Anomaly Detection

5/21

Outline

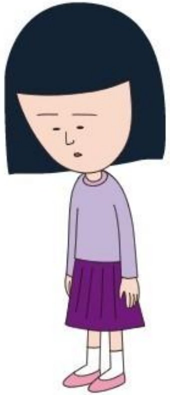
- What is Anomaly Detection
- Classic Method
  - With Classifier
  - GMM (Gaussian Mixture Model)
  - Auto-Encoder
  - PCA
  - Isolation Forest
  - Summary
- Anomaly Detection on image
  - AnoGAN
  - EGBAD
  - GANomaly
  - Summary
- Anomaly Detection on Audio
  - GMGAN

What is Anomaly Detection

What is Anomaly

- Training Data





Anomaly

Anomaly

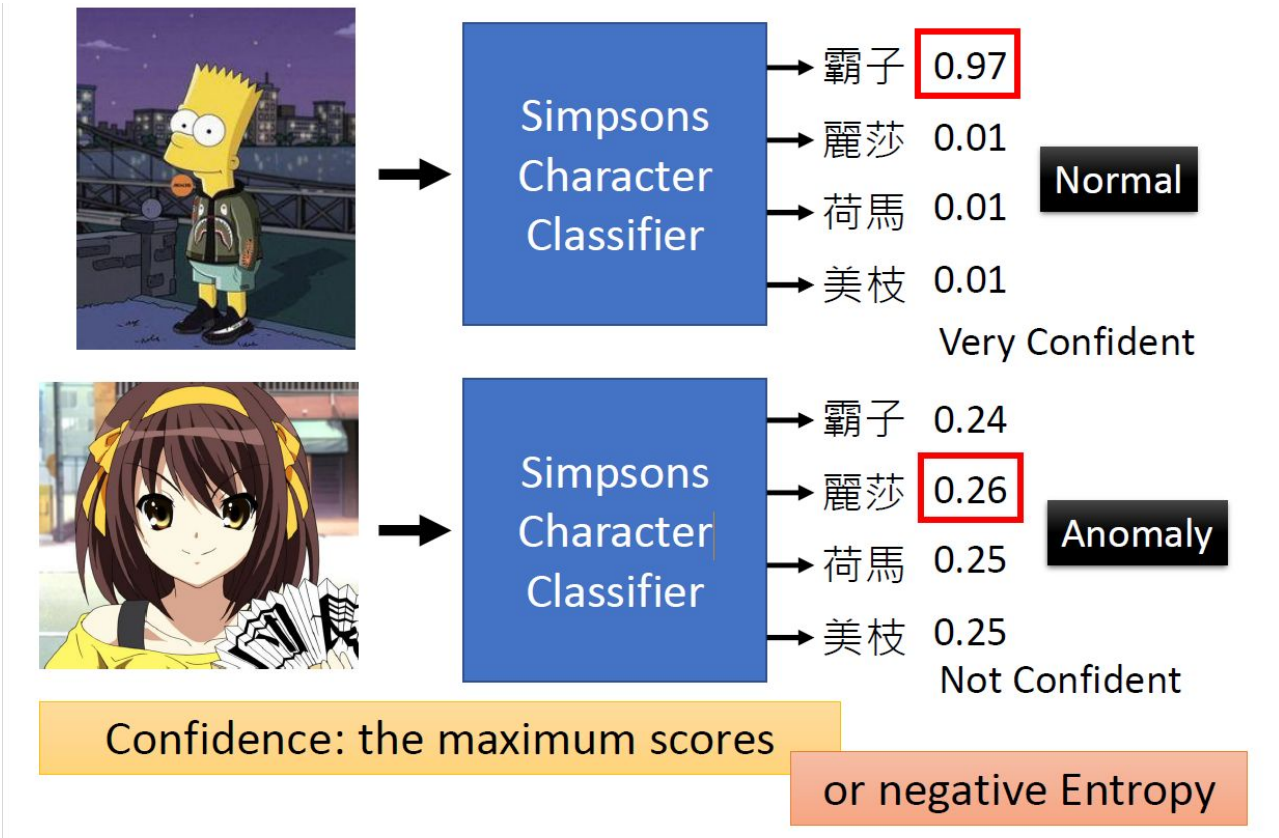
Classic Method

With Classifier

Anomaly Detection:

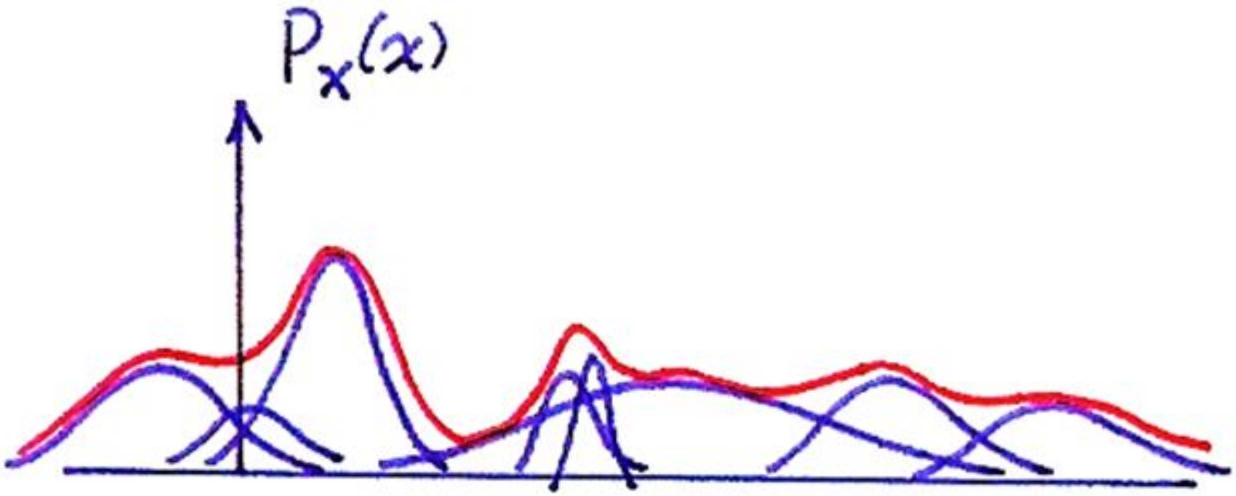
$$f(x) = \begin{cases} normal, & c(x) > \lambda \\ anomaly, & c(x) \leq \lambda \end{cases}$$

With Classifier



GMM (Gaussian Mixture Model)

