Unsupervised Anomaly Detection on X-Ray Images

Pranav Lodha

Subarna Chowdhury Soma

Jeyasri Subramanian

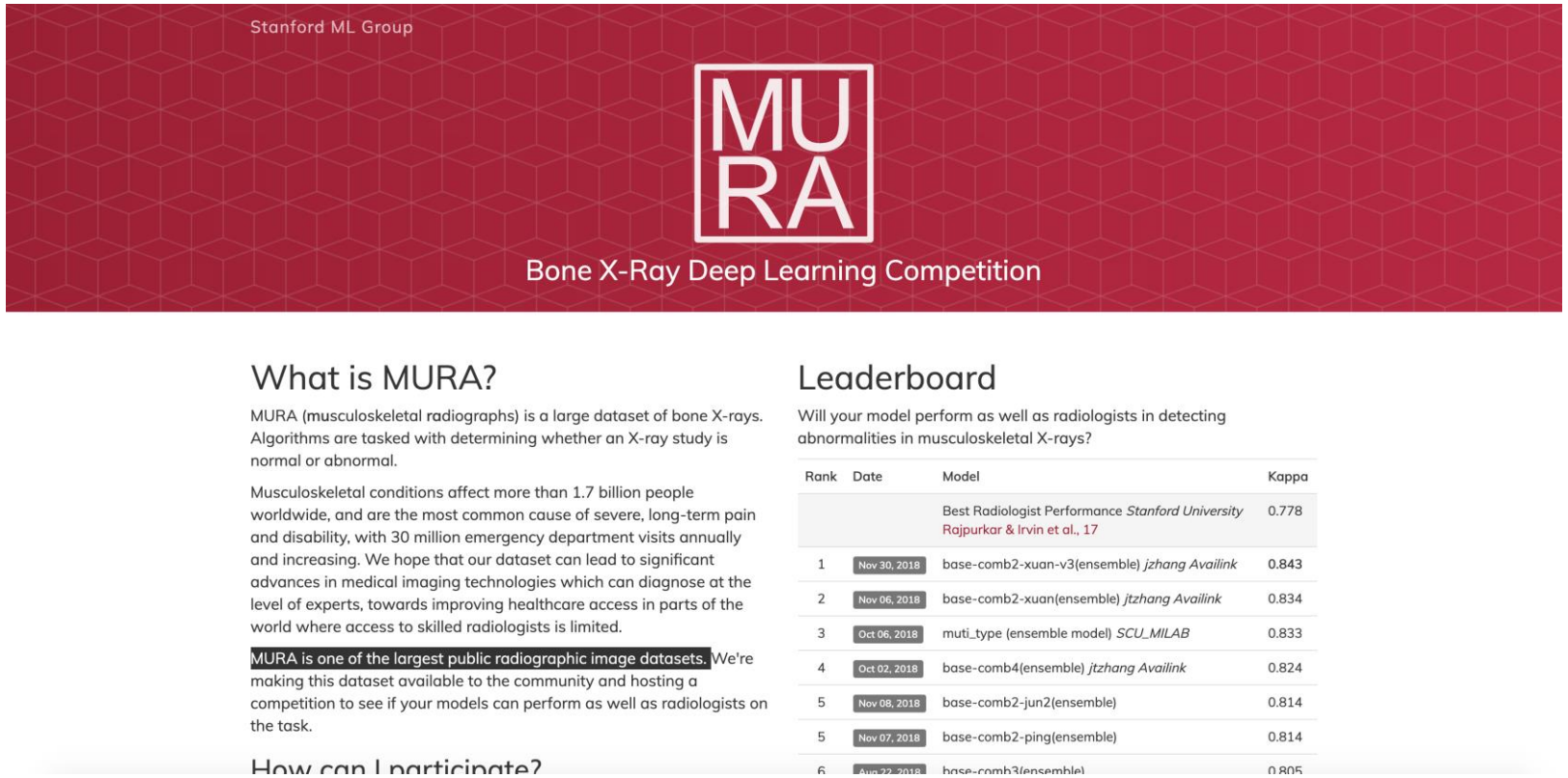
Problem Statement

Detect Anomalies in X-ray images using Generative Adversarial Networks.

GANs are mainly used to generate large volume of synthetic data.

We have utilized GAN’s discriminator and generated synthetic images to detect anomaly in X-ray images

Dataset (MURA-v1.1)



Stanford ML Group

MURA

Bone X-Ray Deep Learning Competition

What is MURA?

MURA (musculoskeletal radiographs) is a large dataset of bone X-rays. Algorithms are tasked with determining whether an X-ray study is normal or abnormal.

Musculoskeletal conditions affect more than 1.7 billion people worldwide, and are the most common cause of severe, long-term pain and disability, with 30 million emergency department visits annually and increasing. We hope that our dataset can lead to significant advances in medical imaging technologies which can diagnose at the level of experts, towards improving healthcare access in parts of the world where access to skilled radiologists is limited.

MURA is one of the largest public radiographic image datasets. We're making this dataset available to the community and hosting a competition to see if your models can perform as well as radiologists on the task.

How can I participate?

Leaderboard

Will your model perform as well as radiologists in detecting abnormalities in musculoskeletal X-rays?

