Deep Learning Generative Models in Wireless Networks

Wireless AI Innovation @ Verizon (WAIV)

March 2019

WAIV’s deep learning pipeline

Deep Learning for Time Series

- 2017: ideation, research
- 2018: prototype, GTC talk
- 2019: going into production

Anomaly Detection

- 2018: ideation, research
- 2019: prototype, GTC talk

KPI Generation

- 2019: ideation, research

The use case

Network anomalies come in different flavors

Poor performance

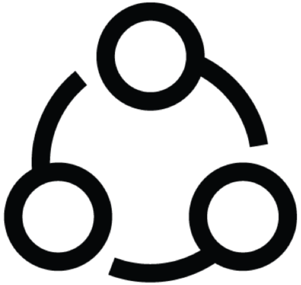
- Sudden performance shifts
- Exceeding thresholds
- Generating alarms

"Broken Sites"

Exhausting capacities

- Reaching operational limits
- Exceeding thresholds
- Performing normally

"Sites Needing Attention"



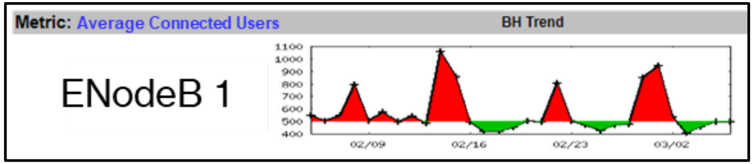
Relative performance

- Clusters performing unusually well
- Neighboring clusters performing worse
- No obvious causes

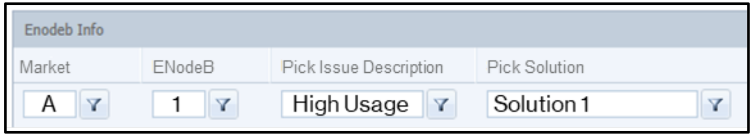
"Areas Requiring Analysis"

Network anomaly detection today

10’s of thousands of sites. 100’s of thousands of carriers. Millions of metrics. Multiple tools to navigate.



Tool 1



Tool 2

Your Site ENodeB 1 continues to exceed Average Connected Users. Access Tool 1 to view more information and consider building a solution in Tool 2. Tool 3 is available for further analysis.

Alert 1

Areas where today's approach can improve

Reduce time to detect anomalies

Reduce the number of tools needed to detect anomalies

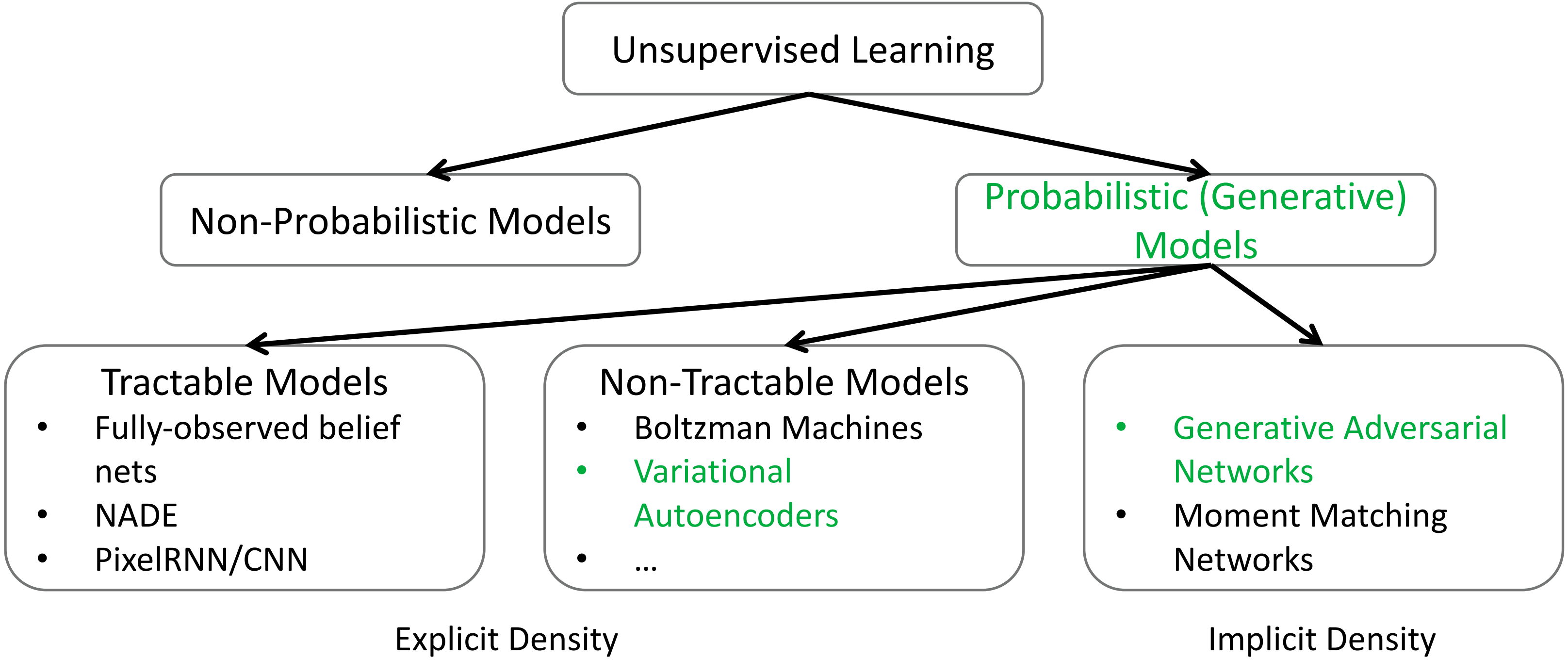
Enable detection based on more than just hard thresholds