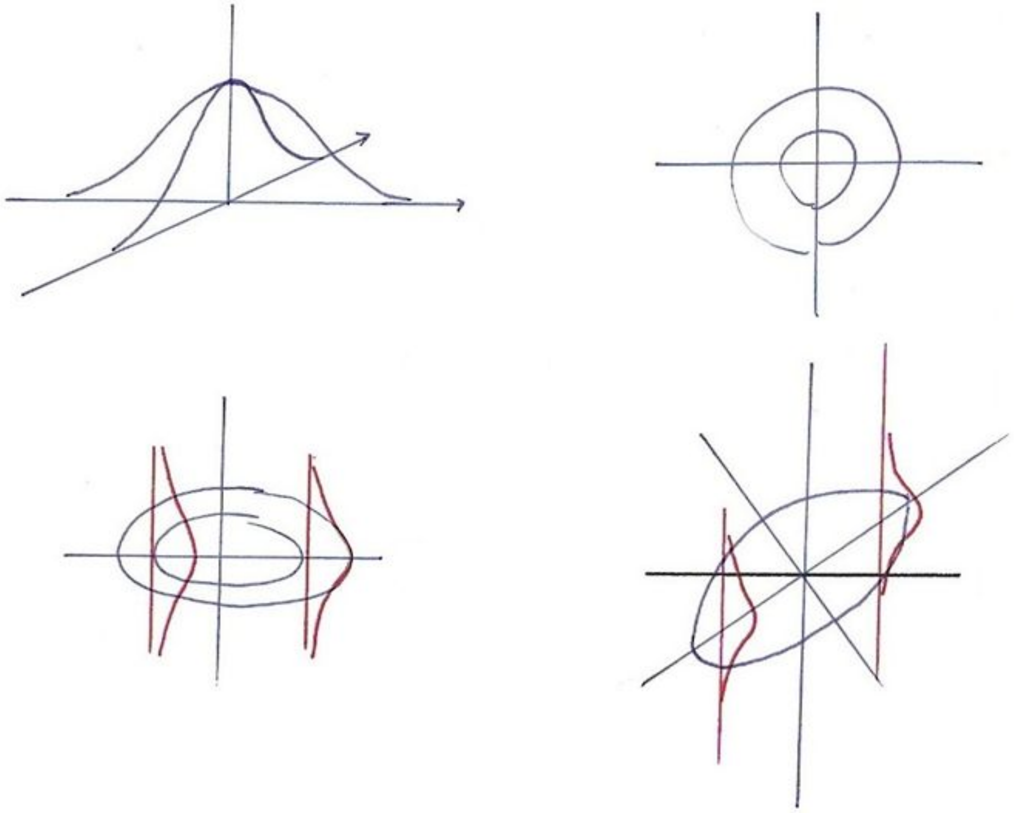
1-dim Gaussian Mixtures



http://speech.ee.ntu.edu.tw/DSP2019Autumn/Slides/2.0.pptx

GMM (Gaussian Mixture Model)

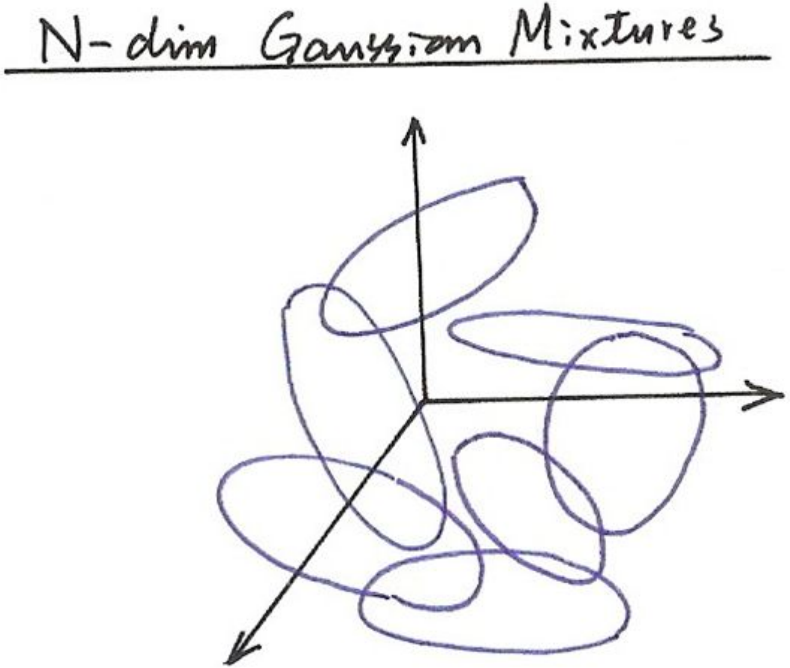
<u>2-dim Gaussian</u>



http://speech.ee.ntu.edu.tw/DSP2019Autumn/Slides/2.0.pptx

GMM (Gaussian Mixture Model)

N-dim Gaussian Mixtures



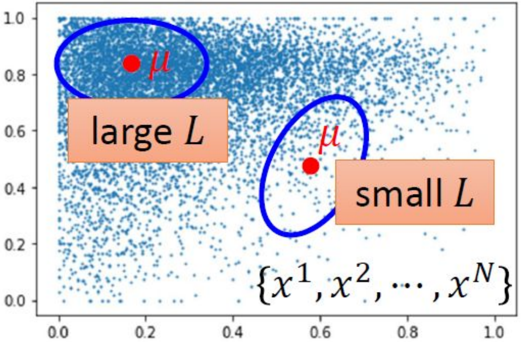
http://speech.ee.ntu.edu.tw/DSP2019Autumn/Slides/2.0.pptx

GMM (Gaussian Mixture Model)

$$f_{\mu,\Sigma}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$

Input: vector x, output: probability of sampling x

θ which determines the shape of the function are **mean** $\boldsymbol{\mu}$ and **covariance matrix** $\boldsymbol{\Sigma}$



$$L(\theta) = f_\theta(x^1) f_\theta(x^2) \cdots f_\theta(x^N)$$

$$L(\mu,\Sigma) = f_{\mu,\Sigma}(x^1) f_{\mu,\Sigma}(x^2) \cdots f_{\mu,\Sigma}(x^N)$$

$$\theta^* = arg \max_\theta L(\theta)$$

$$\mu^*, \Sigma^* = arg \max_{\mu,\Sigma} L(\mu,\Sigma)$$

$$\mu^* = \frac{1}{N}\sum_{n=1}^{N} x^n = \begin{bmatrix} 0.29 \\ 0.73 \end{bmatrix}$$

$$\Sigma^* = \frac{1}{N}\sum_{n=1}^{N} (x-\mu^*)(x-\mu^*)^T = \begin{bmatrix} 0.04 & 0 \\ 0 & 0.03 \end{bmatrix}$$

GMM (Gaussian Mixture Model)

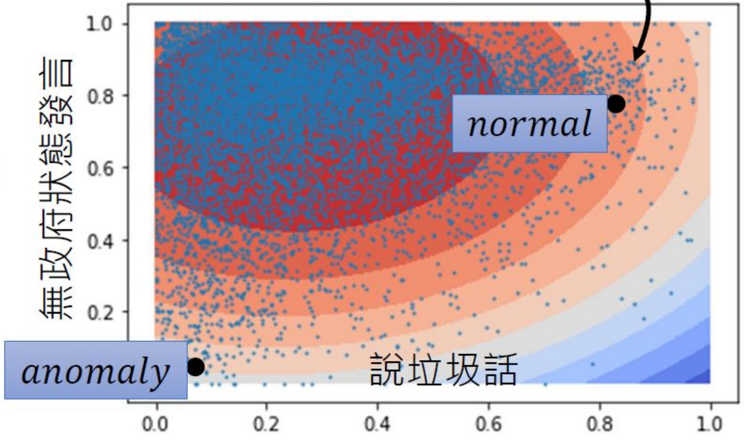
$$f_{\mu^*,\Sigma^*}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^*|^{1/2}} exp\left\{ -\frac{1}{2}(x-\mu^*)^T {\Sigma^*}^{-1}(x-\mu^*) \right\}$$

$$\mu^* = \begin{bmatrix} 0.29 \\ 0.73 \end{bmatrix} \quad \Sigma^* = \begin{bmatrix} 0.04 & 0 \\ 0 & 0.03 \end{bmatrix}$$

$$\boxed{f(x) = \begin{cases} normal, & f_{\mu^*,\Sigma^*}(x) > \lambda \\ anomaly, & f_{\mu^*,\Sigma^*}(x) \leq \lambda \end{cases}}$$

