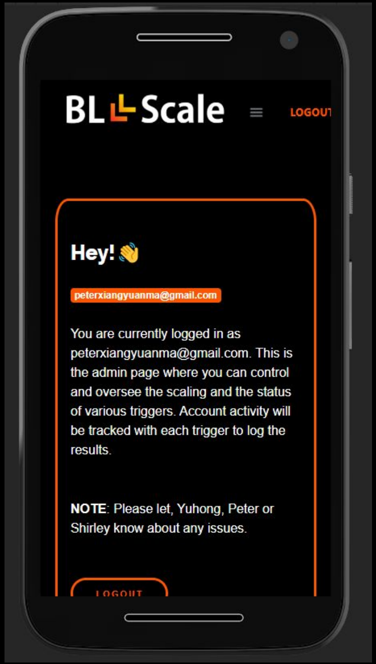
HIP1152

Trigger

index

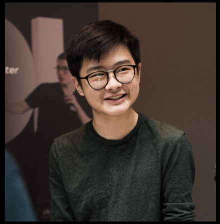


Login

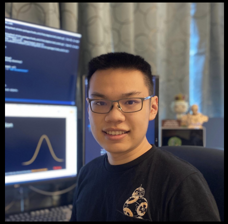


Dashboard

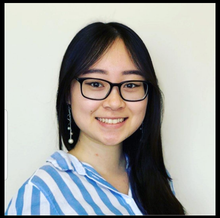
Backend & Networking



Peter Ma



Yuhong Chen



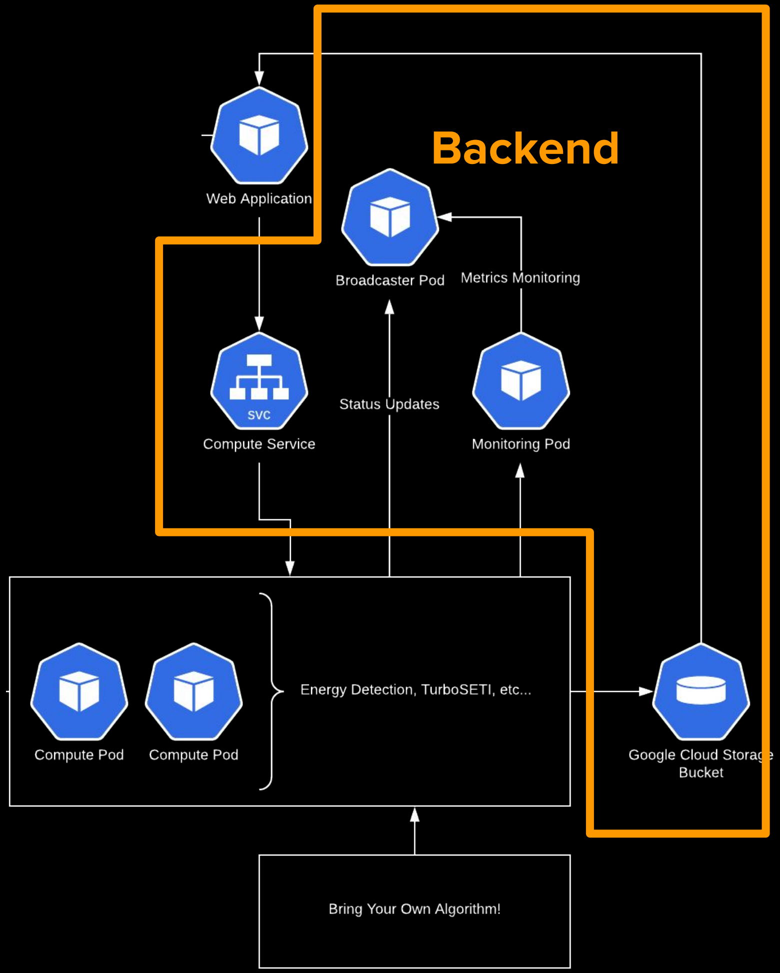
Shirley Wang

Backend & Networking

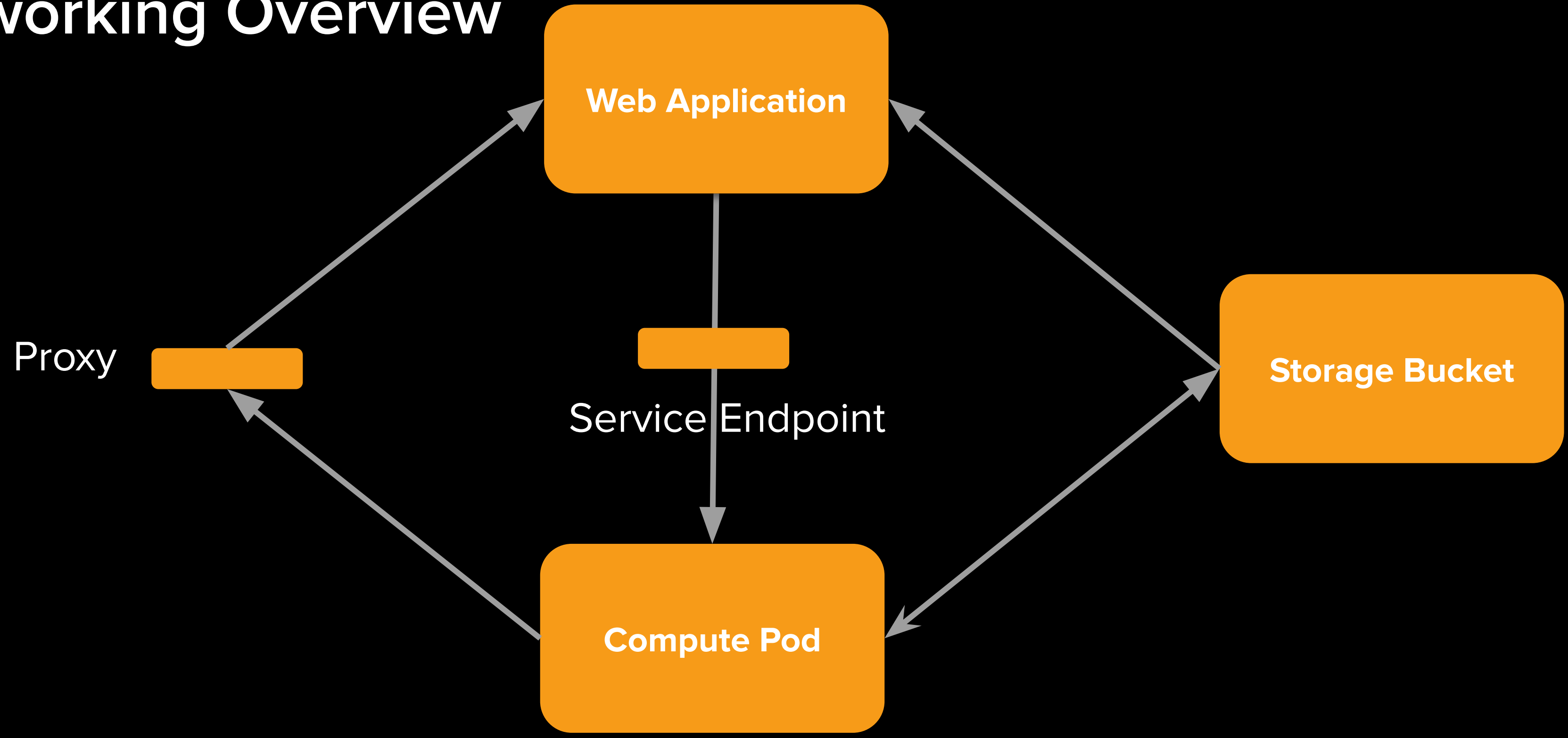
Socket Networking

Extended Networking

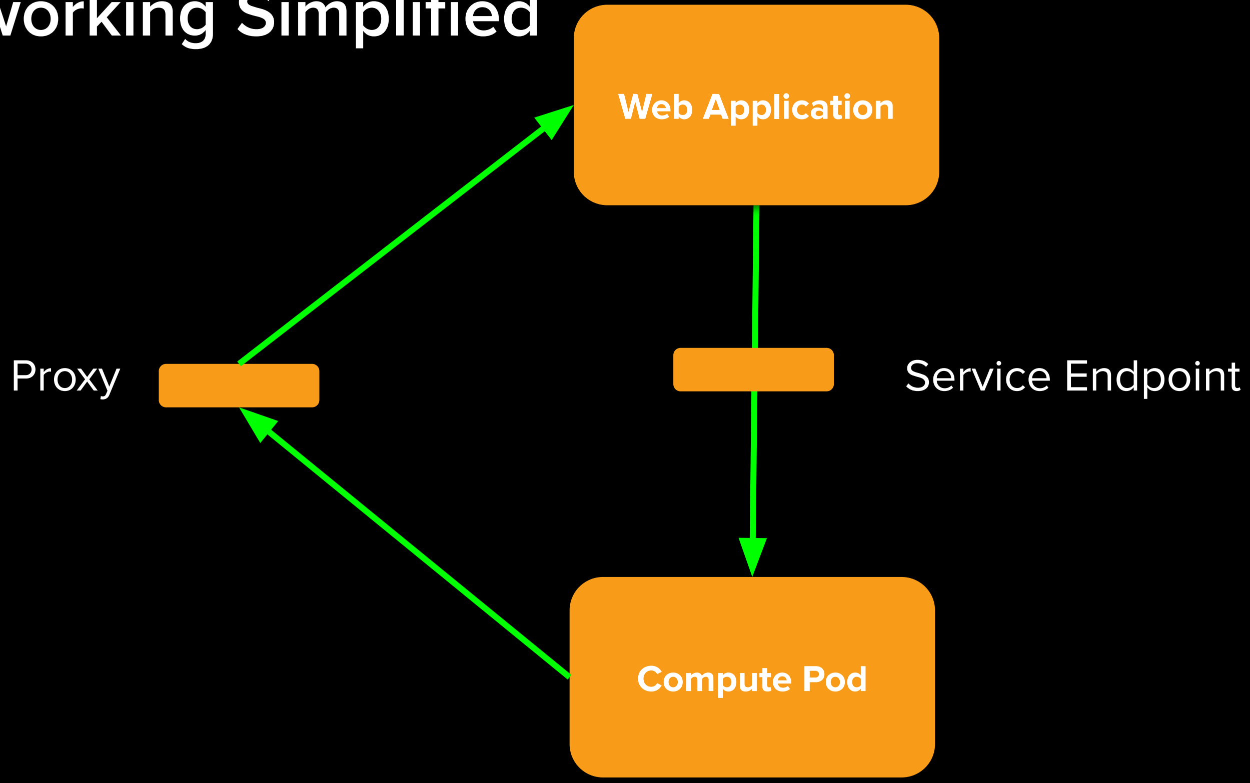
Pod Environments


Backend
Web Application
Broadcaster Pod
Metrics Monitoring
SVC
Compute Service
Status Updates
Monitoring Pod
Compute Pod
Compute Pod
Energy Detection, TurboSETI, etc...
Google Cloud Storage
Bucket
Bring Your Own Algorithm!