Take advantage of all possible data correlations

Automate steps, especially when detecting complex anomalies

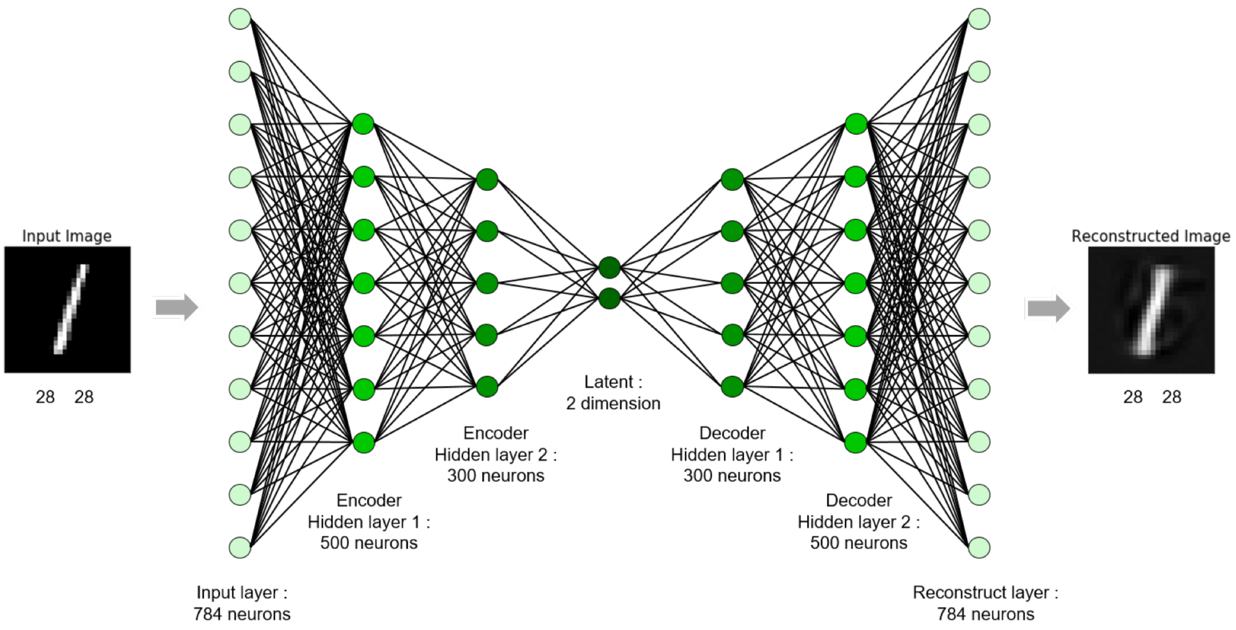
The generative modeling approach

Hierarchy of unsupervised learning



Using an autoencoder for anomaly detection

The embedded/latent space will <hopefully> contain information that is useful for anomaly detection



https://i-systems.github.io/HSE545/machine%20learning%20all/Workshop/Hanwha/Lecture/image_files/AE_arch2.png

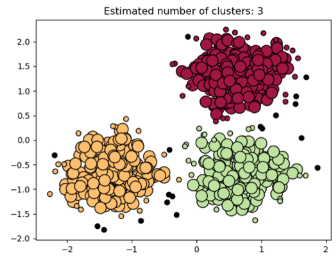
Generative modeling architecture for anomaly detection

Initial goal is to generate clusters of sites that could be anomalies

Then develop a supervised learning model to automatically identify clusters that contain verified anomalies

Network performance data

→ Train Generative Model → Produce Reconstructed Output → Find and Cluster Anomalies → SME Analysis



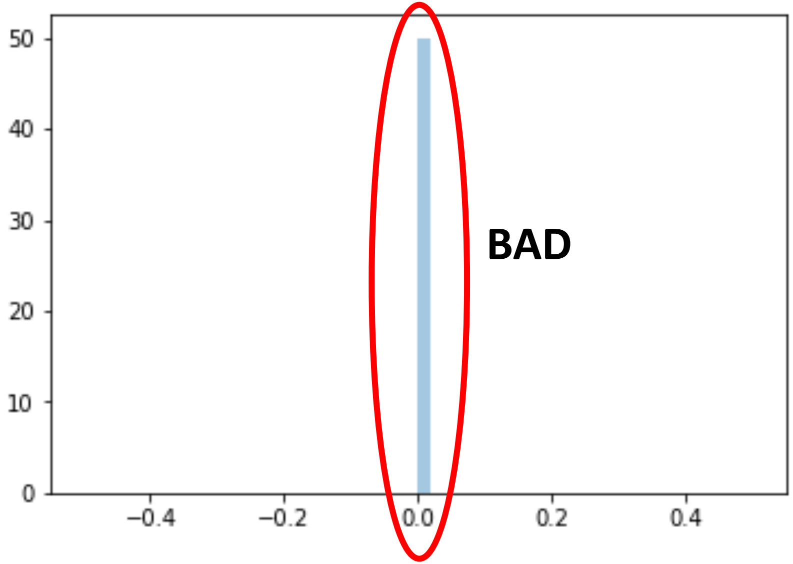
Labeled anomaly clusters

https://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html#sphx-glr-auto-examples-cluster-plot-dbscan-py