λ is a contour line

The colors represents the value of $f_{\mu^*,\Sigma^*}(x)$



GMM (Gaussian Mixture Model)

```
from keras.datasets import mnist
import cv2
import numpy as np

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_ok = x_train[y_train == 1] # 6742 筆
x_test = x_test[(y_test == 7) | (y_test == 1)] # 1135 筆 "1", 1028 筆 "7"
y_test = y_test[(y_test == 7) | (y_test == 1)]

def reshape_x(x):
    new_x = np.empty((len(x), 56, 56))
    for i, e in enumerate(x):
        new_x[i] = cv2.resize(e, (56, 56))

    new_x = np.expand_dims(new_x, axis=-1)
    new_x = np.repeat(new_x, 3, axis=-1)
    return new_x

x_ok = reshape_x(x_ok)
x_test = reshape_x(x_test)
```

https://towardsdatascience.com/a-simple-way-to-detect-anomaly-3d5a48c0dae0

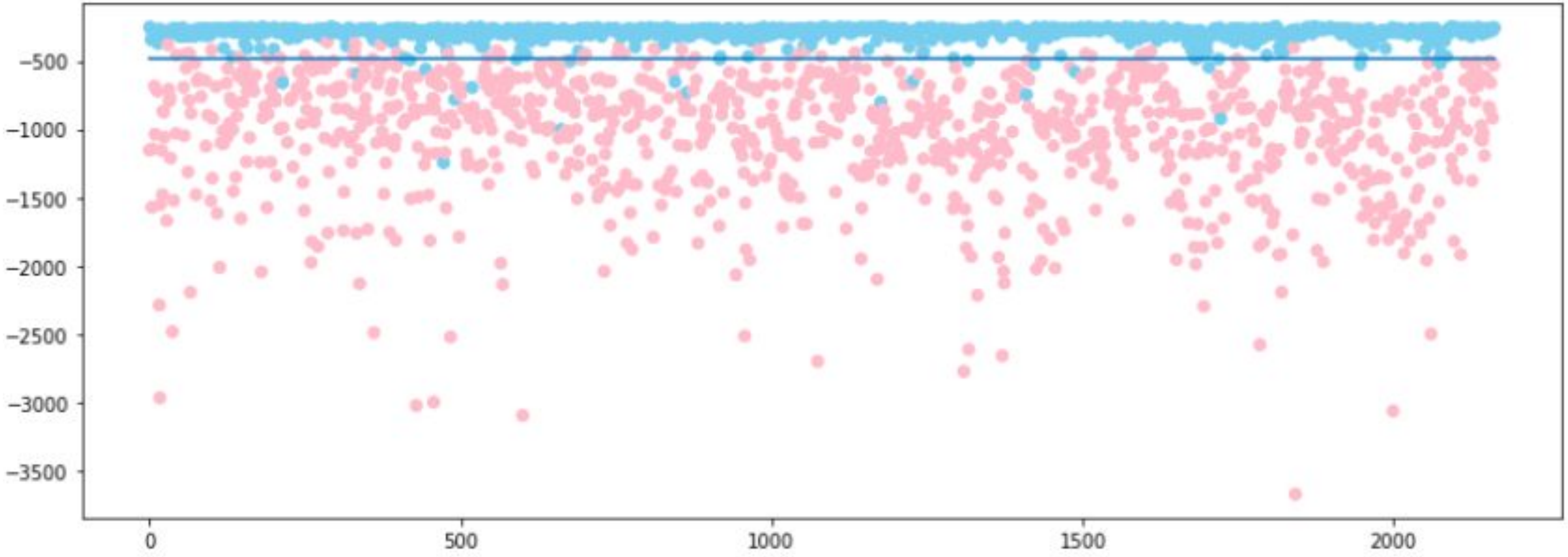
GMM (Gaussian Mixture Model)

```
features = model.predict(x_ok)
gmm.fit(features)
OKscore = gmm.score_samples(features)
thred = OKscore.mean() - 3 * OKscore.std()

test_features = model.predict(x_test)
score = gmm.score_samples(test_features)

print('normal accuracy: %.2f' % (len(score[(y_test == 1) & (score > thred)]) / 1135))
print('abnormal accuracy: %.2f' % (len(score[(y_test == 7) & (score < thred)]) / 1028))
```

```
normal accuracy: 0.98
abnormal accuracy: 0.96
```

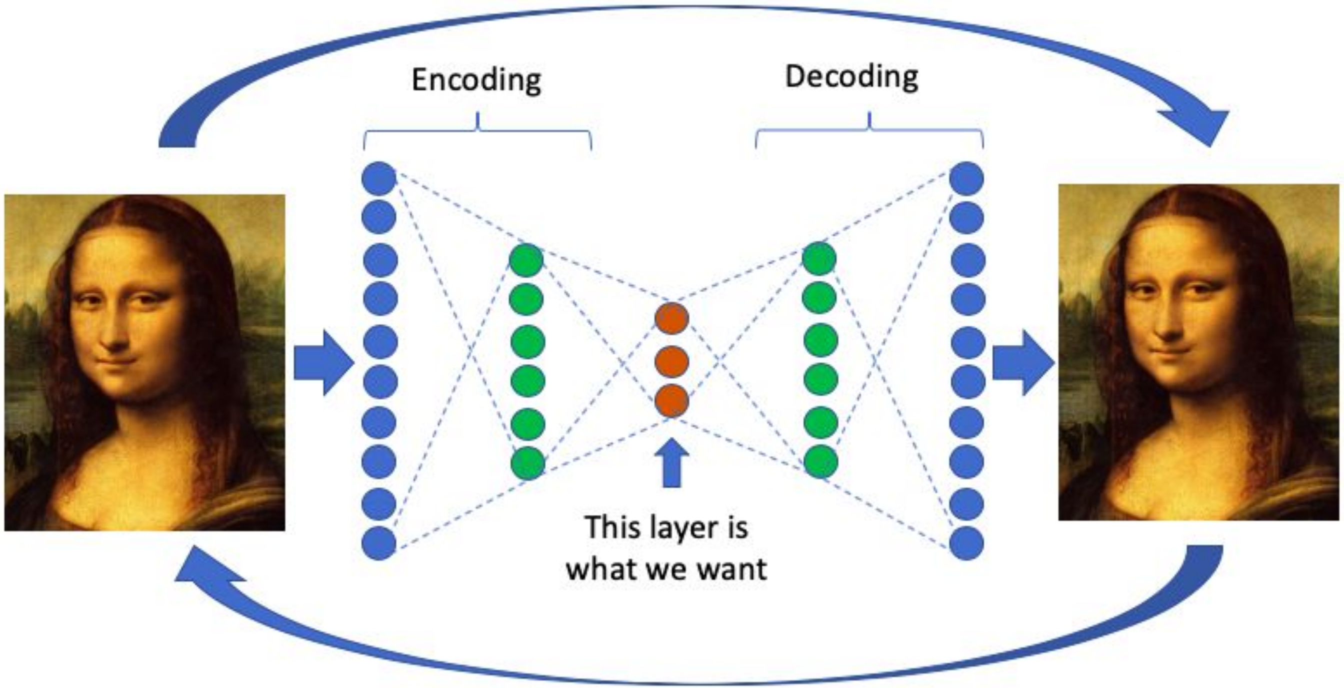
https://towardsdatascience.com/a-simple-way-to-detect-anomaly-3d5a48c0dae0