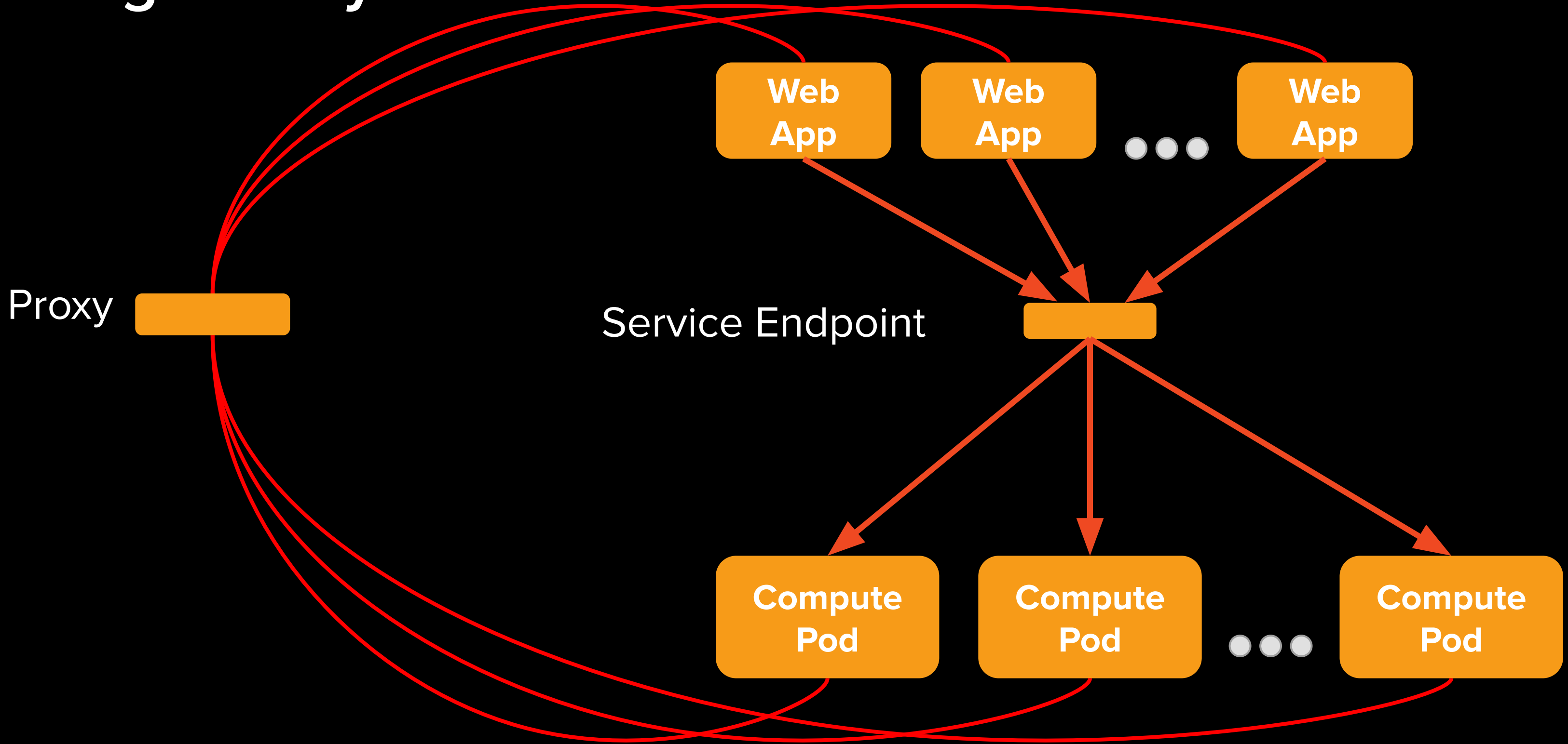
Networking Overview



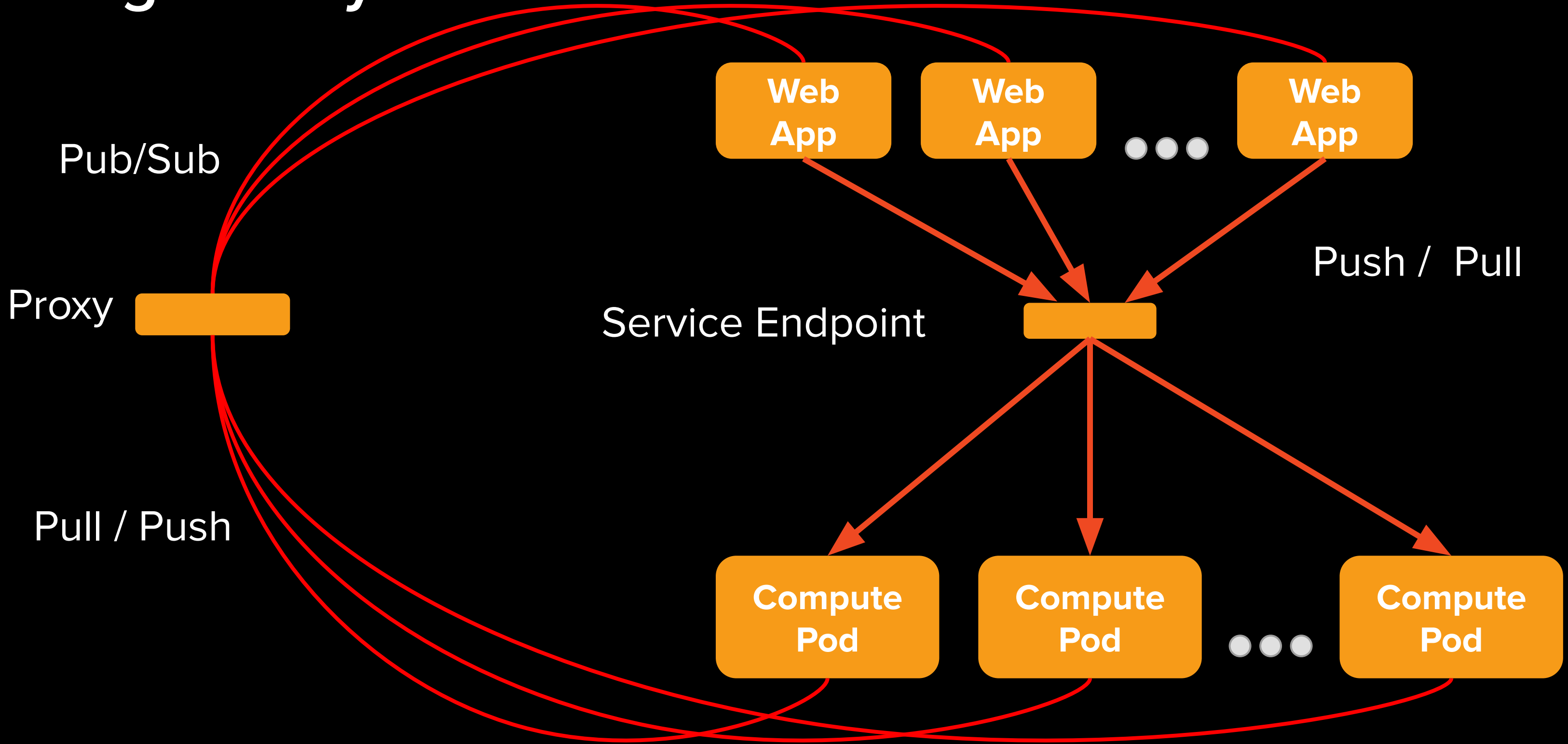
Networking Simplified