Rank	Date	Model	Kappa
		Best Radiologist Performance *Stanford University* Rajpurkar & Irvin et al., 17	0.778
1	Nov 30, 2018	base-comb2-xuan-v3(ensemble) *jzhang Availink*	**0.843**
2	Nov 06, 2018	base-comb2-xuan(ensemble) *jtzhang Availink*	0.834
3	Oct 06, 2018	muti_type (ensemble model) *SCU_MILAB*	0.833
4	Oct 02, 2018	base-comb4(ensemble) *jtzhang Availink*	0.824
5	Nov 08, 2018	base-comb2-jun2(ensemble)	0.814
5	Nov 07, 2018	base-comb2-ping(ensemble)	0.814
6	Aug 22, 2018	base-comb3(ensemble)	0.805

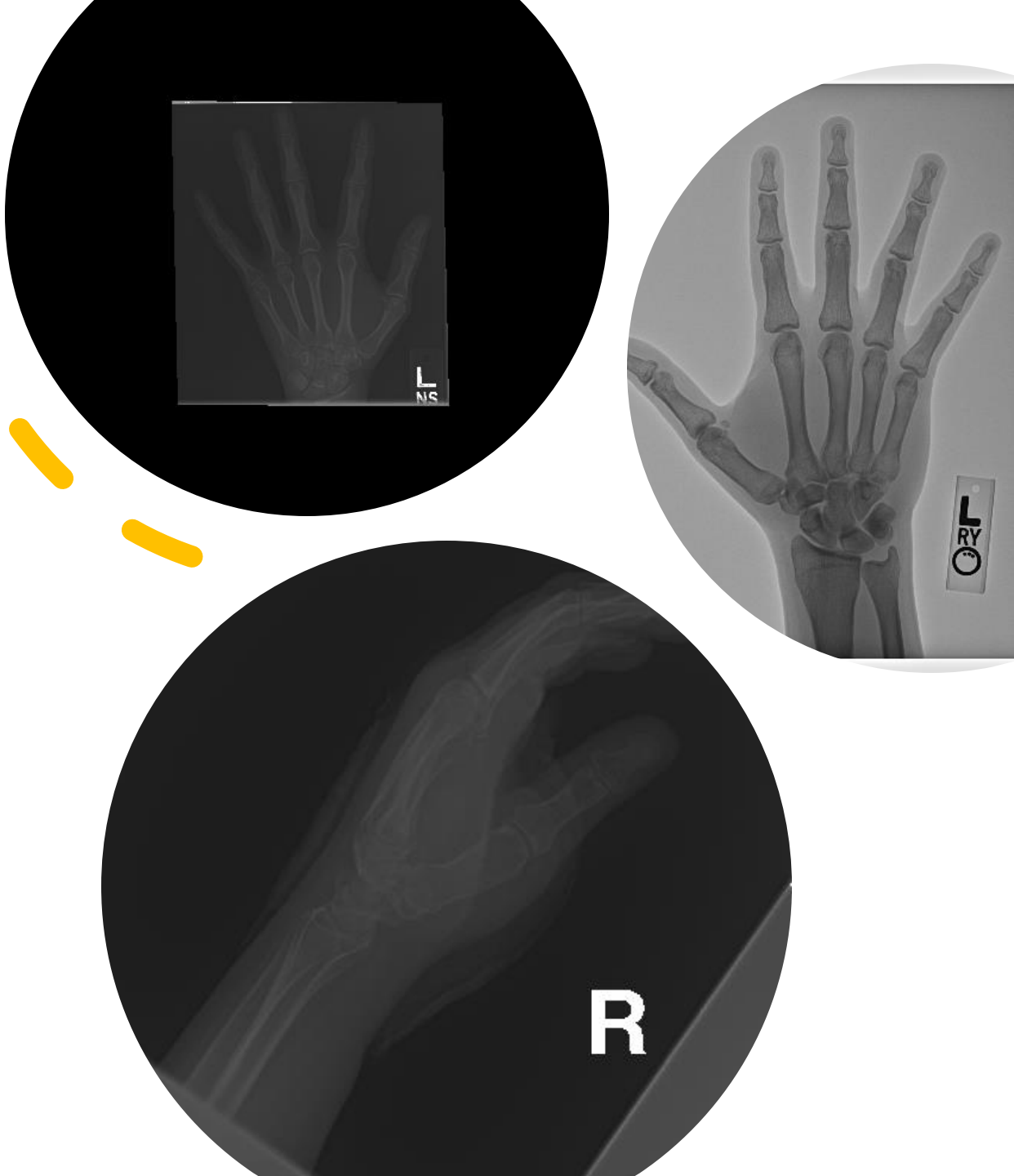
- MURA is one of the largest public radiographic image datasets.
- MURA (**mu**sculoskeletal **ra**diographs) is a large dataset of bone X-rays.
- Total Size: 3.45gb