GMM (Gaussian Mixture Model)



https://towardsdatascience.com/a-simple-way-to-detect-anomaly-3d5a48c0dae0

Auto-Encoder



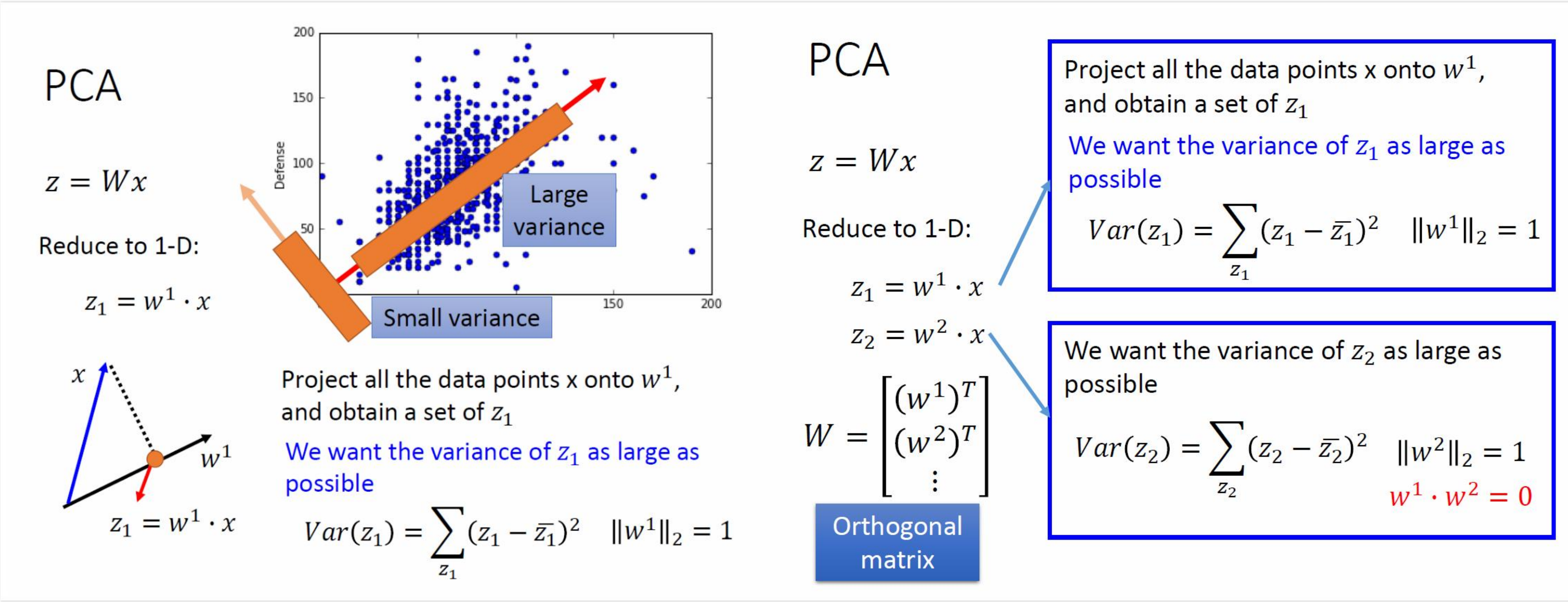
Auto-Encoder

Examples:

https://www.kaggle.com/tikedameu/anomaly-detection-with-autoencoder-pytorch

https://towardsdatascience.com/anomaly-detection-with-autoencoder-b4cdce4866a6

PCA



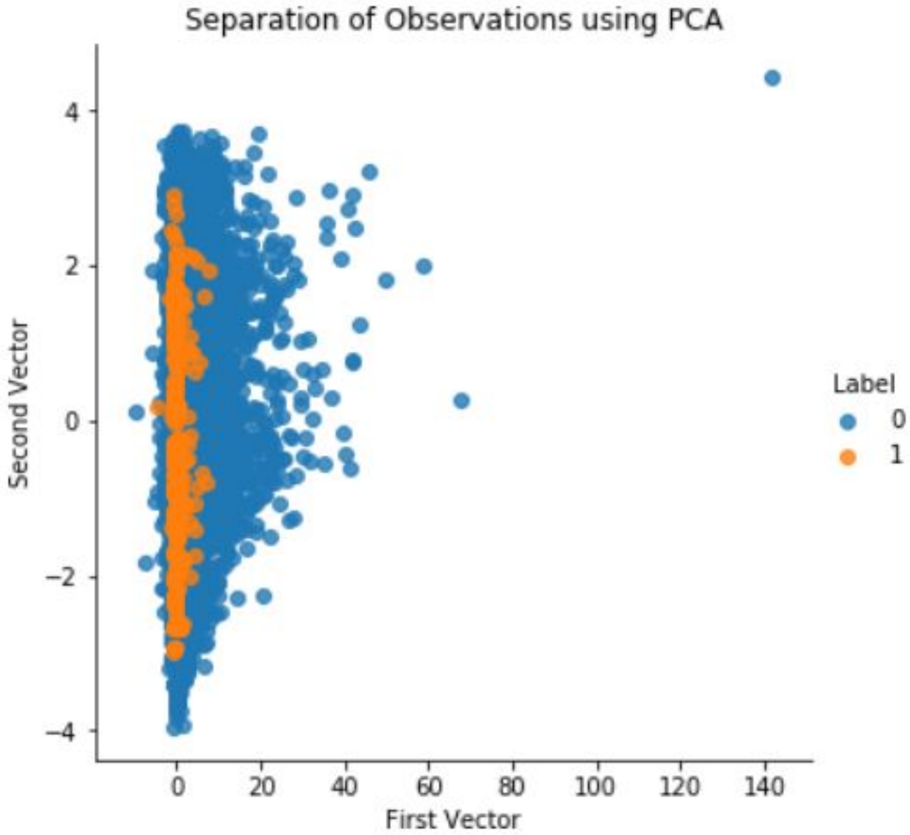
http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2017/Lecture/PCA%20(v3).pdf

PCA

```python
# 30 principal components
from sklearn.decomposition import PCA

n_components = 30
whiten = False
random_state = 2018

pca = PCA(n_components=n_components, whiten=whiten, \
          random_state=random_state)

X_train_PCA = pca.fit_transform(X_train)
X_train_PCA = pd.DataFrame(data=X_train_PCA, index=X_train.index)

X_train_PCA_inverse = pca.inverse_transform(X_train_PCA)
X_train_PCA_inverse = pd.DataFrame(data=X_train_PCA_inverse, \
                                   index=X_train.index)

scatterPlot(X_train_PCA, y_train, "PCA")
```

https://www.oreilly.com/library/view/hands-on-unsupervised-learning/9781492035633/ch04.html