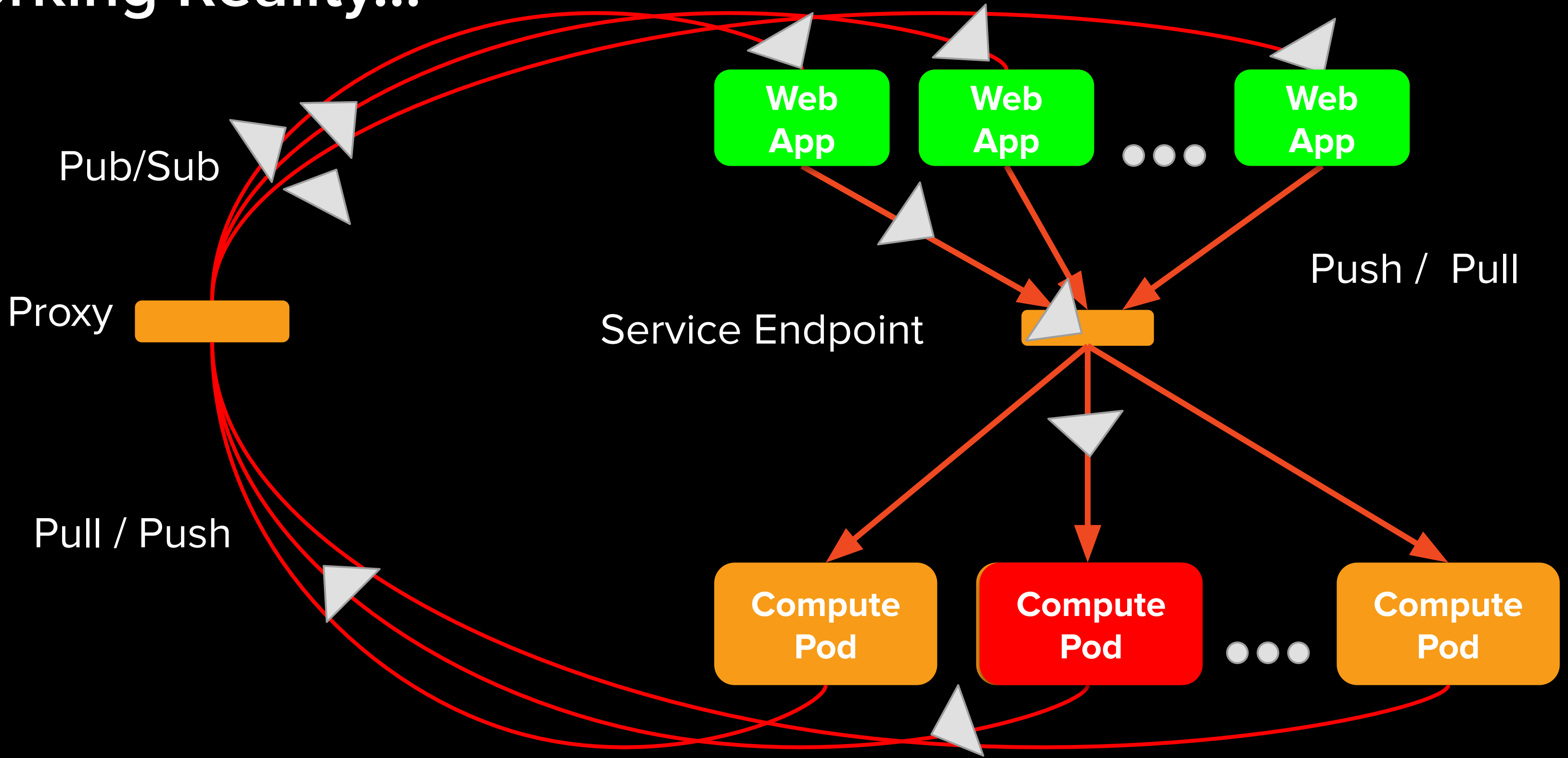
Networking Reality...



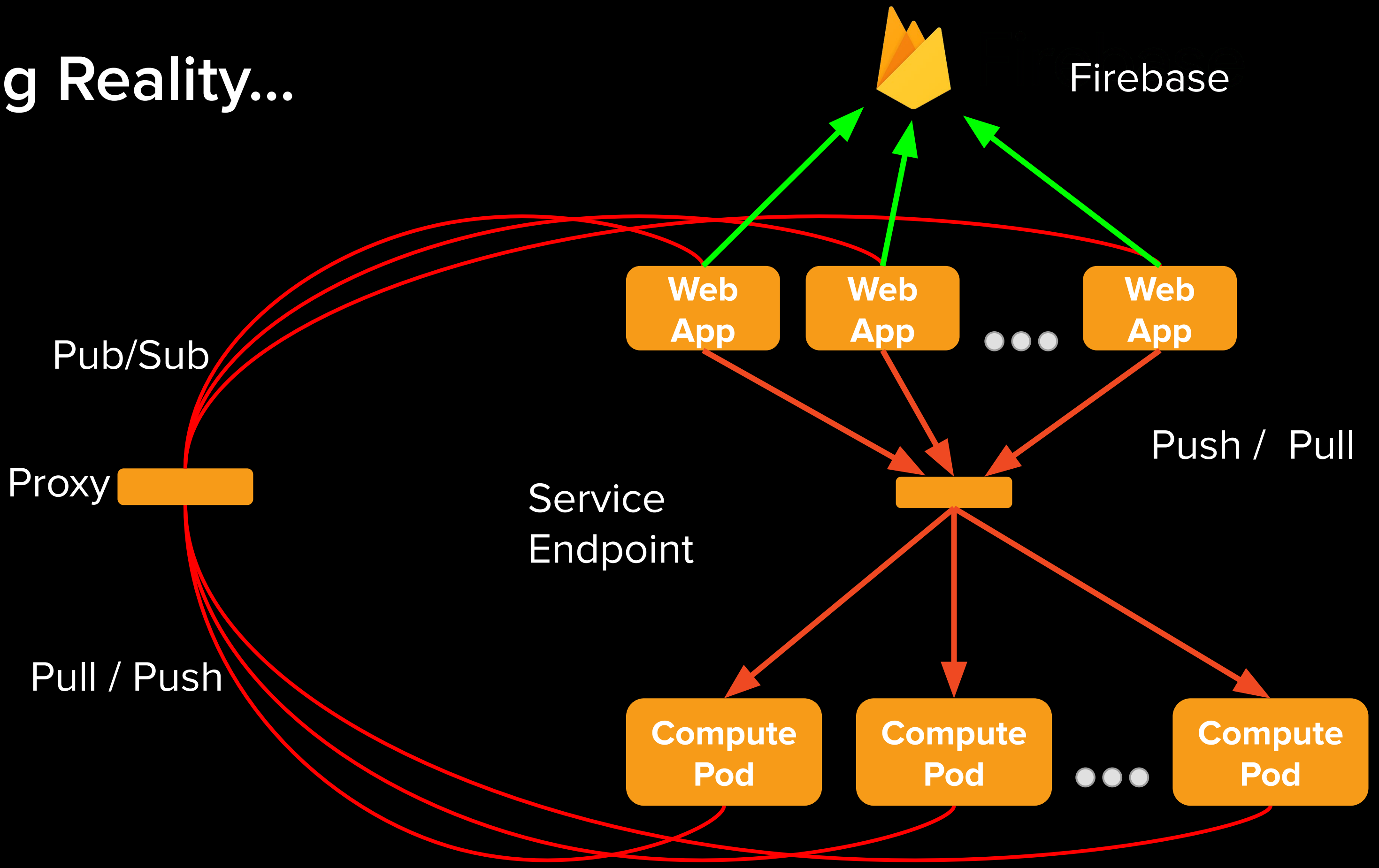
Networking Reality...