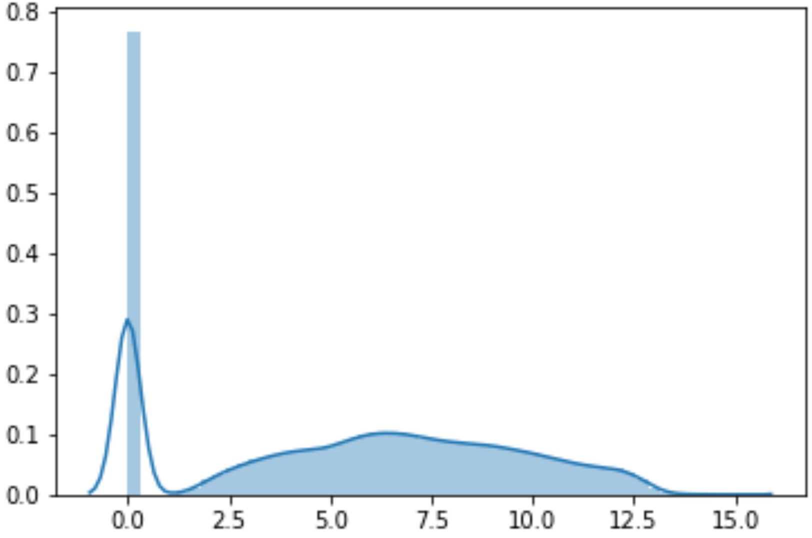
Training the model

Hyperparameter tuning – very important

Dimension 1 Distribution

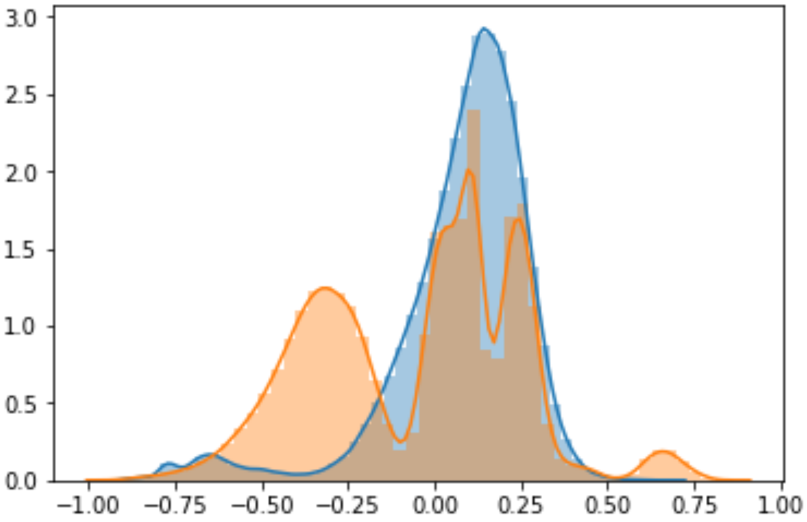


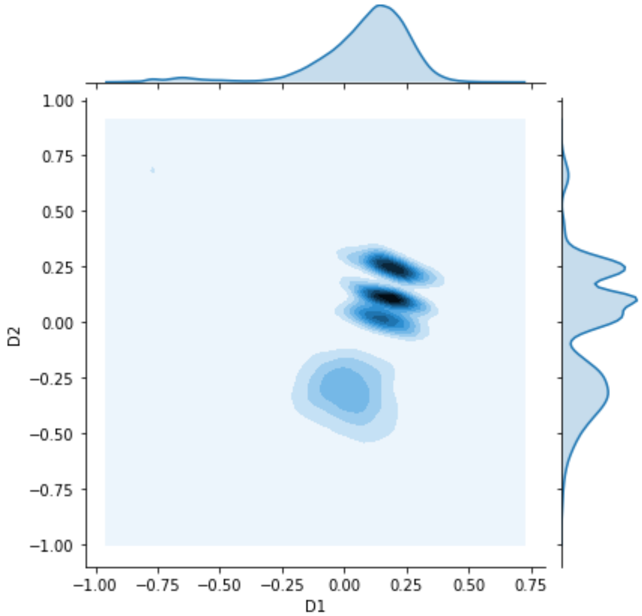
Dimension 2 Distribution



verizon

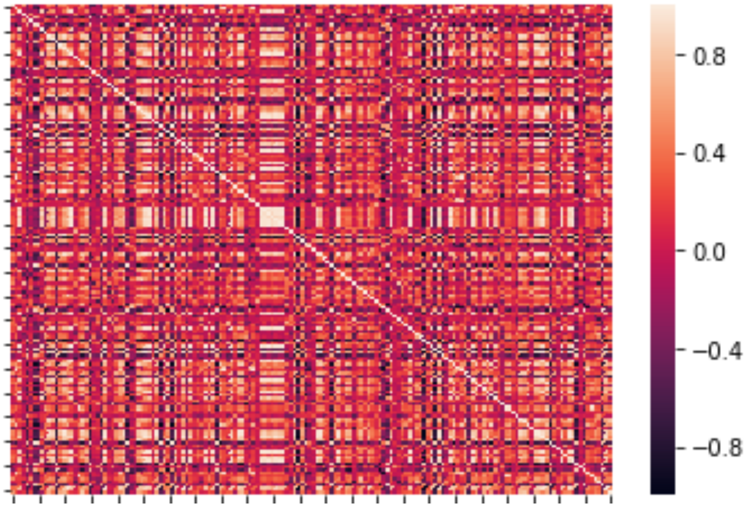
Model outputs when embedded space is 2-dimensional