PCA



https://www.oreilly.com/library/view/hands-on-unsupervised-learning/9781492035633/ch04.html

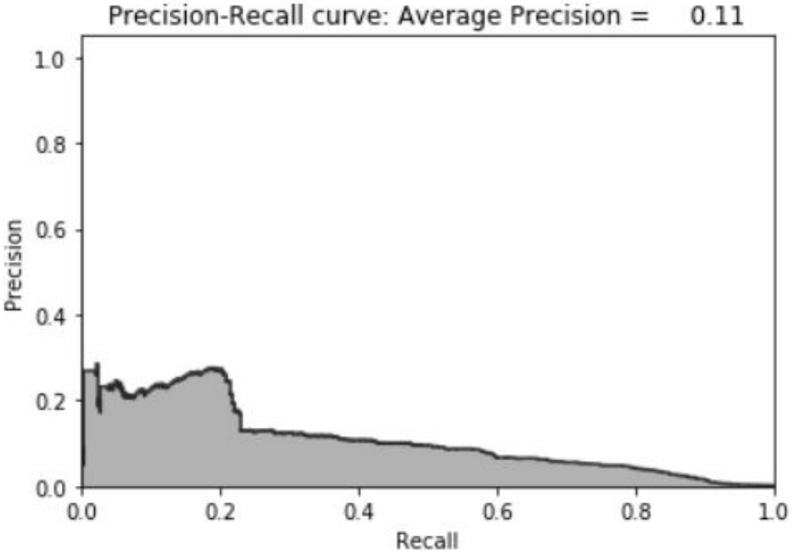
PCA

```python
def anomalyScores(originalDF, reducedDF):
    loss = np.sum((np.array(originalDF)-np.array(reducedDF))**2, axis=1)
    loss = pd.Series(data=loss,index=originalDF.index)
    loss = (loss-np.min(loss))/(np.max(loss)-np.min(loss))
    return loss
```

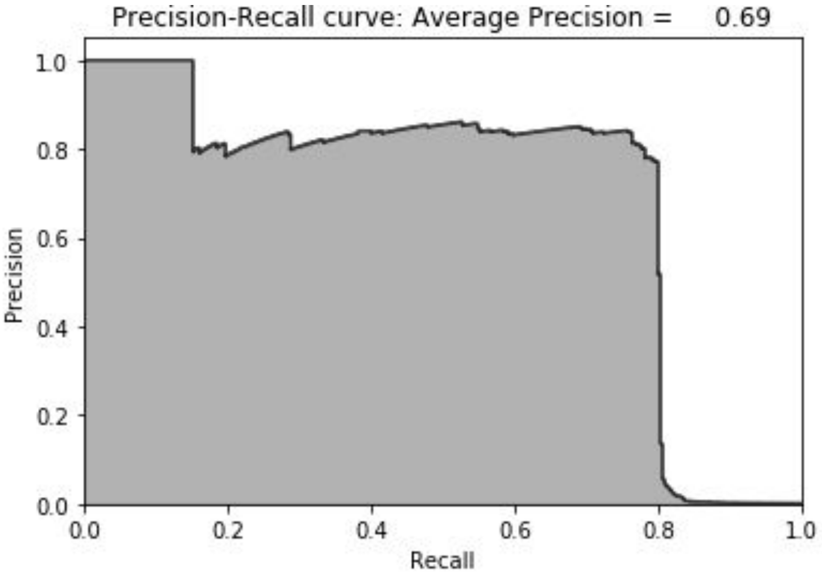
```python
anomalyScoresPCA = anomalyScores(X_train, X_train_PCA_inverse)
preds = plotResults(y_train, anomalyScoresPCA, True)
```

https://www.oreilly.com/library/view/hands-on-unsupervised-learning/9781492035633/ch04.html

PCA



n_components = 30

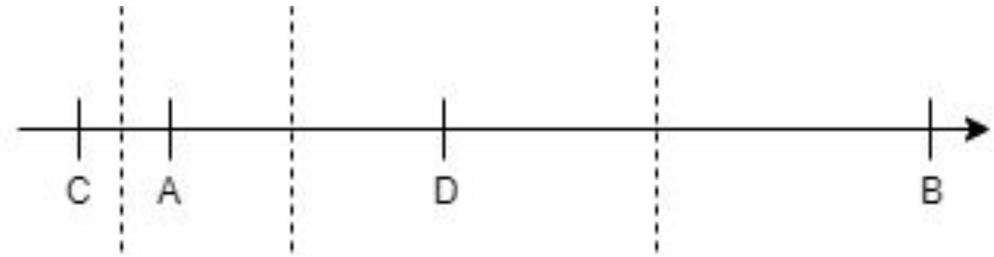


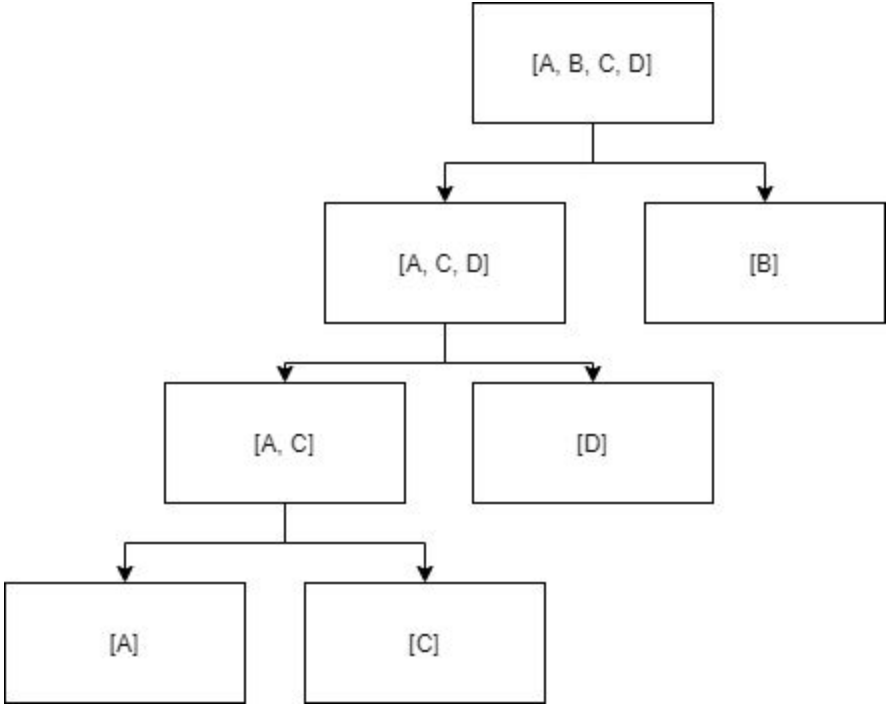
n_components = 27

https://www.oreilly.com/library/view/hands-on-unsupervised-learning/9781492035633/ch04.html