Networking Reality...

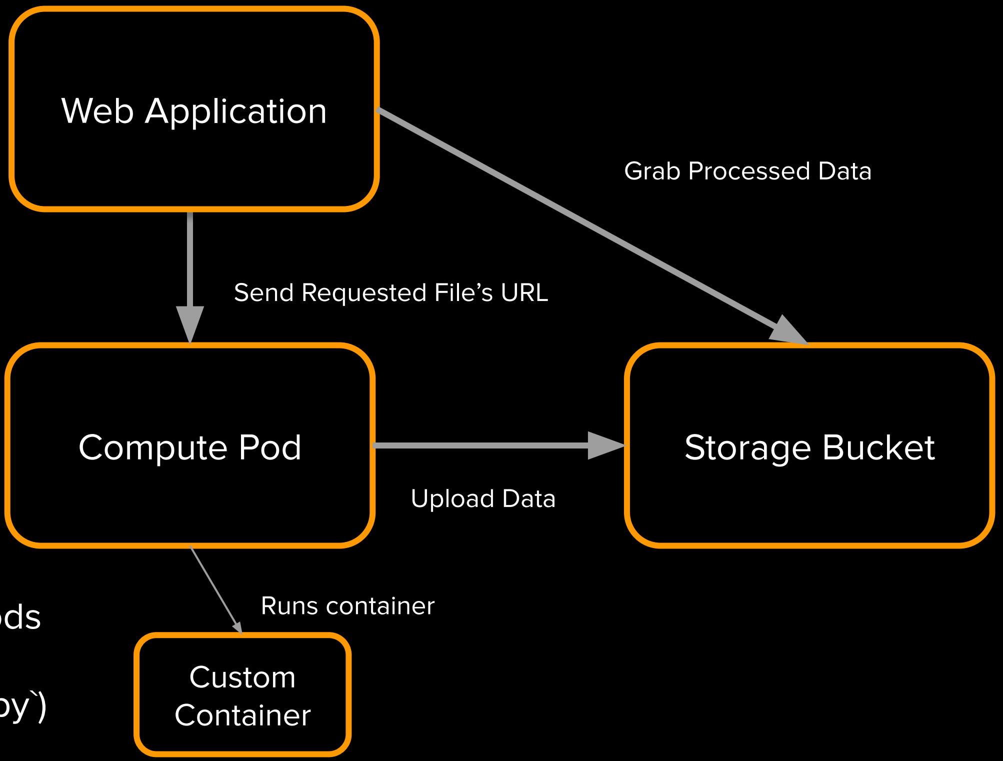


Networking Reality...



First Iteration: Running Code

```
while True:
        requested_url = receive()
        # download input file
        wget.download(file)
        # run algorithm
        os.system("...")
        # upload files
        upload(<output_file>)
```



Required developers to ship their code in specific docker containers that compute pods would pull and run
(Compute Pod runs `<selected_algorithm>.py`)

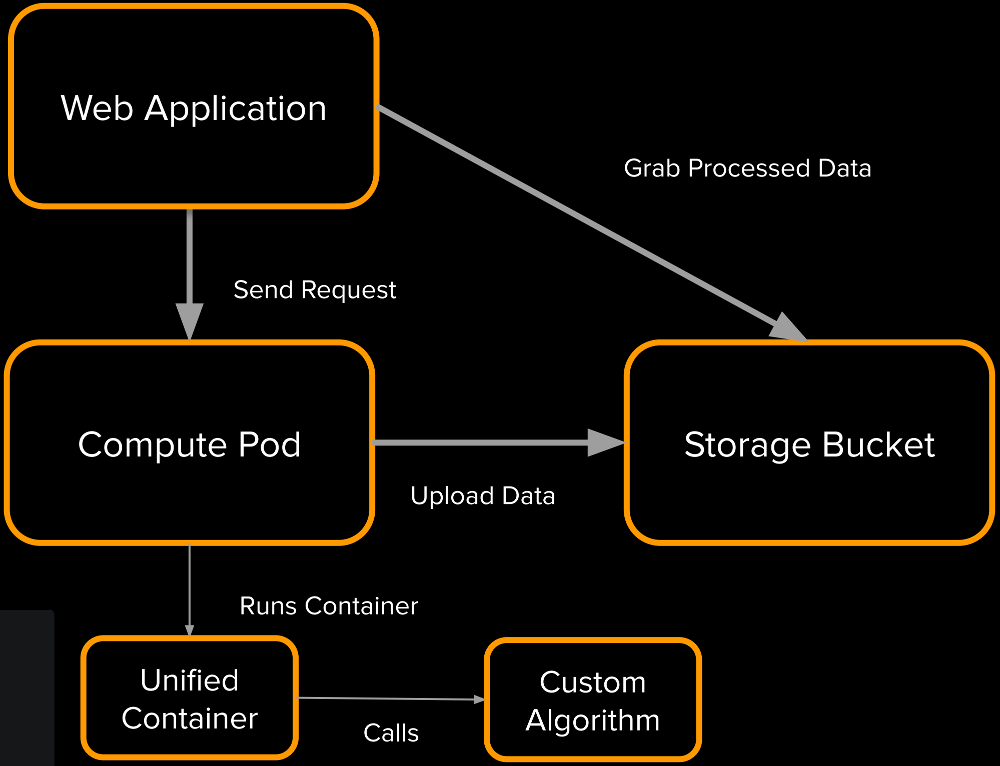
Upgrades: Unified Containers

Lets users request different algorithms from frontend without needing to switch containers in the backend

Shift networking responsibility to us

User still has to use our upload utility, requires us to have knowledge about the output structure

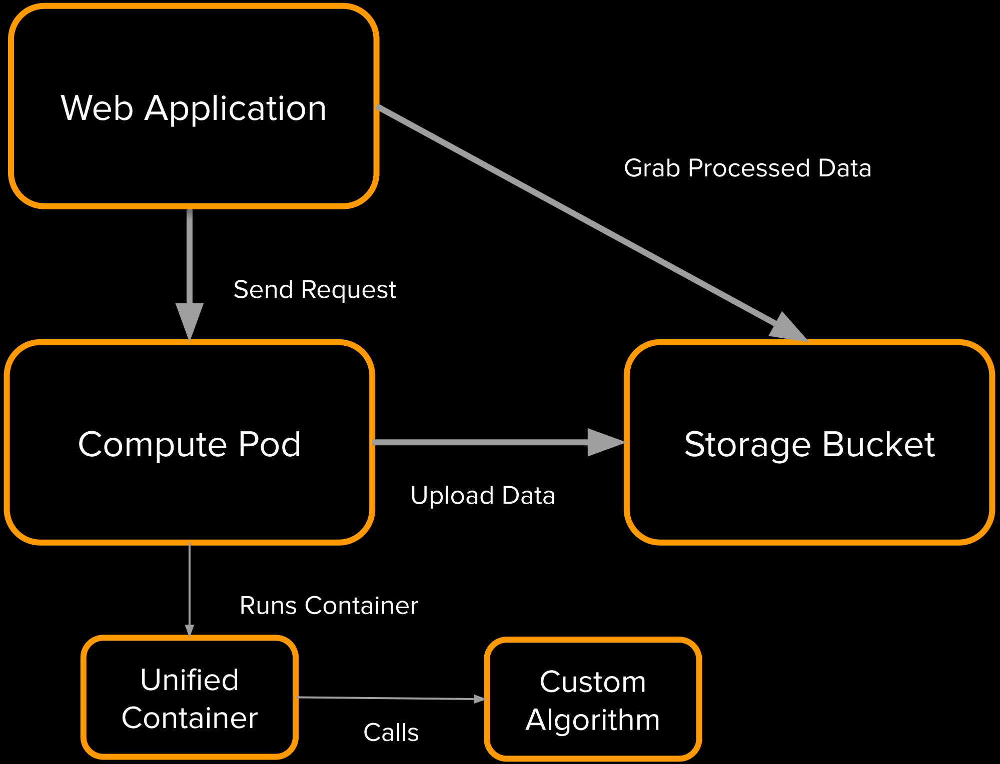
```
while True:
  request_dict = receive()
  requested_algorithm = request_dict["algorithm"]
  requested_file = request_dict["file"]

  # download file
  wget.download(requested_file)

  # run algorithm
  os.system("python3 <requested_algorithm> <file> <output_directory>")

  # upload to bucket
  upload("<output_directory>")
```