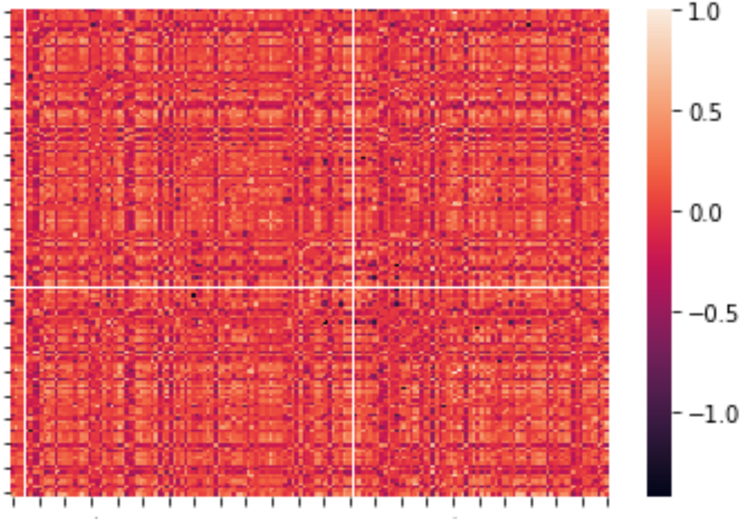


Correlation heatmap for 2-dimensional embedded space

Original Space



Reconstructed Space

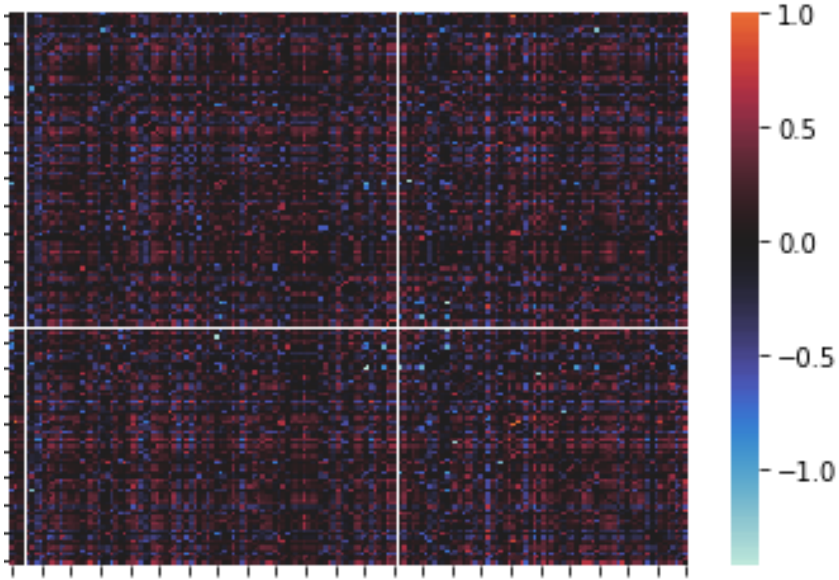


Heatmap of delta between original and reconstructed correlations

Black entries mean the original and reconstructed spaces have similar correlations

We use correlations as a proxy for model quality

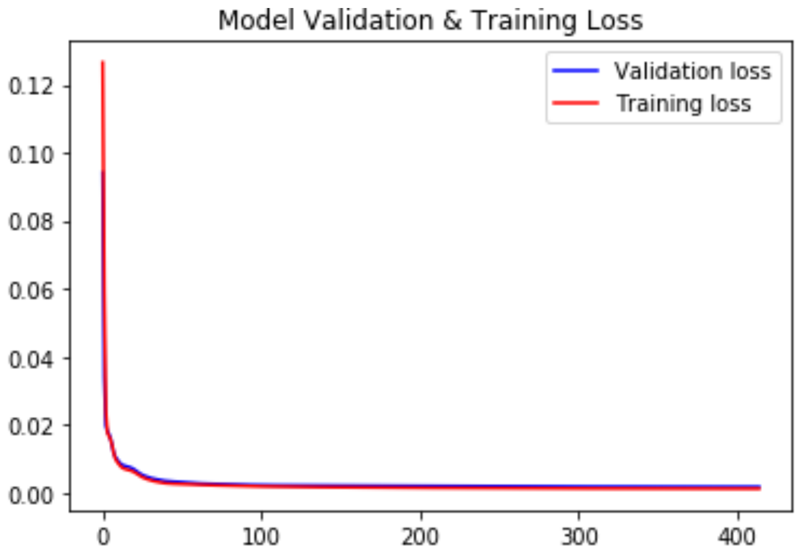
There is lots of red and blue in the diagram, so the <u>model is NOT GOOD</u>



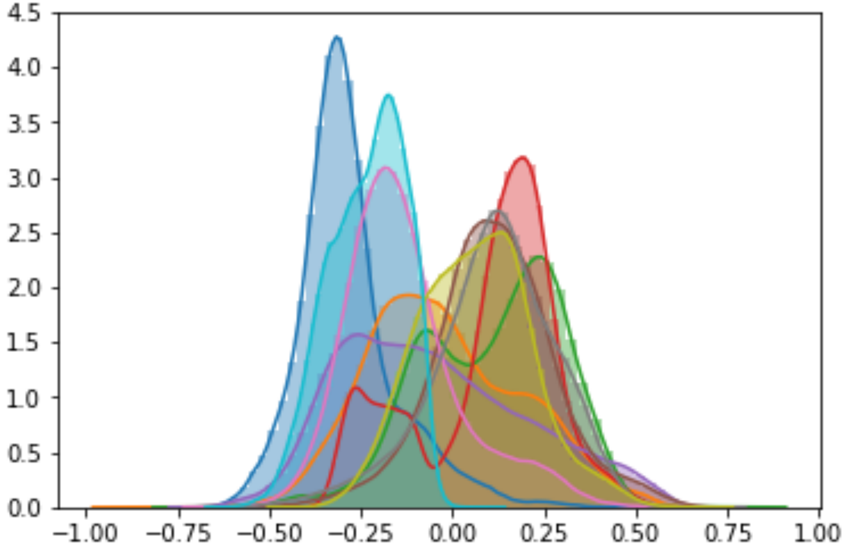
Learning Curve for 10-dimensional embedded space