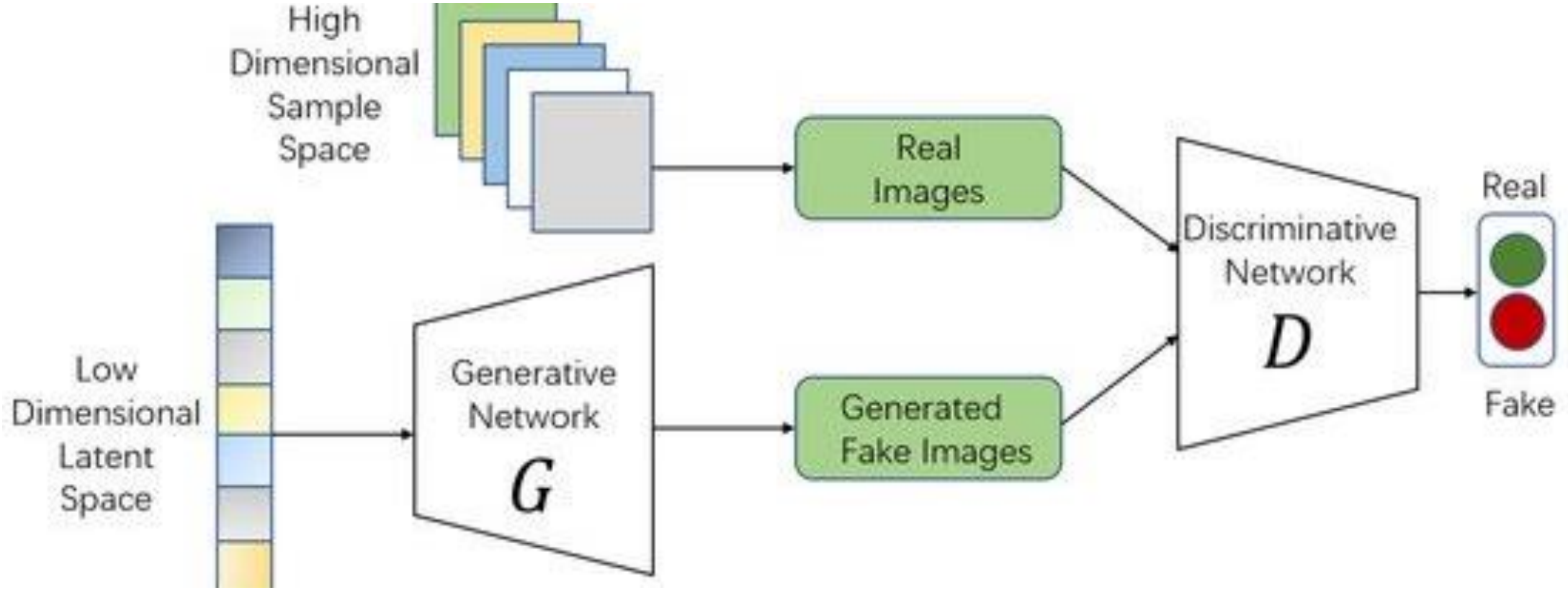
Dataset Samples

- Elbow
  - Positive (2006)
  - Negative (2925)
- Finger
  - Positive (1968)
  - Negative (3138)
- Forearm
  - Positive (661)
  - Negative (1164)
- Hand
  - Positive (1484)
  - Negative (4059)
- Humerus
  - Positive (599)
  - Negative (673)
- Shoulder
  - Positive (4168)
  - Negative (4211)
- Wrist
  - Positive (3987)
  - Negative (5765)

Preprocessing

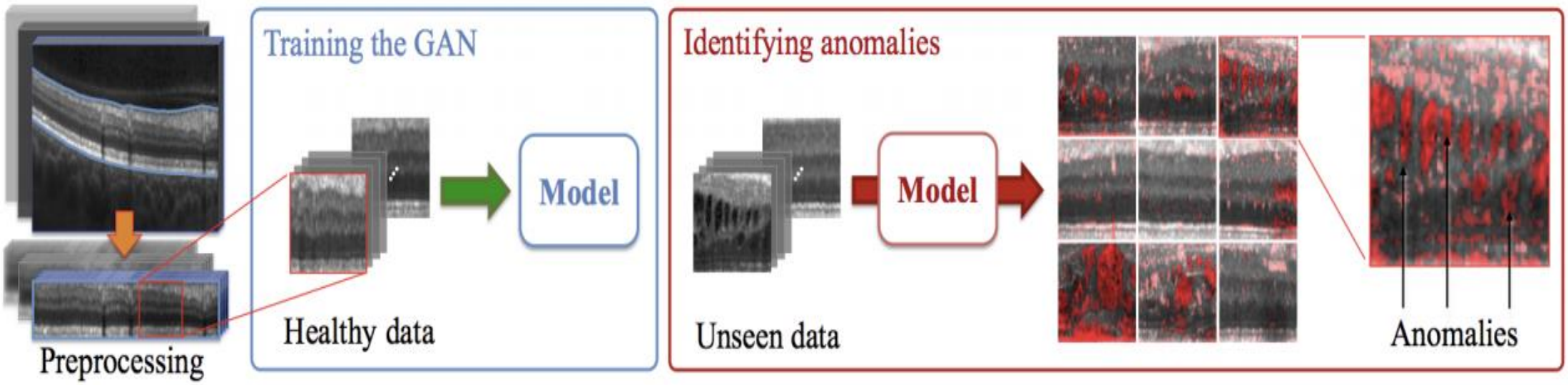
- Crop Image: Using OpenCV, we cropped the image based on where contours were visible
- Resize Image: Reduce the size of image to a size that is much more manageable
- Augmentation
  - Flipping
  - Rotating
  - Zoom in/out





Alpha GAN

- Loss: Binary cross entropy between fake/reconstructed images
- Outlier Score: Mean Square Error