Upgrades: FUSE Mount

Google Storage FUSE mounts lets us directly mount a storage bucket as a local directory

Allows us to support anything that one can run as a shell command, without worrying about output structure



Upgrades: Autoscaling

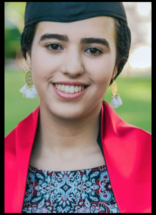
Pod Virtual Environments

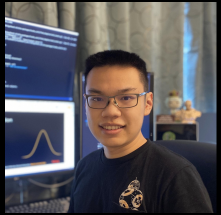
Isolate each algorithm package's dependencies into its own virtual environment

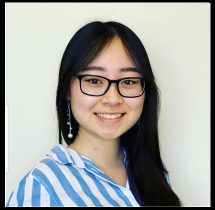
Call algorithms from different working directories to allow them to access their own subpackages properly

TurboSETI x

Machine Learning & Algorithms







Fatima Zaidouni | **Peter Ma** | **Yuhong Chen** | **Shirley Wang** | **Rachel Zhong**

Machine Learning Algorithms

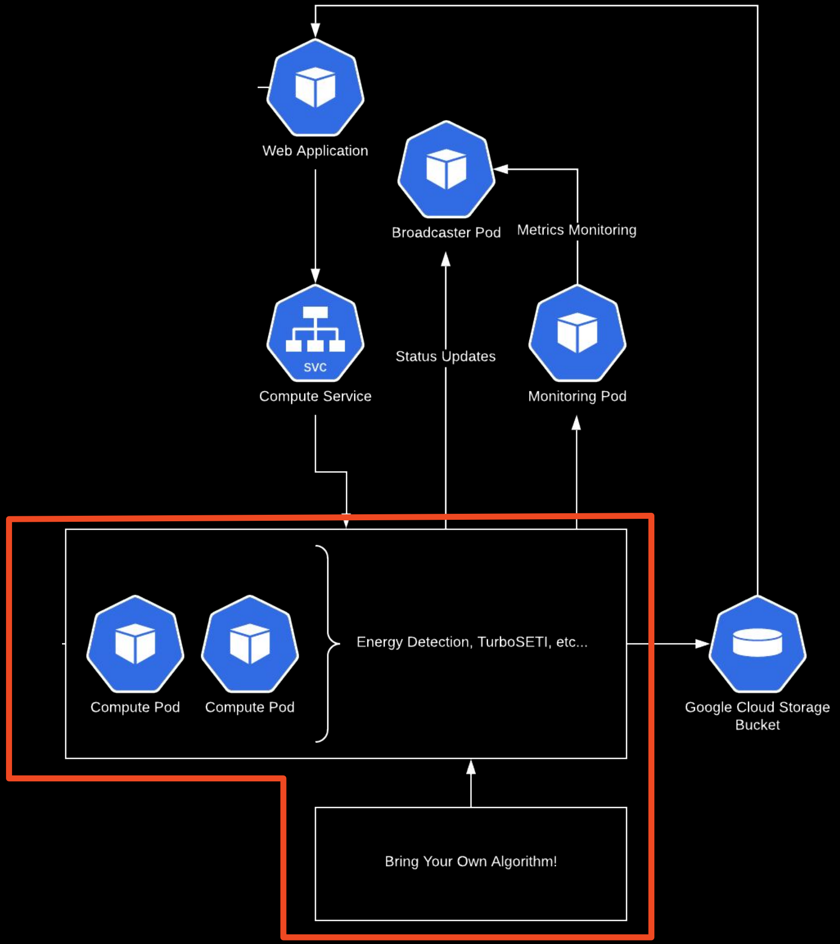
Rachel and Fatima's Algorithms

DeepSETI

Clustering

Energy Detection w/ ABT

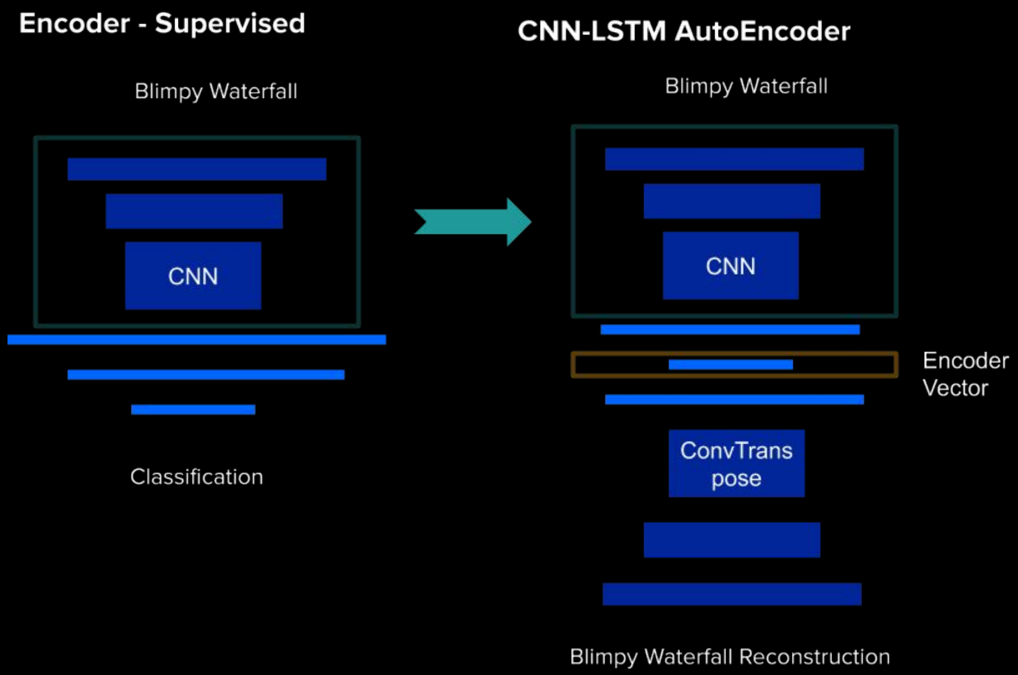
Algorithms



Web Application
Broadcaster Pod
Metrics Monitoring
Status Updates
SVC
Compute Service
Monitoring Pod
Compute Pod
Compute Pod
Energy Detection, TurboSETI, etc...
Google Cloud Storage Bucket
Bring Your Own Algorithm!

Rachel’s and Fatima’s Algorithms

DeepSETI



Problem

We think we know what we're looking for, but we're also unsure on what to exactly looking for.

Solution

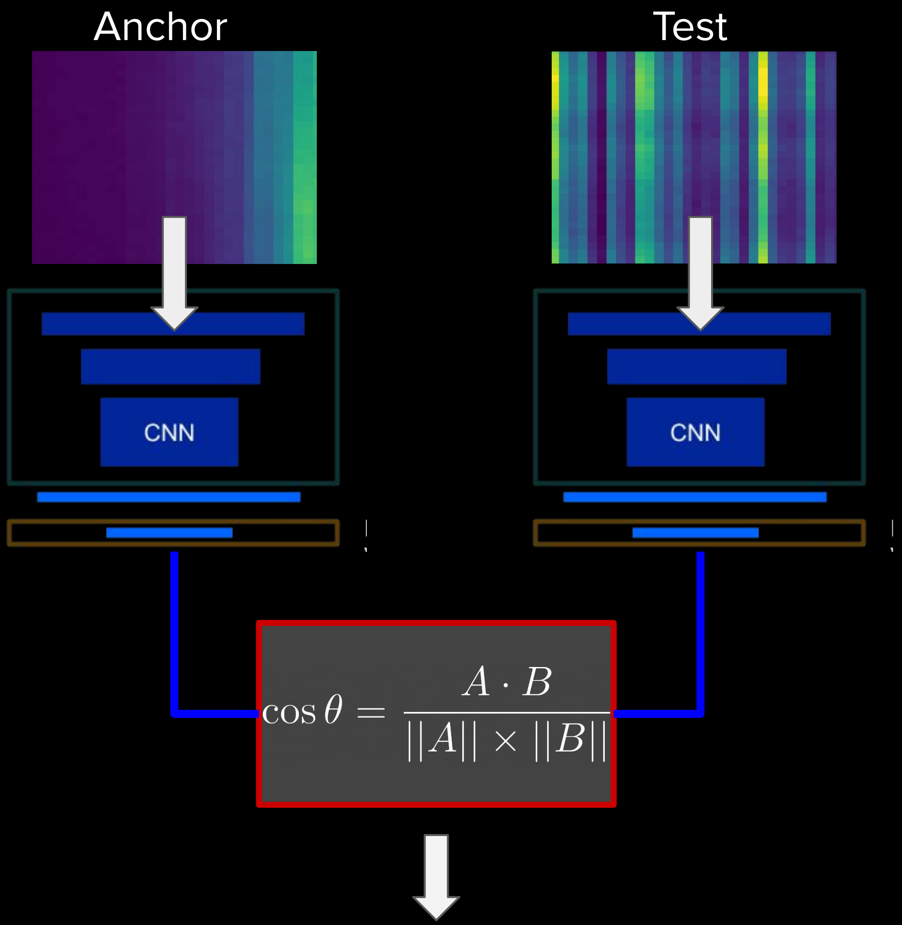
We combine supervised learning (what we know about) with an unsupervised approach (explore signals we don't expect)