Isolation Forest

- Use tree like structure to split data



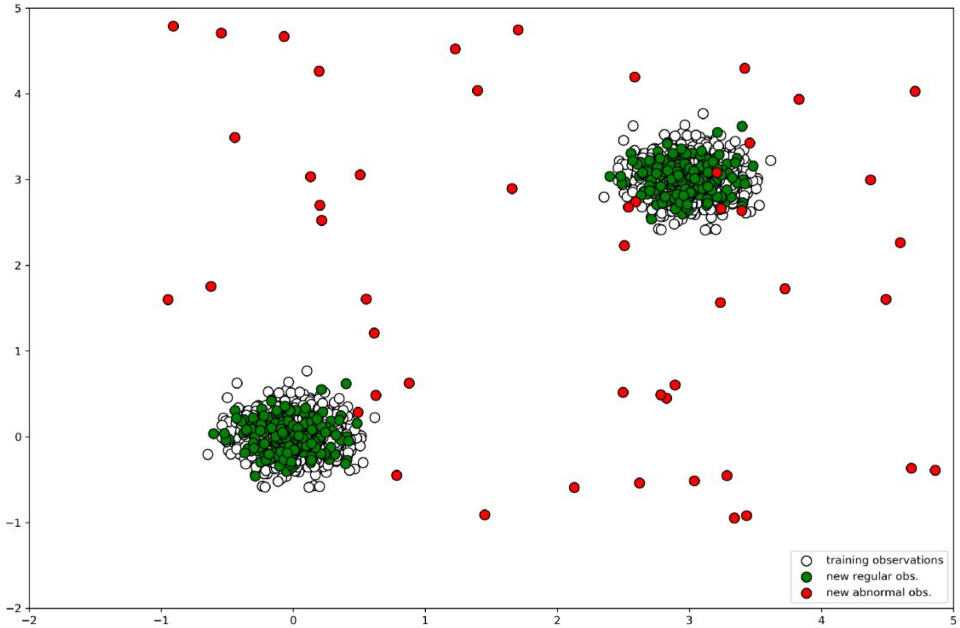


Isolation Forest

```
# importing libaries ----
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pylab import savefig
from sklearn.ensemble import IsolationForest


# Generating data ----

rng = np.random.RandomState(42)

# Generating training data
X_train = 0.2 * rng.randn(1000, 2)
X_train = np.r_[X_train + 3, X_train]
X_train = pd.DataFrame(X_train, columns = ['x1', 'x2'])

# Generating new, 'normal' observation
X_test = 0.2 * rng.randn(200, 2)
X_test = np.r_[X_test + 3, X_test]
X_test = pd.DataFrame(X_test, columns = ['x1', 'x2'])

# Generating outliers
X_outliers = rng.uniform(low=-1, high=5, size=(50, 2))
X_outliers = pd.DataFrame(X_outliers, columns = ['x1', 'x2'])
```

https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e

Isolation Forest



https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e

Isolation Forest

```
# Isolation Forest ----

# training the model
clf = IsolationForest(max_samples=100, random_state=rng)
clf.fit(X_train)

# predictions
y_pred_train = clf.predict(X_train)
y_pred_test = clf.predict(X_test)
y_pred_outliers = clf.predict(X_outliers)
```

https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e

Isolation Forest

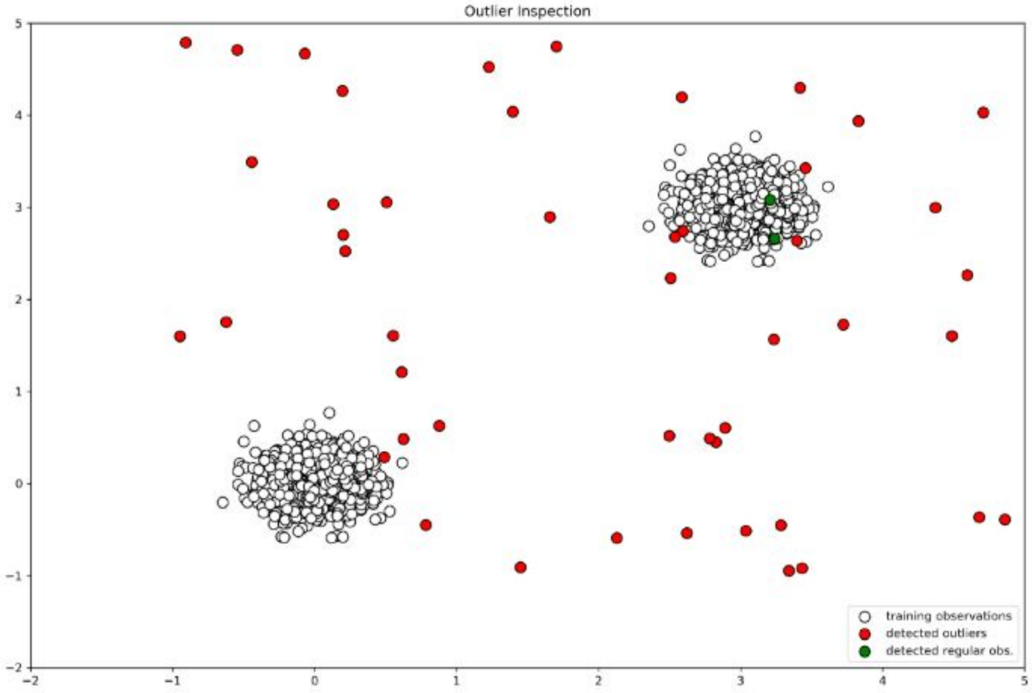


Figure 3 Inspecting outlier classification

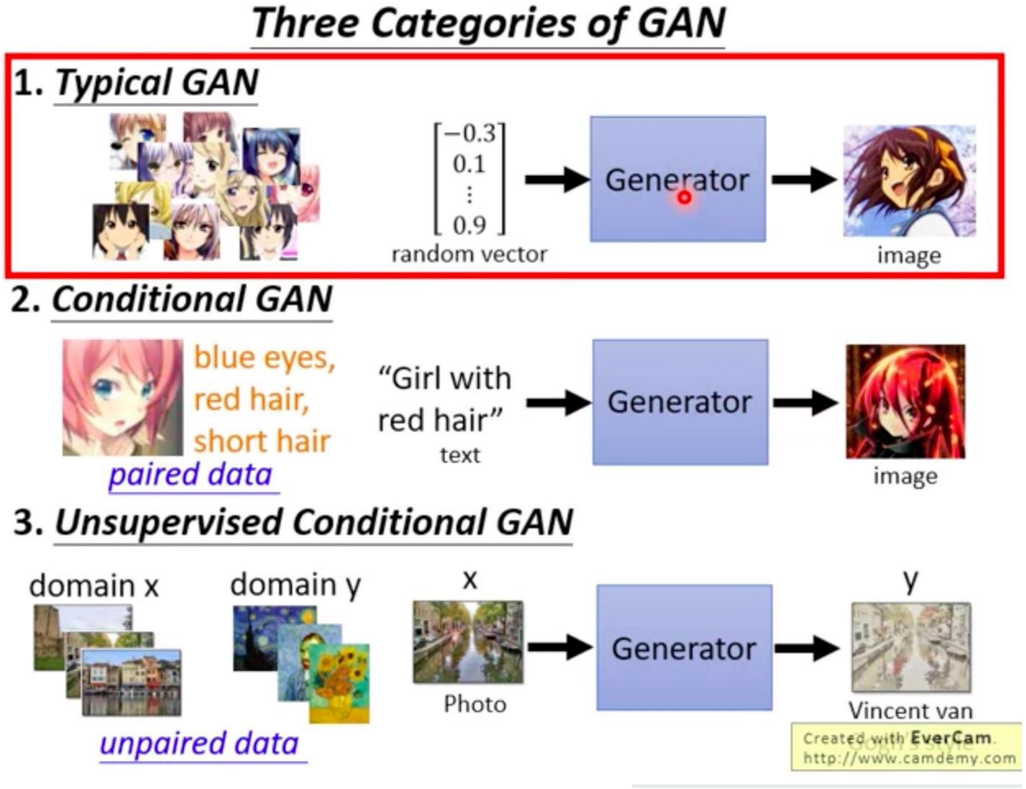
https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e