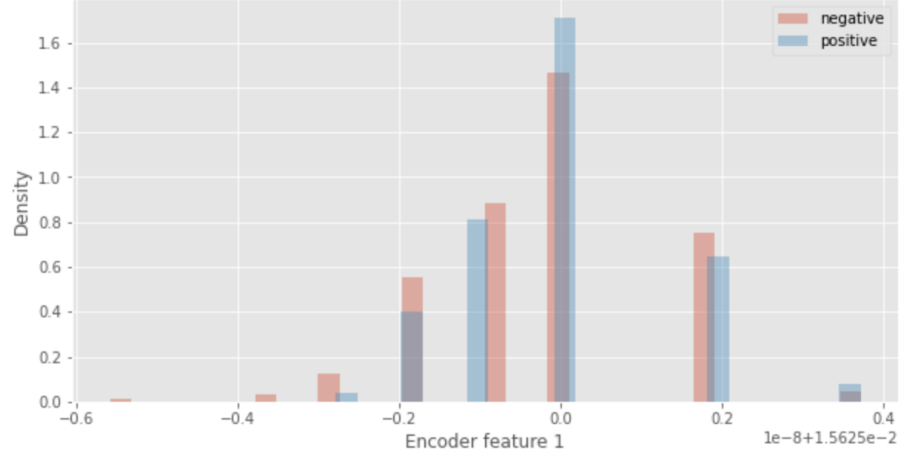


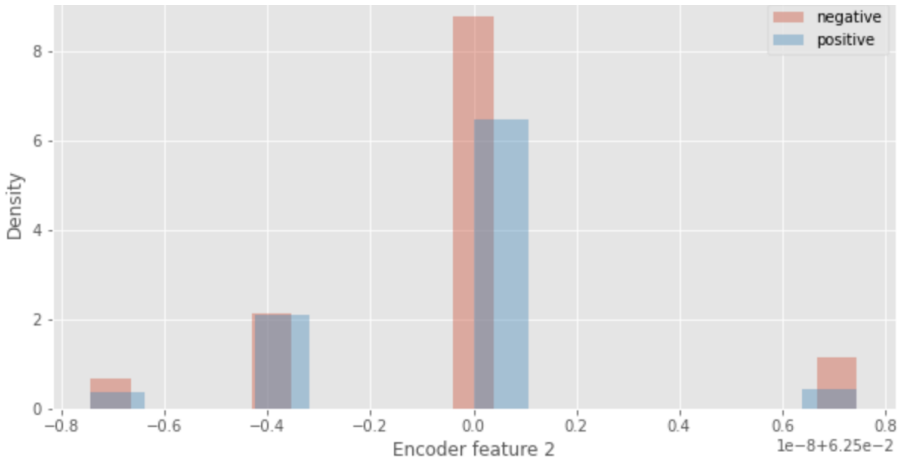
F-AnoGAN

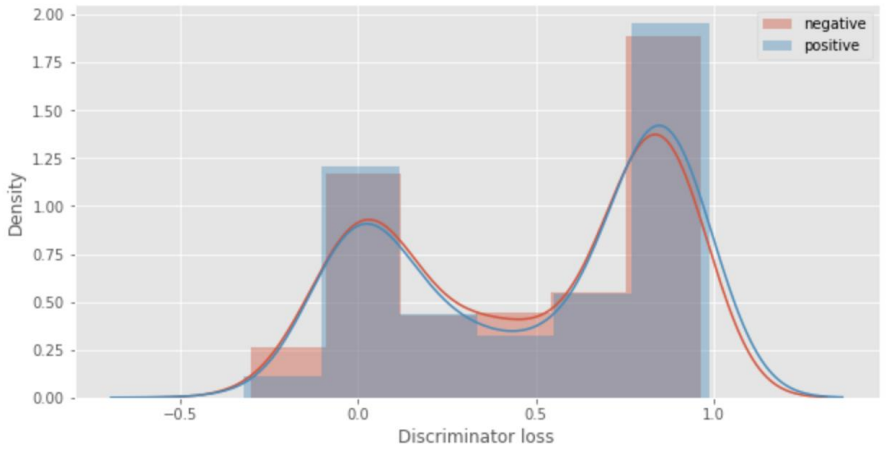
- Loss: Adversarial loss
- Formula for loss = mean(fake_data) - mean(real_data) + lambda_gp * gradient_penalty

Loss Definitions

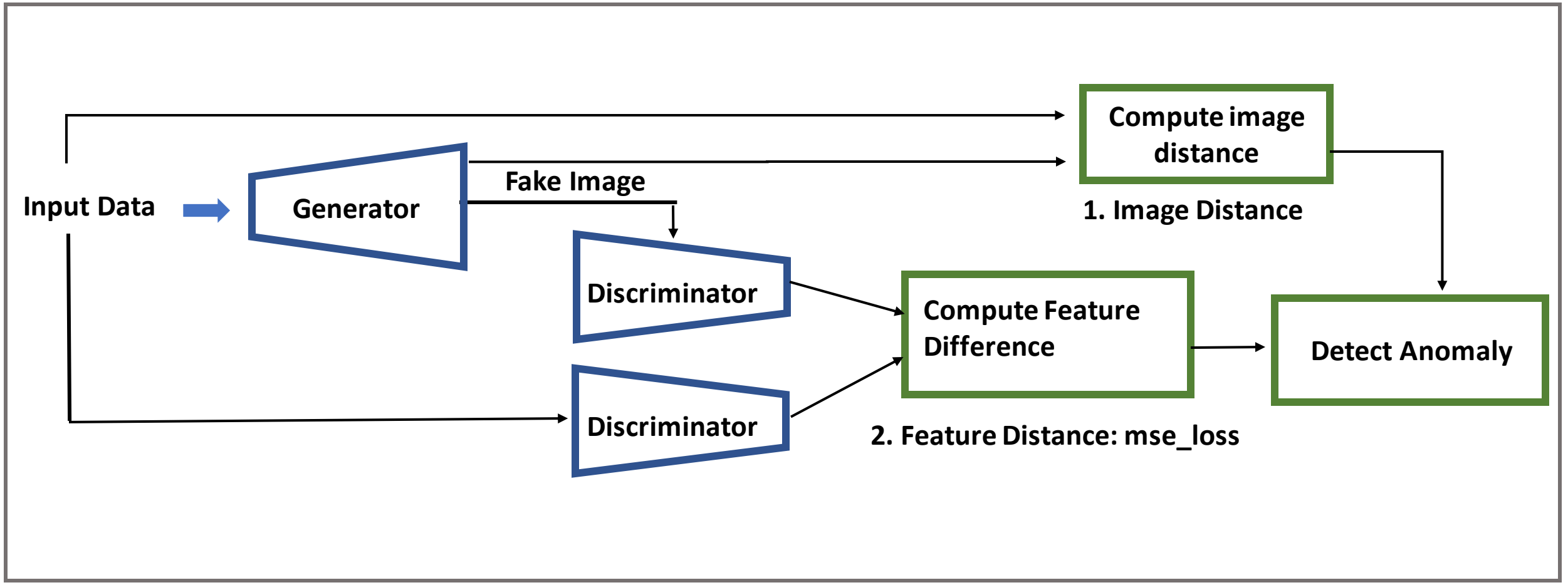
- Reconstruction Loss: Differences between the original and reconstructed images
- Feature Matching Loss: Differences between the encoded features of hidden layers in the encoder and discriminator
- Discriminator Loss: Output of the discriminator







Inference - Architecture



Inference pipeline

Generate Fake images using trained generator model

Get discriminator values for real and fake images

Compute feature distance from the discriminator outputs.