DeepSETI

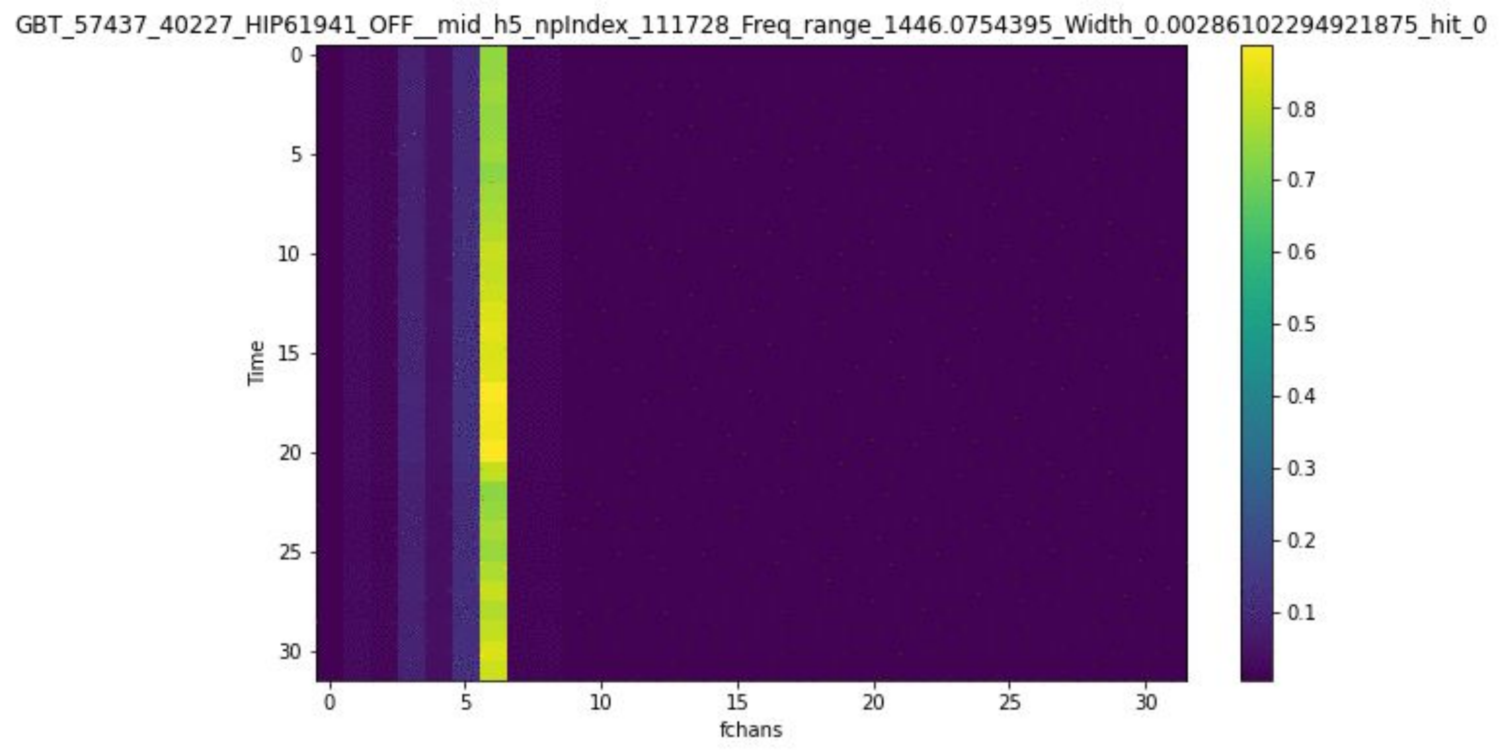
Prediction

We want to compare how anomalous a signal might appear by taking the cosine distance between encoded signals.

Larger the cosine angle the more anomalous the input spectrogram is to its respective anchor.



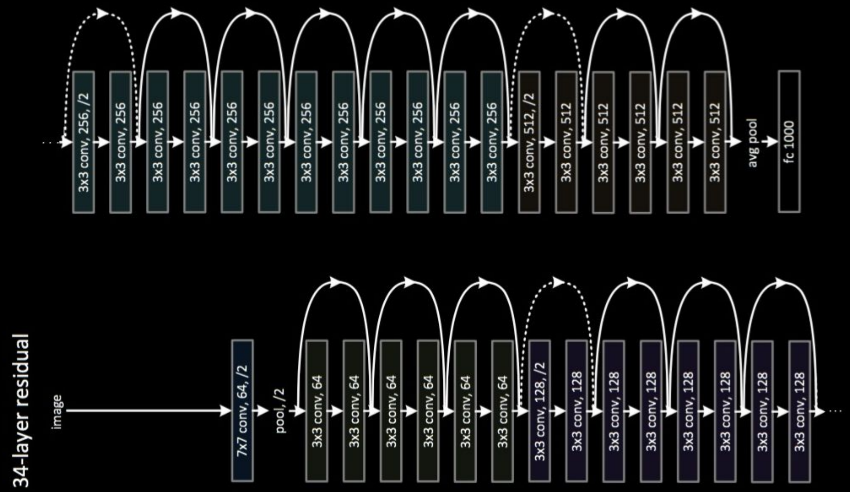
DeepSETI



GBT_57437_40227_HIP61941_OFF__mid_h5_npIndex_111728_Freq_range_1446.0754395_Width_0.00286102294921875_hit_0
Time
fchans
0
5
10
15
20
25
30
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1

RESNET 50 Clustering

[Kaggle Notebook]



Adaptive Bayesian Thresholding

Null Hypothesis H0: The signal is **not** present
Alternate Hypothesis H1: The signal **is** present
Lambda_FA = 0.8

$$T_j = \sigma_j \sqrt{2 \log_e (N)}$$

$$|X(j,k)| \underset{\mathcal{H}_1}{\overset{\mathcal{H}_0}{\lesseqgtr}} \frac{\hat{\mu}_j}{2} + \frac{\hat{\sigma}_j^2}{\hat{\mu}_j} \log_e \gamma_j \triangleq \Theta_j \quad \forall k \in \mathcal{B}$$

$$log_e \gamma_j = \log_e [\frac{\lambda_{FA}}{1 - \lambda_{FA}} - \frac{P(H_0)}{P(H_1)}] \triangleq LL_M + \log_e \frac{P(H_0)}{P(H_1)}$$