Training and validation loss are almost identical

This means the model is learning the dataset very well



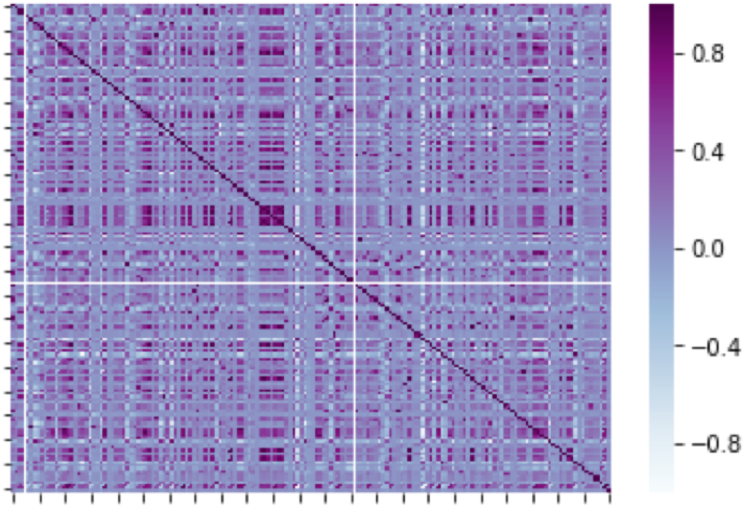
Model outputs when embedded space is 10-dimensional



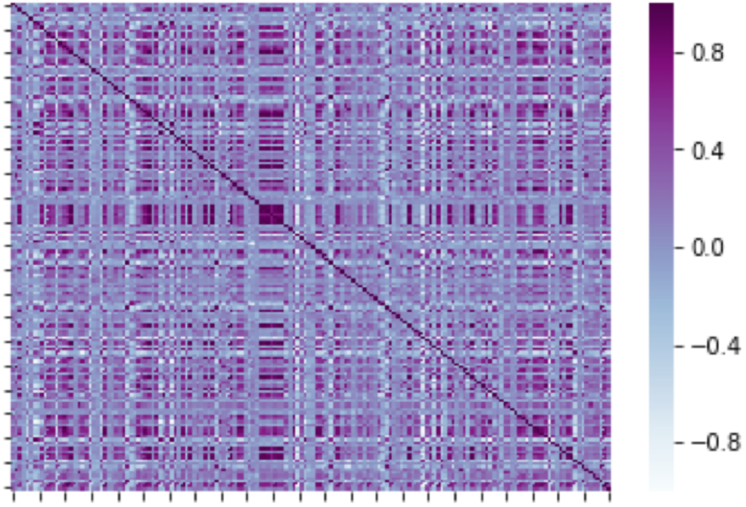
ENODEB	Dim1	Dim10	Dim2	Dim3	Dim4	Dim5	Dim6	Dim7	Dim8	Dim9
1	-0.26	-0.08	0.00	0.23	0.21	-0.11	-0.03	-0.22	0.14	0.12
2	-0.37	-0.10	0.36	-0.07	-0.10	-0.34	0.35	-0.34	0.34	-0.11
3	-0.34	-0.12	-0.07	0.20	0.21	0.07	0.12	-0.20	0.02	0.07
4	-0.28	-0.06	0.28	-0.13	-0.11	-0.35	0.28	-0.15	0.27	-0.05
5	-0.09	-0.32	-0.08	0.17	0.28	-0.43	0.17	-0.31	0.09	0.11
6	-0.33	-0.09	0.23	-0.03	-0.10	-0.31	0.28	-0.25	0.37	-0.12
7	-0.31	-0.08	0.12	-0.06	-0.13	-0.25	0.21	-0.18	0.29	-0.18
8	-0.21	-0.13	0.08	0.25	0.33	-0.15	0.10	-0.24	0.18	0.24
9	-0.03	-0.37	-0.09	0.32	0.29	-0.30	0.23	-0.34	0.23	0.06
10	-0.08	-0.35	-0.13	0.19	0.29	-0.37	0.14	-0.29	0.23	0.08
11	-0.14	-0.20	-0.35	-0.05	0.16	0.04	0.10	-0.28	-0.01	-0.02
12	-0.29	-0.17	-0.08	0.26	0.26	0.02	0.03	-0.27	0.12	0.07
13	-0.33	-0.04	0.09	-0.06	-0.16	-0.29	0.25	-0.14	0.21	-0.13
14	-0.32	-0.06	0.23	-0.04	-0.12	-0.32	0.34	-0.19	0.31	-0.05
15	-0.31	-0.10	-0.08	0.28	0.23	-0.07	0.07	-0.22	0.07	0.16
16	-0.25	-0.16	-0.08	0.24	0.28	0.08	0.16	-0.13	0.16	0.14