Compute image distance from the real and fake images

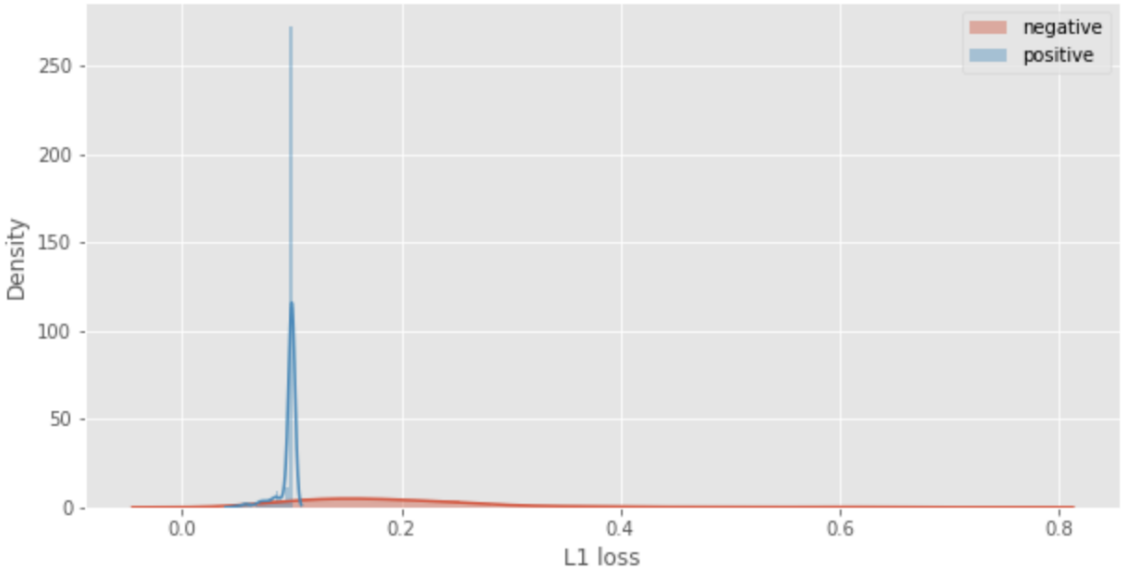
Compute Anomaly score from the distances