Preparing the Data

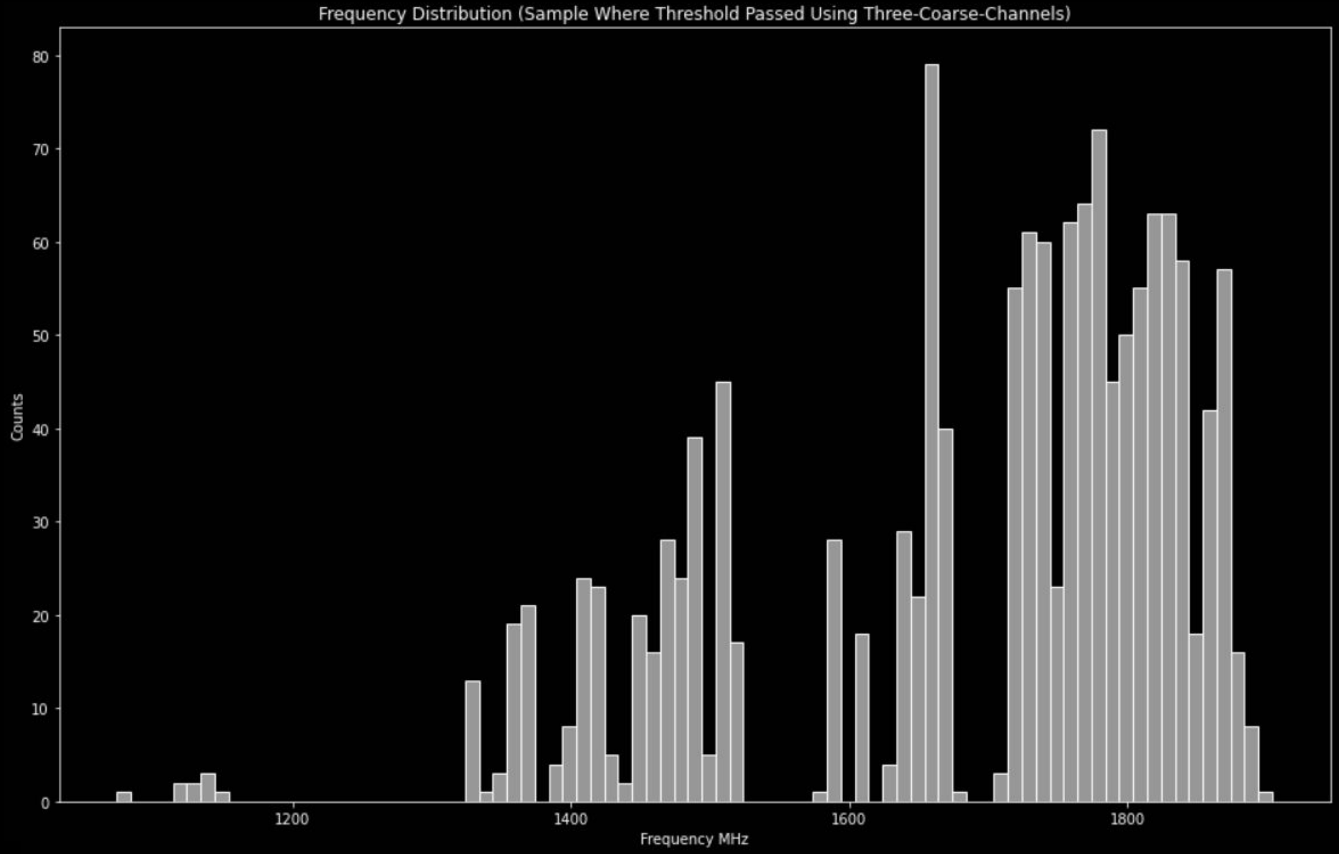
	index	statistic	pvalue	freqs	coarse_channel	three_final_threshold
867390	111039360	29.487015	3.953447e-07	1616.029143	105	31216.029902
1722188	220466944	77.145136	1.770706e-17	1310.292006	210	27129.379536
233464	29886976	18.501610	9.603432e-05	1842.766285	28	32.687244
774895	99198592	13.844376	9.856709e-04	1649.111867	94	44.369179
162409	20790784	19.273751	6.527668e-05	1868.180752	19	32.977823
...	...	...	...	...	...	...
1255883	160772608	36.472754	1.202381e-08	1477.076054	153	39.648164
6475	828800	2025.826816	0.000000e+00	1923.953891	0	62491.725425
746019	95502080	52.605792	3.773960e-12	1659.439802	91	27.395793
2480780	317578496	25.549333	2.831607e-06	1038.965464	302	4485.579595
451324	57776512	22.496587	1.302951e-05	1764.843822	55	26.198138

10000 rows × 6 columns

- Took 28 observations from our GCP and concatenated them together ~ 70 million rows
  - Took a random sample of 10,000 rows
- Calculated the threshold for each 3-coarse channel window (there are 308 channels)
  - For example, channel 3 median and mad calculated by taking the median and mad of channels 2,3,4. Channel 4 by calculating median and mad of channels 3,4,5 etc.