Correlation heatmap for 10-dimensional embedded space

Original Space



Reconstructed Space



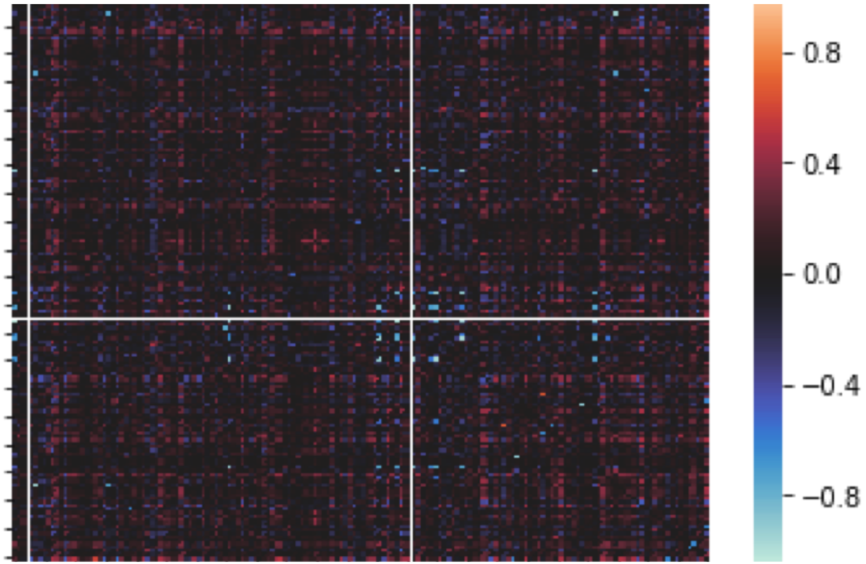
Structural Similarity Index 0.92

Heatmap of delta between original and reconstructed correlations

Black entries mean the original and reconstructed spaces have similar correlations

We use correlations as a proxy for model quality

There's lots of black in the diagram, so <u>the model is good</u>



Final autoencoder model architecture

Key Features

- 163 input features
- 9 hidden layers
- 10-dimensional embedded space
- Mean Squared Error (MSE) loss
- Adam (RMSprop + Momentum) optimizer

Encoder

Decoder

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 163)	0
dense (Dense)	(None, 200)	32800
dense_1 (Dense)	(None, 100)	20100
dense_2 (Dense)	(None, 40)	4040
dense_3 (Dense)	(None, 20)	820
dense_4 (Dense)	(None, 10)	210
dense_5 (Dense)	(None, 20)	220
dense_6 (Dense)	(None, 40)	840
dense_7 (Dense)	(None, 100)	4100
dense_8 (Dense)	(None, 200)	20200
dense_9 (Dense)	(None, 163)	32763