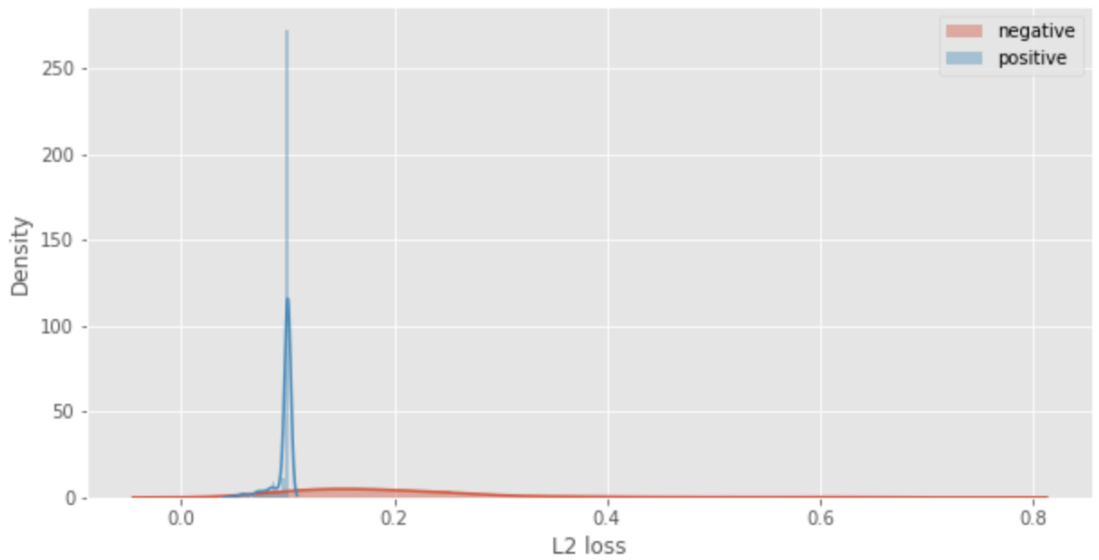
Google Cloud Platform TPU

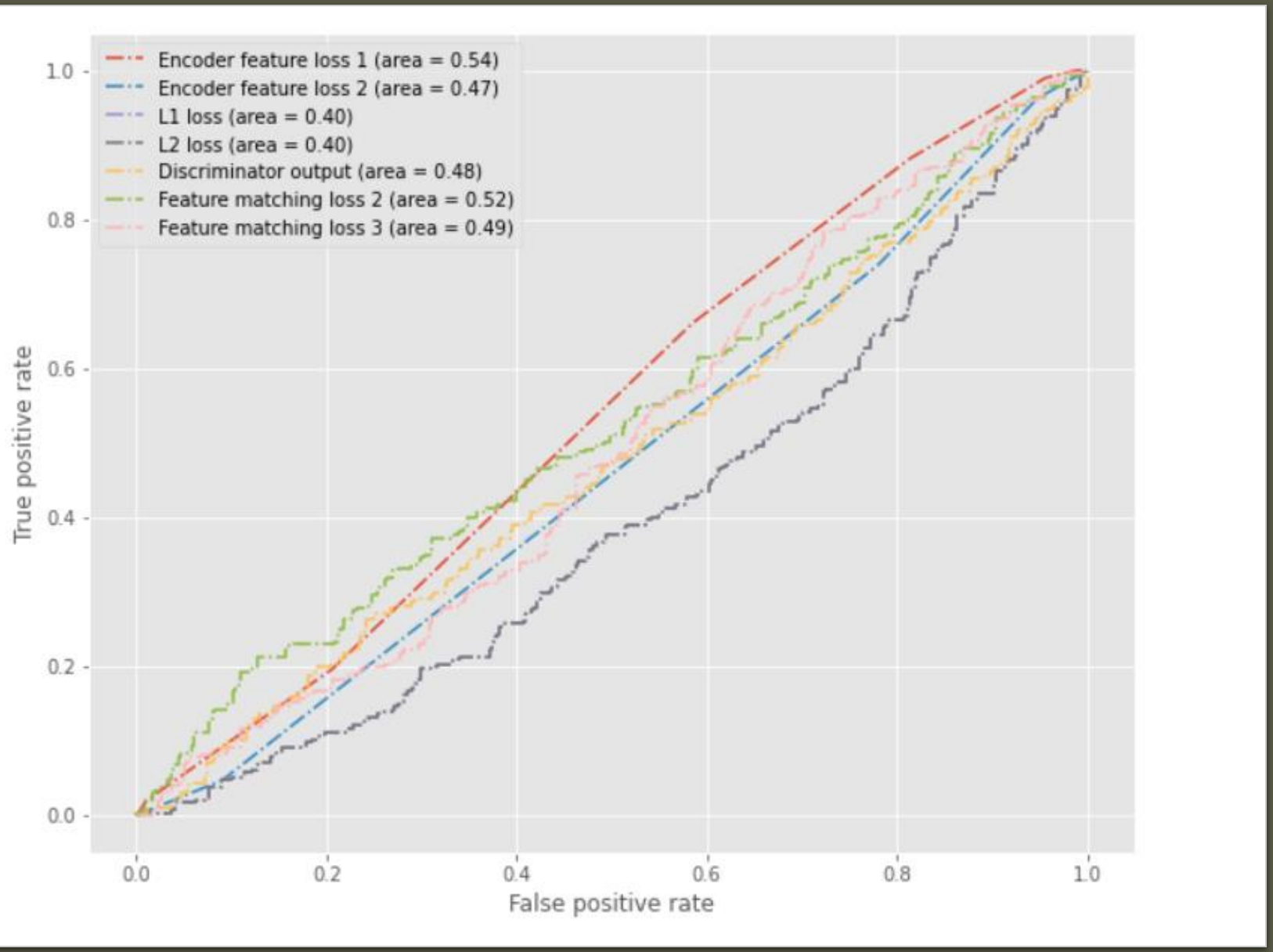
```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to euphoric-diode-296917.
Use “gcloud config set project [PROJECT_ID]” to change to a different project.
pranav_lodha@cloudshell:~ (euphoric-diode-296917)$ export PROJECT_ID=euphoric-diode-296917
pranav_lodha@cloudshell:~ (euphoric-diode-296917)$ gcloud config set project $PROJECT_ID
Updated property [core/project].
pranav_lodha@cloudshell:~ (euphoric-diode-296917)$
```

```
                                                424242,
                                                42424242]},
                      'masked_loss_on_val': True,
                      'z_dim': 100},
 'pipeline': {'adaptive_hist_equilization': False,
              'hist_equalisation': True,
              'normalisation': (0,
                                1),
              'otsu_filter': False},
 'random_seed': [42,
                 4242,
                 424242,
                 42424242],
 'trainable_params': 6594824,
 'z_dim': 100}
==========Epoch [1/50]============
Training: 100% 253/253 [2:36:53<00:00, 37.21s/it]
Loss on last train batch: {'generator loss': 1.079276442527771, 'encoder loss': 1.079276442527771, 'discriminator loss': 1.7739592790603638, 'codiscr
Validation: 100% 181/181 [07:18<00:00,  2.42s/it]
ROC-AUC on mse: 0.42671065553921217. APS on mse: 0.6308430390719538. Mean mse: 0.1523047387599945
ROC-AUC on mse_top_k: 0.48080508493578245. APS on mse_top_k: 0.65338154322373. Mean mse_top_k: 0.7781074643135071
ROC-AUC on proba: 0.5187512332280978. APS on proba: 0.6617365774192119. Mean proba: 0.7584964632987976
ROC-AUC on coproba: 0.4992466097438473. APS on coproba: 0.6893208747984073. Mean coproba: 0.37139028310775757
ROC-AUC on proba_coproba: 0.5187515135072109. APS on proba_coproba: 0.6617365774192119. Mean proba_coproba: 0.5649433732032776
==========Epoch [2/50]============
Training:   4% 11/253 [06:50<2:43:50, 40.62s/it]
```