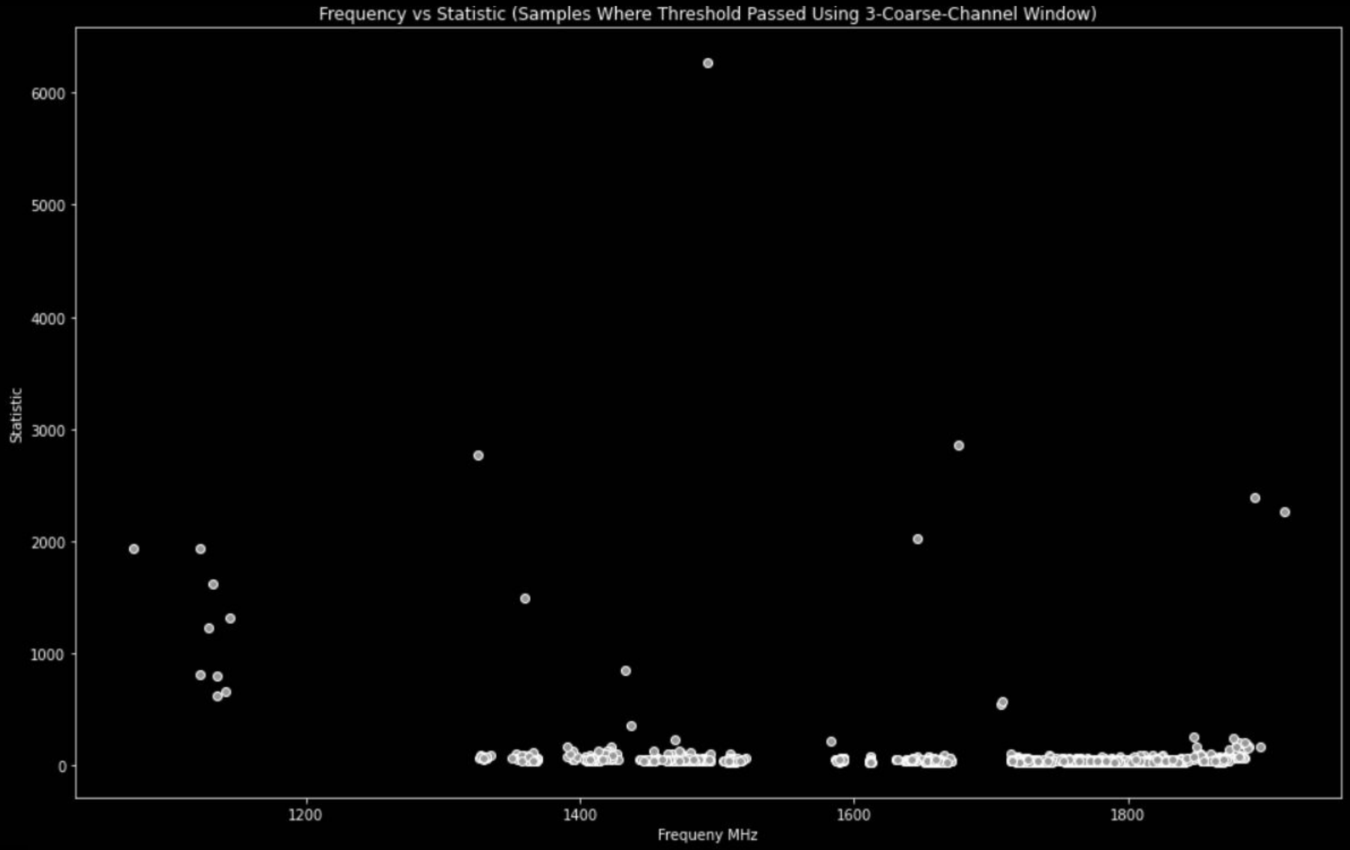
Analysis for 3-Coarse Channel Window

Using 10000 Randomly Sampled Rows



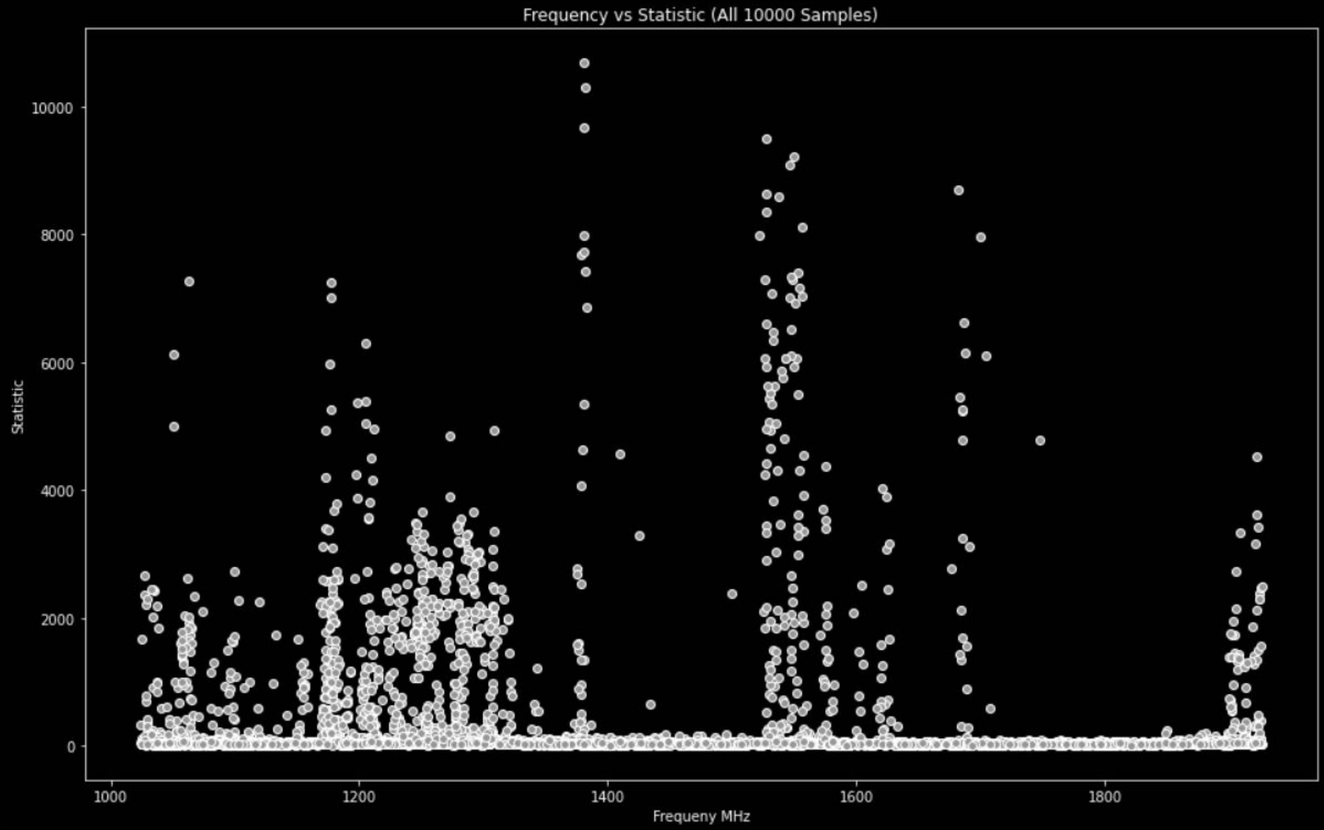
Analysis for 3-Coarse Channel Window

Using 10000 Randomly Sampled Rows



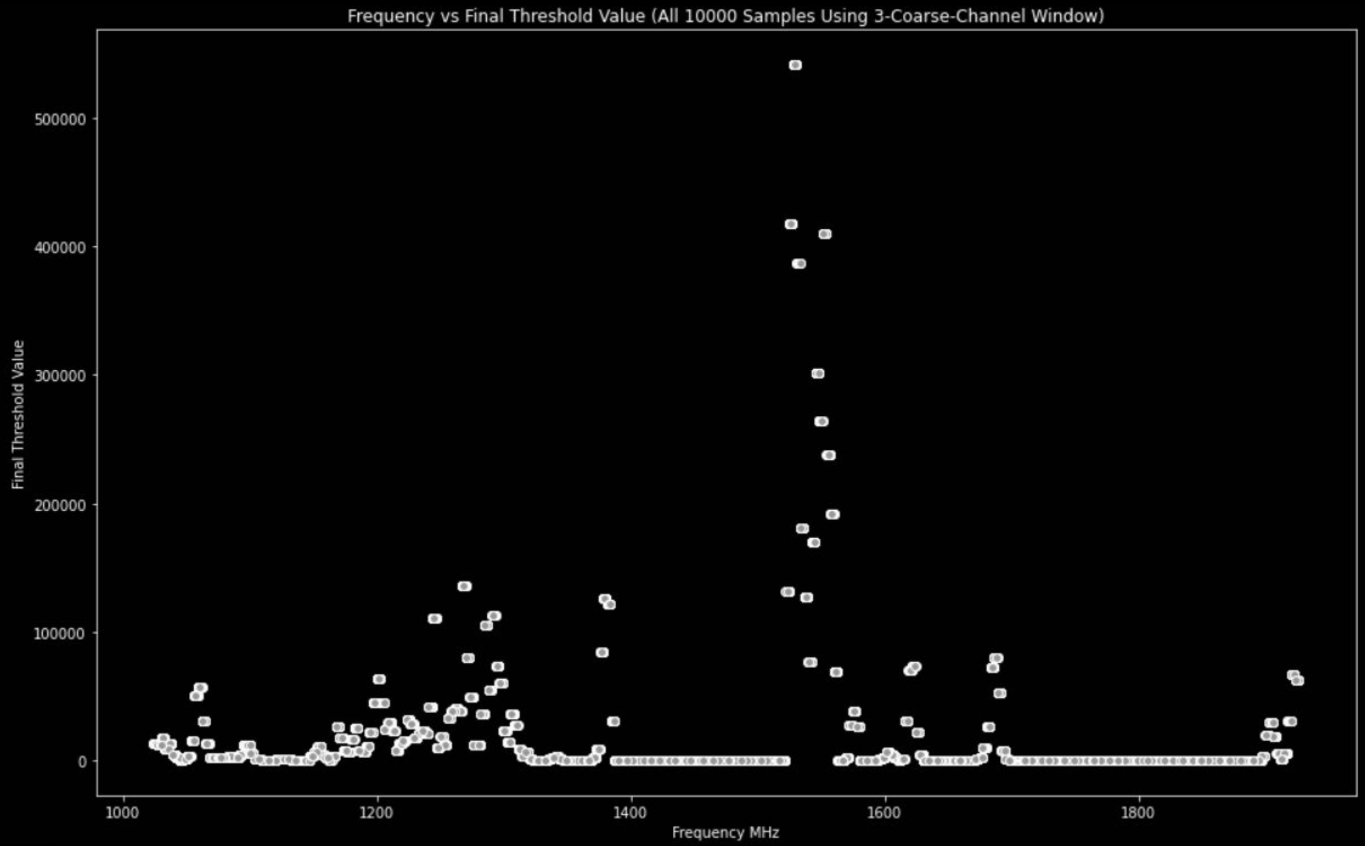
Analysis for 3-Coarse Channel Window

Using 10000 Randomly Sampled Rows



Analysis for 3-Coarse Channel Window

Using 10000 Randomly Sampled Rows



Conclusion

Scaling: Backend allows researchers to run 10s of replicas of their algorithms at once, without worrying about infrastructure problems like spinning up GCP VMs, installation, etc.

We can now virtually spin up one of the largest searches for extraterrestrial life at the click of a button.

Variety: We developed a reservoir of search algorithms for various applications like Anomaly Detection, Clustering, Object Detection, and more!

We've built the groundwork for the next generation SETI research. Allowing collaboration from the larger ML community.

Next Steps?

- Batch processing (Frontend)
- GPU support
- Doing a truly large scale search with the data we have
- Integration for Open Data Archive + more
- Deploying to observatories like GBT for better integration

Acknowledgements

Andrew Siemion

Steve Croft

Matt Lebofsky

Walter Reade

Tarin Ziyayee

Breakthrough Listen

NSF

NSF

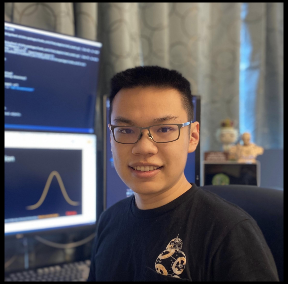
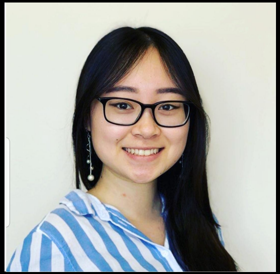
BREAKTHROUGH LISTEN

BERKELEY SETI RESEARCH CENTER

Questions?

Find Us!




Fatima Zaidouni

Email: fzaidoun@u.rochester.edu
LinkedIn:
https://www.linkedin.com/in/fatima-zaidouni-05177b17a/
Github: @fzaidouni

Peter Ma

Website: peterma.ca
Email: peterxiangyuanma@gmail.com
Twitter: @peterma02
Github: @PetchMa

Yuhong Chen

Email: yuhongc212@gmail.com
LinkedIn:
https://www.linkedin.com/in/yuhongc/
Github: @FX196

Shirley Wang

Linkedin: linkedin.com/in/shirleywang57
Email: shirleywang57@berkeley.edu
Github: @shirls537

Rachel Zhong

Linkedin:linkedin.com/in/rachel-zhong-507022151
Email: rzhong34@gatech.edu
Github: @RachelZhong98