Summary of classic method

- With Classifier
- GMM (Gaussian Mixture Model)
- Auto-Encoder
- PCA
- Isolation Forest

Anomaly detection on image

Typical GANs



http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2018/Lecture/GAN%20(v2).pdf

Typical GANs

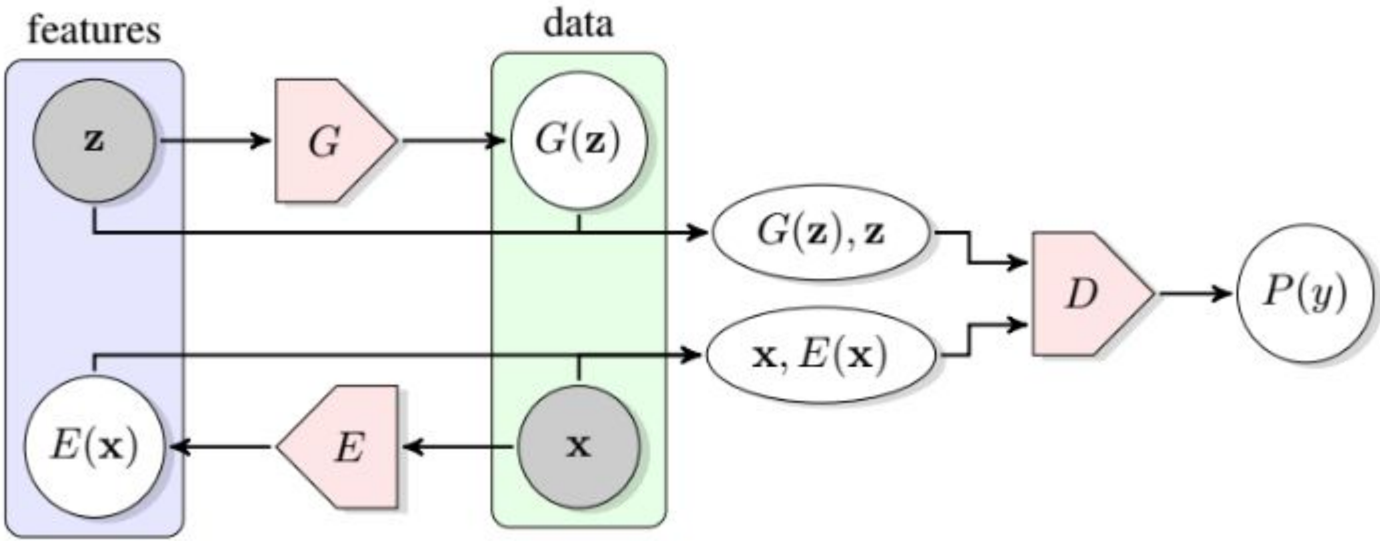


Figure 1. The structure of BiGAN proposed in (Donahue et al., 2016).

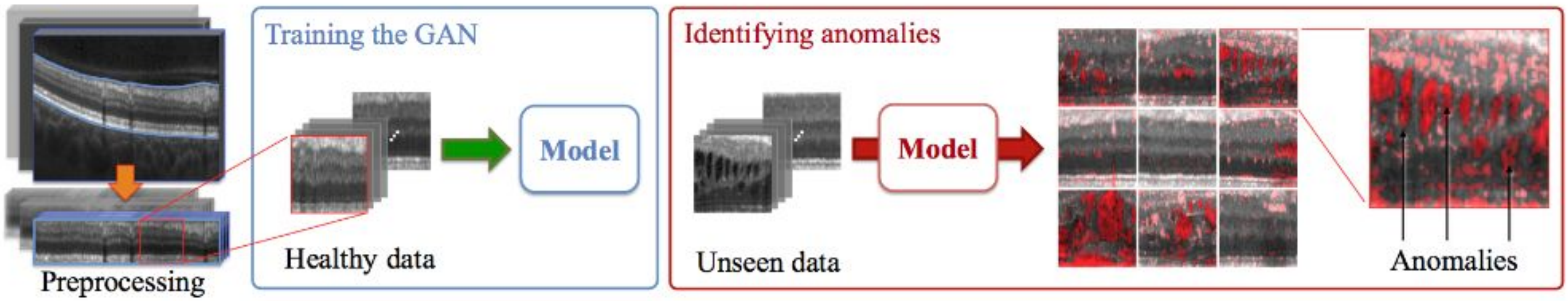
https://arxiv.org/pdf/1906.11632.pdf

Example

(a) Normal Data (X-ray Scans) (b) Normal + Abnormal Data (X-ray Scans)

AnoGAN

- Train a standard GAN only on positive samples



https://arxiv.org/pdf/1906.11632.pdf

AnoGAN

- Anomaly Score

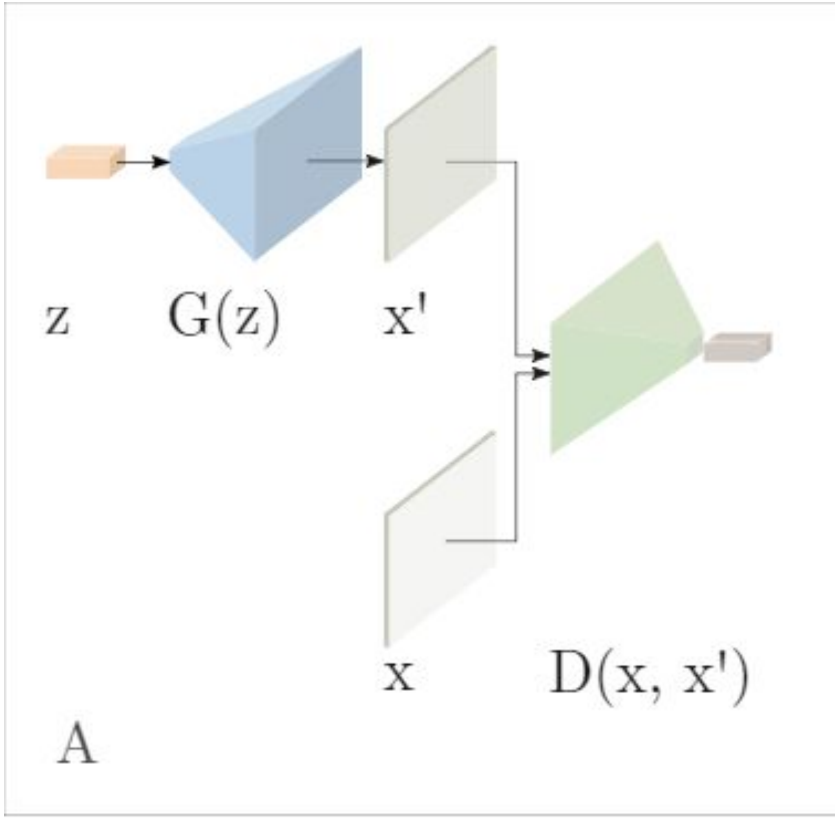
$$A(\mathbf{x}) = \mathcal{L}(\mathbf{z}_\Gamma)$$