Metrics visualization for AlphaGAN







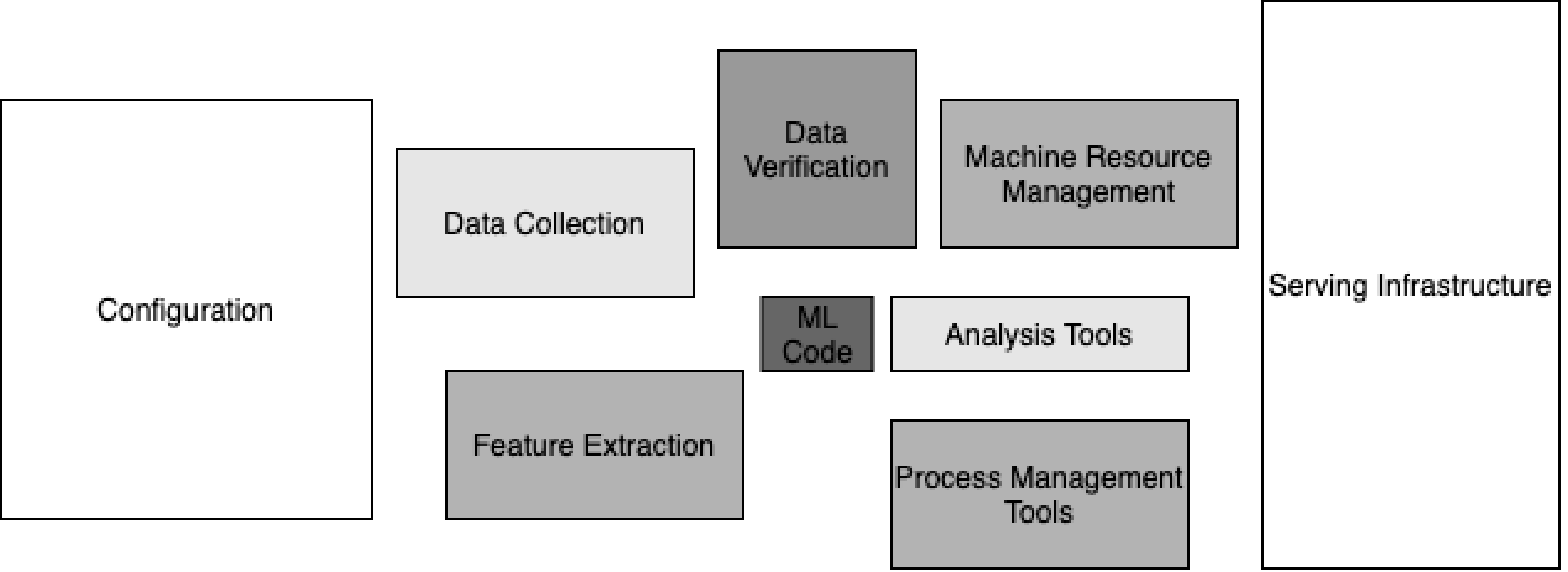
True Positive Rate
vs.
False Positive Rate

AnoGAN Metrics

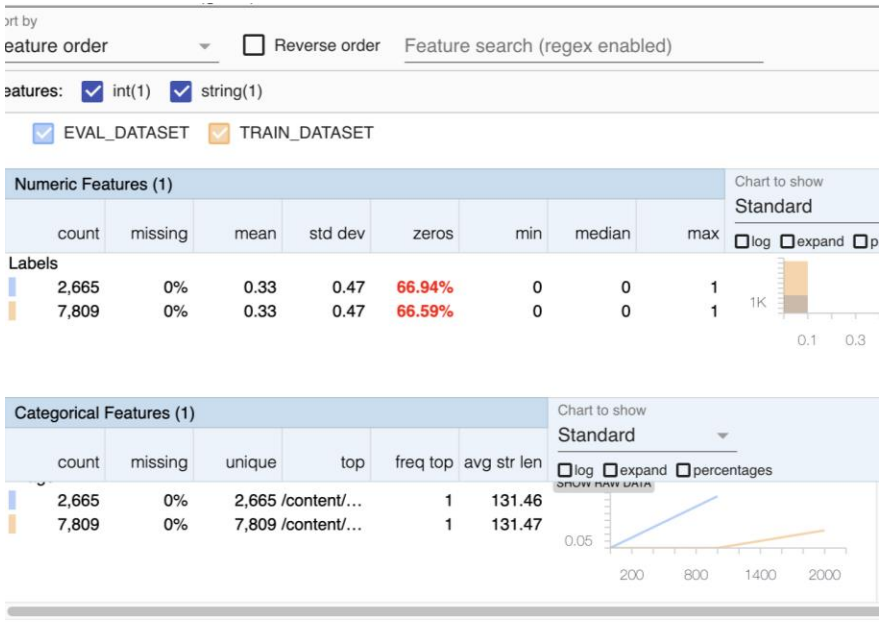
No.	Category	ROC- AUC
1	ELBOW	0.88
2	FINGER	0.92
3	FOREARM	0.93
4	HAND	0.78
5	HUMERUS	0.91
6	SHOULDER	0.89
7	WRIST	0.97

Comparison: f-AnoGAN vs. AlphaGAN

No.	Category	F-AnoGAN	AlphaGAN
1	ELBOW	0.88	0.62
2	FINGER	0.92	0.65
3	FOREARM	0.93	0.69
4	HAND	0.78	0.58
5	HUMERUS	0.91	0.60
6	SHOULDER	0.89	0.65
7	WRIST	0.97	0.74



Integration Pipeline



'f0f4c4d31d0

tatisticsGen at 0x7f0f4a3cdda0

xamples'] ▼**Channel** of type **'Examples'** (1 artifact) at 0x7f0f4a3e2c88

.type_name Examples

._artifacts **[0]** ▶**Artifact** of type **'Examples'** (uri: /tmp/tfx-interactive-2
06T22_08_31.009649-8e9oc3yc/ImportExampleGen/e
0x7f0f4c4769b0

tatistics'] ▼**Channel** of type **'ExampleStatistics'** (1 artifact) at 0x7f0f4a399e48

.type_name ExampleStatistics

._artifacts **[0]** ▶**Artifact** of type **'ExampleStatistics'** (uri: /tmp/tfx-inte
06T22_08_31.009649-8e9oc3yc/StatisticsGen/statistics

```
14] context.run(example_gen)
```

▼**ExecutionResult** at 0x7f0f4a38e240

.execution_id 5

.component ▶**ImportExampleGen** at 0x7f0f4a3e2a20

.component.inputs **['input']** ▶**Channel** of type **'ExternalArtifact'** (1 artifact) at 0x7f0f4a3e2e80