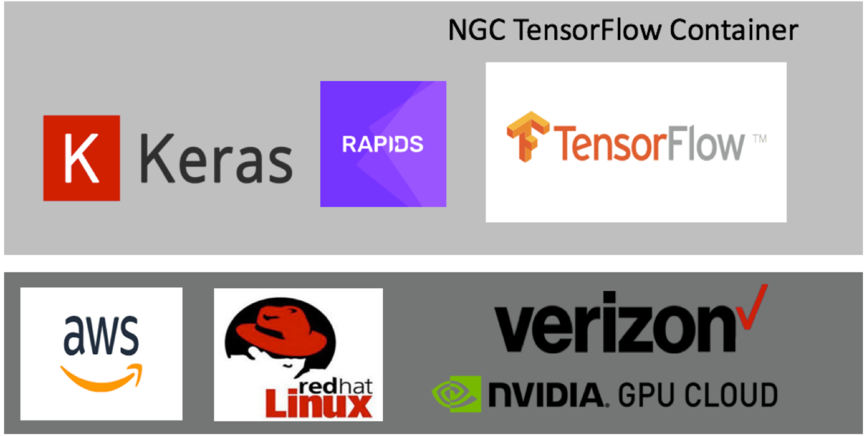
Autoencoder implementation

NGC TensorFlow container optimized for Nvidia Volta GPU

- RAPIDS
- Keras
- TensorFlow

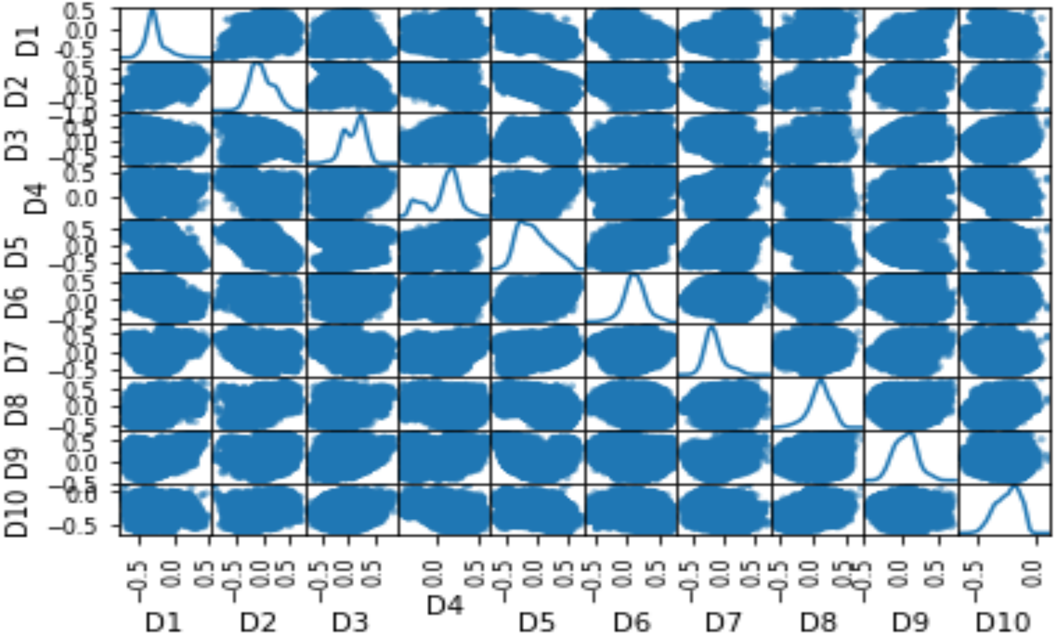
AWS P3 Instance

- Verizon RHEL AMI

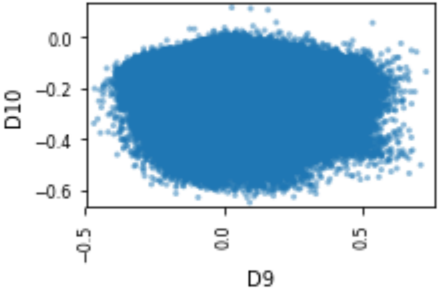


Using the model outputs

Looking for anomalies in the embedded space



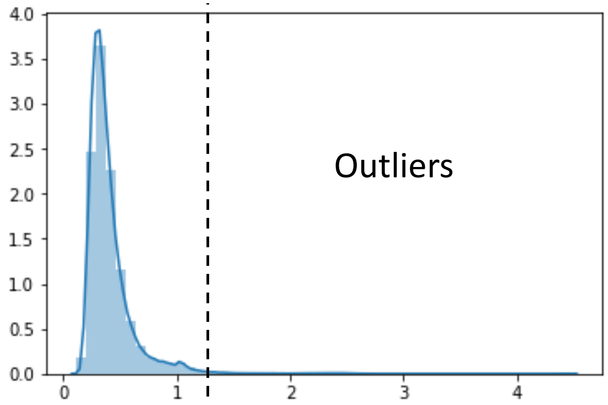
Kernel Density Estimation duration:
0:10:40.224371



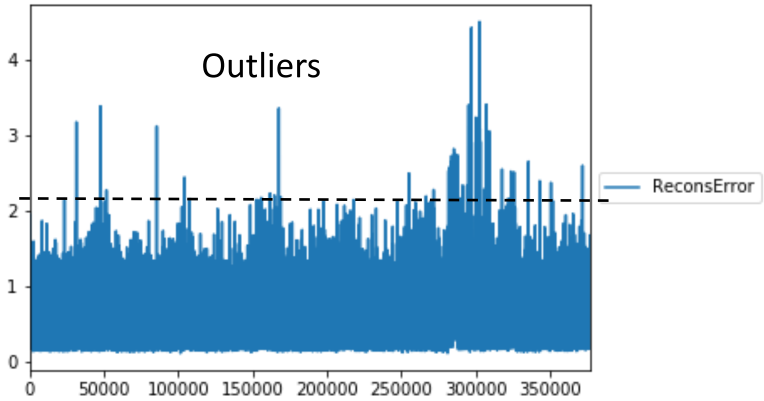
Looking for anomalies based on reconstructed outputs

Can we perform anomaly detection by identifying outliers based on reconstruction error?

Kernel Density Estimation



Profile Plot

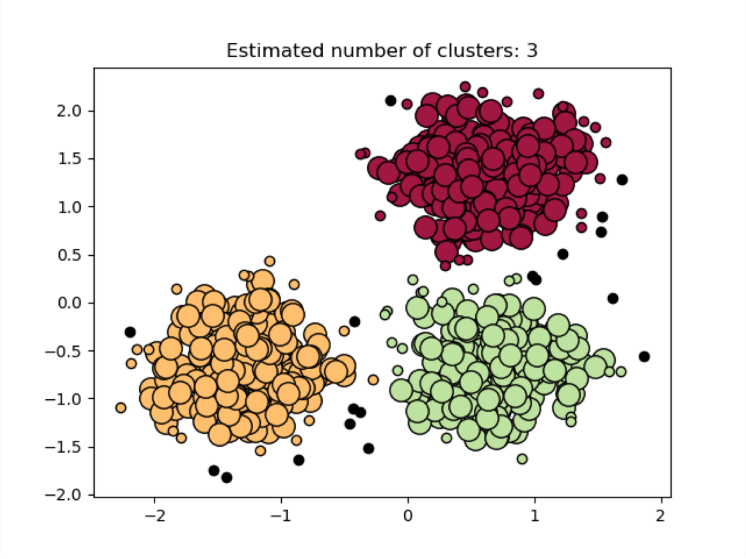


Clustering the reconstruction error outliers

Too many outliers to analyze individually

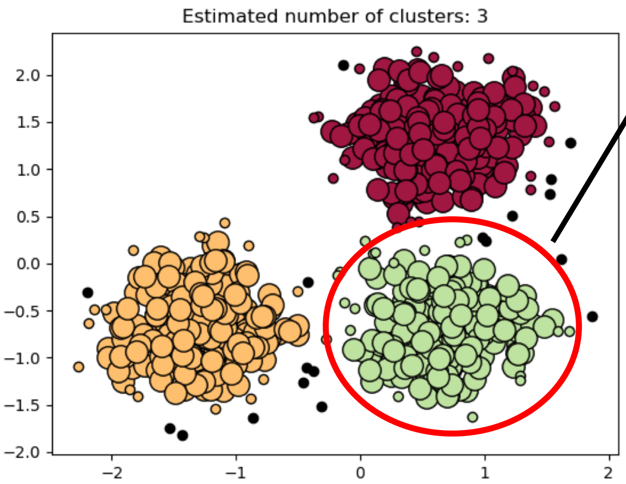
Cluster the outliers into groups to ease analysis

Use DBSCAN to find clusters and estimate the number of clusters



https://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html#sphx-glr-auto-examples-cluster-plot-dbscan-py

Analyzing the clustering results



Network engineers manually inspected performance data from cell sites in each cluster

They found that a cluster that included all high-usage sites from the input data

This means the model automatically learned to detect an anomaly that we train our new engineers to find

https://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html#sphx-glr-auto-examples-cluster-plot-dbscan-py