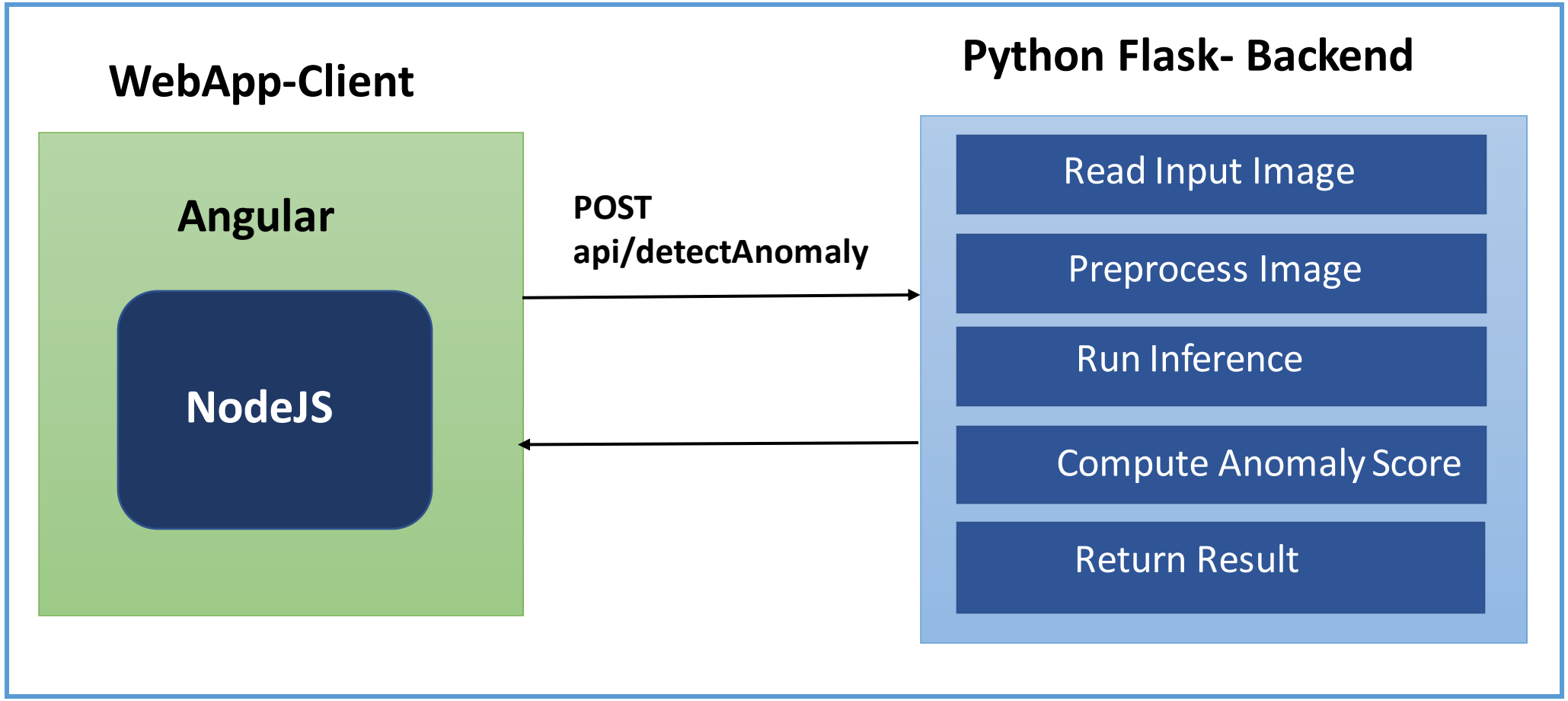
.component.outputs **['examples']** ▼**Channel** of type **'Examples'** (1 artifact) at 0x7f0f4a3e2c88

.type_name Examples

._artifacts **[0]** ▶**Artifact** of type **'Examples'** (uri: /tmp/tfx-interactive-2020-06T22_08_31.009649-8e9oc3yc/ImportExampleGen/examp 0x7f0f4c4769b0

TensorFlow Extended (TFX)

Anomaly Detection Web App. Integration



Results Summary

We learnt how GAN can be applied to anomaly detection

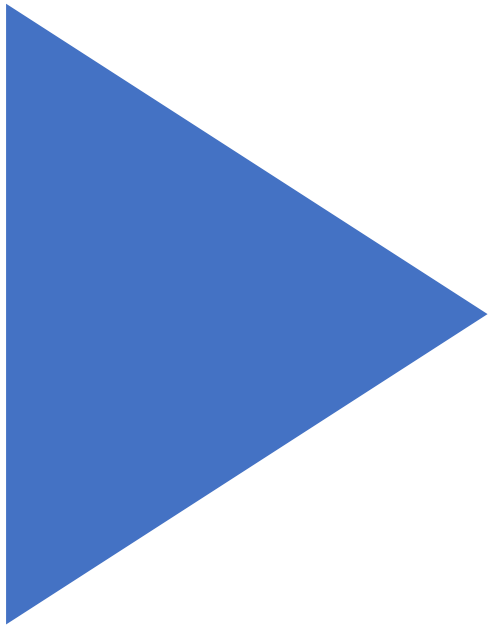
We trained two GANs end to end with mura dataset

We compared the results of the two GANs

Anomaly GAN performance is better compared to AlphaGAN

Delta

- Training f-AnoGAN with Mura dataset
- Metrics comparison of AlphaGAN and f-AnoGAN on X-ray images
- TFX



Demo

Reference

- F-AnoGAN - https://arxiv.org/pdf/1703.05921v1.pdf
- AlphaGAN - https://arxiv.org/pdf/1807.10088.pdf

Thank you