What’s next

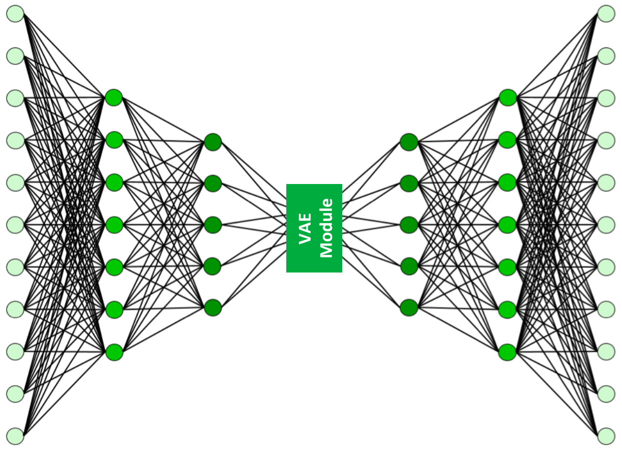
Mature the anomaly detection capability

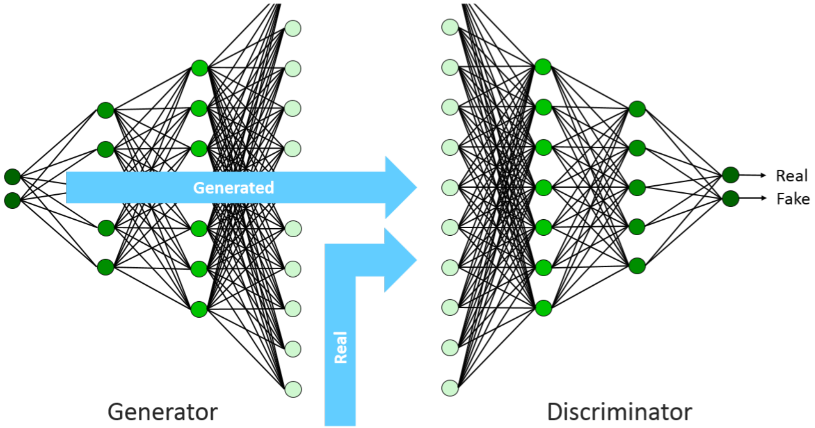
Simple Anomalies
- Simple outliers (e.g., High-usage sites)

Known Complex Anomalies
- Complex multivariate anomalies (e.g., Passive Intermodulation)

Unknown Complex Anomalies
- New issues not seen before (e.g., SW Bug on new counter)

Try more generative models





https://i-systems.github.io/HSE545/machine%20learning%20all/Workshop/Hanwha/Lecture/image_files/AE_arch2.png

https://i-systems.github.io/HSE545/machine%20learning%20all/12%20Deep%20learning/image_files/my_GAN.png

Variational autoencoder model architecture

Layer (type)	Output Shape	Param #
encoder_input (InputLayer)	(None, 163)	0
vae_latent_sample (Model)	(None, 10)	58190
vae_output_sample (Model)	(None, 163)	90996

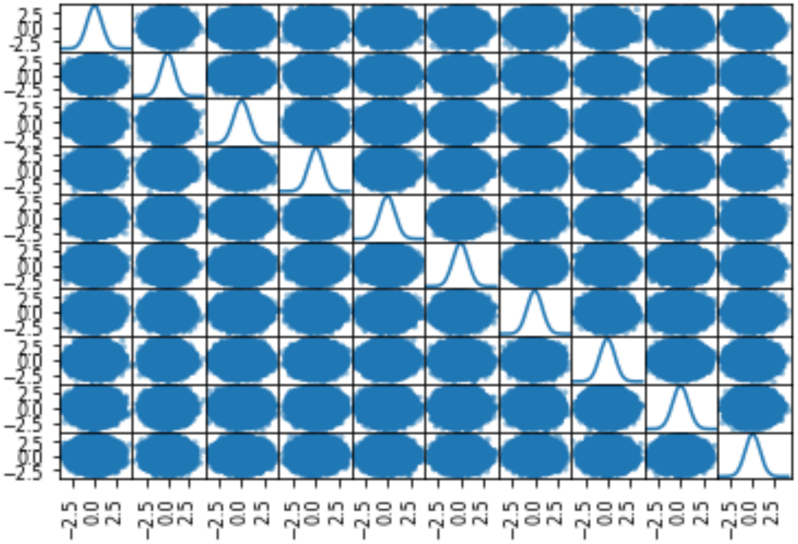
Layer (type)	Output Shape	Param #
encoder_input (InputLayer)	(None, 163)	0
encoder_1 (Dense)	(None, 200)	32800
encoder_2 (Dense)	(None, 100)	20100
encoder_3 (Dense)	(None, 40)	4040
encoder_4 (Dense)	(None, 20)	820
encoder_5 (Dense)	(None, 10)	210
latent_mean (Dense)	(None, 10)	110
latent_variance (Dense)	(None, 10)	110

Layer (type)	Output Shape	Param #
input_latent_values (InputLayer	(None, 10)	0
decoder_1 (Dense)	(None, 10)	110
decoder_2 (Dense)	(None, 20)	220
decoder_3 (Dense)	(None, 40)	840
decoder_4 (Dense)	(None, 100)	4100
decoder_5 (Dense)	(None, 200)	20200
output_mean (Dense)	(None, 163)	32763
output_variance (Dense)	(None, 163)	32763

Generated embedded space from variational autoencoder

How can this be used?

Come back next year to learn more!



Kernel Density Estimation of the latent space generated by the variational autoencoder

Future work

Model improvements to increase anomaly detection rate

Optimize code for the Rapids platform

Modular anomaly detection framework

Combined KPI forecast and anomaly detection frameworks

Use variational autoencoders and GANs to improve network understanding

Thank you.

verizon

Model evaluation guideline