- for γ = 1, 2, … Γ find proper z

$$\mathcal{L}_R(\mathbf{z}_\gamma) = ||\mathbf{x} - G(\mathbf{z}_\gamma)||_1$$

$$\mathcal{L}_D(\mathbf{z}_\gamma) = ||\mathbf{f}(\mathbf{x}) - \mathbf{f}(G(\mathbf{z}_\gamma))||_1$$

$$\mathcal{L}(\mathbf{z}_\gamma) = (1-\lambda) \cdot \mathcal{L}_R(\mathbf{z}_\gamma) + \gamma \cdot \mathcal{L}_D(\mathbf{z}_\gamma)$$



AnoGAN

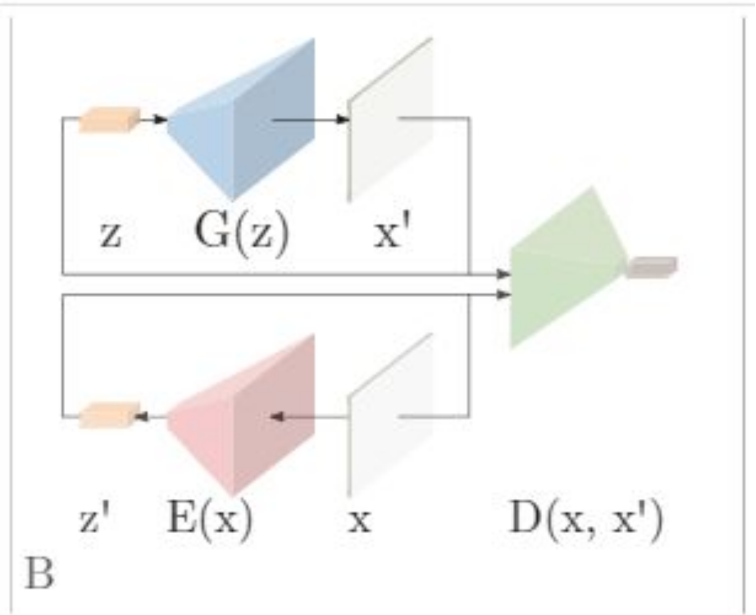
- Pros
  - Showed that GANs can be used for anomaly detection
  - Introduced a new mapping scheme from latent space to input data space.
  - Used the same mapping scheme to define an anomaly score.
- Cons
  - Requires Γ optimization steps for every new input: bad test-time performance
  - The GAN objective has not been modified to take into account the need for the inverse mapping learning.
  - The anomaly score is difficult to interpret, not being in the probability range.

https://arxiv.org/pdf/1906.11632.pdf

EGBAD (Efficient GAN-Based Anomaly Detection)

- Train a Bi-GAN only on positive samples

$$\min_{G,E} \max_{D} V(D, G, E) =$$
$$\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\mathbb{E}_{\mathbf{z}\sim p_{E(\mathbf{z}|\mathbf{x})}}[\log D(\mathbf{x},\mathbf{z})]]+$$
$$\mathbb{E}_{\mathbf{z}\sim p_{z}(\mathbf{z})}[\mathbb{E}_{\mathbf{x}\sim p_{G}(\mathbf{x}|\mathbf{z})}[\log(1 - D(\mathbf{x},\mathbf{z}))]].$$



https://arxiv.org/pdf/1906.11632.pdf

EGBAD (Efficient GAN-Based Anomaly Detection)

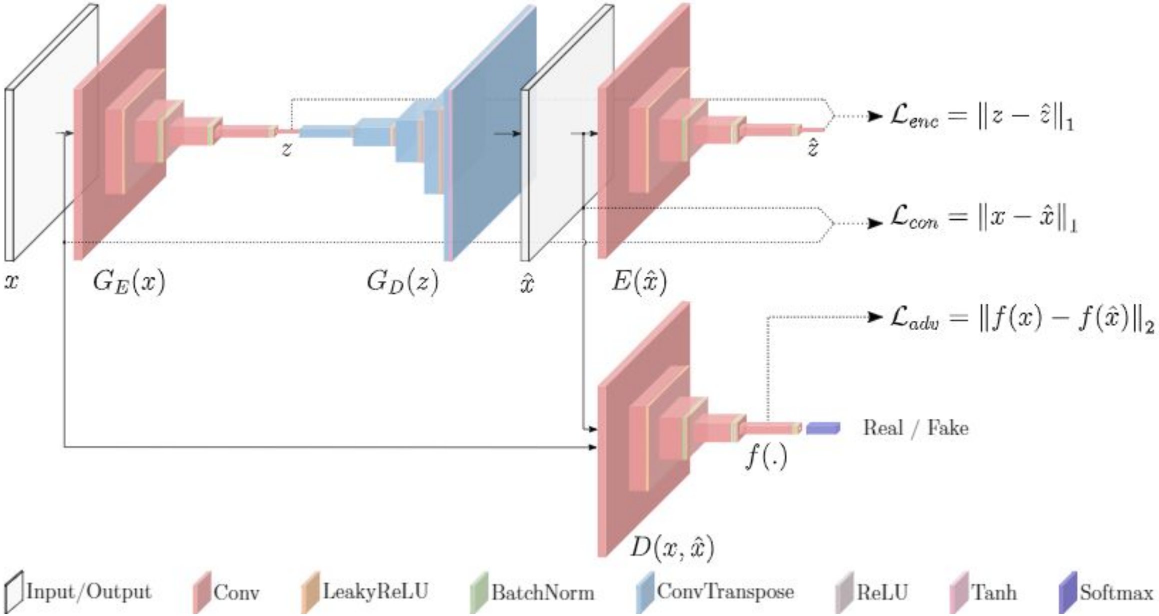
- Pros
  - The Encoder E can learns how to encode image while adversarial training.
  - Therefore, it can bypass Γ optimization steps of AnoGAN while calculating anomaly score.

GANomaly

- Generator is composed of encoder G_E, decoder G_D, and encoder E
- Trained on only normal data.

- Anomaly score

$$\mathcal{A}(\mathbf{x}) = ||G_E(\mathbf{x}) - E(G(\mathbf{x}))||_2$$



https://arxiv.org/pdf/1906.11632.pdf

GANomaly

- Pros
  - An encoder is learned during the training process, so it can bypass the Γ optimization.
  - Using an autoencoder like architecture (no use of noise prior) makes the entire learning process faster.
  - The contextual loss can be used to localize the anomaly.
- Cons
  - Defines a new anomaly score.
  - It allows to detect anomalies both in the image space and in the latent space, but the results couldn't match:
    - a higher anomaly score, that's computed only in the latent space, can be associated with a generated sample with a low contextual loss value and thus very similar to the input - and vice versa.

https://arxiv.org/pdf/1906.11632.pdf

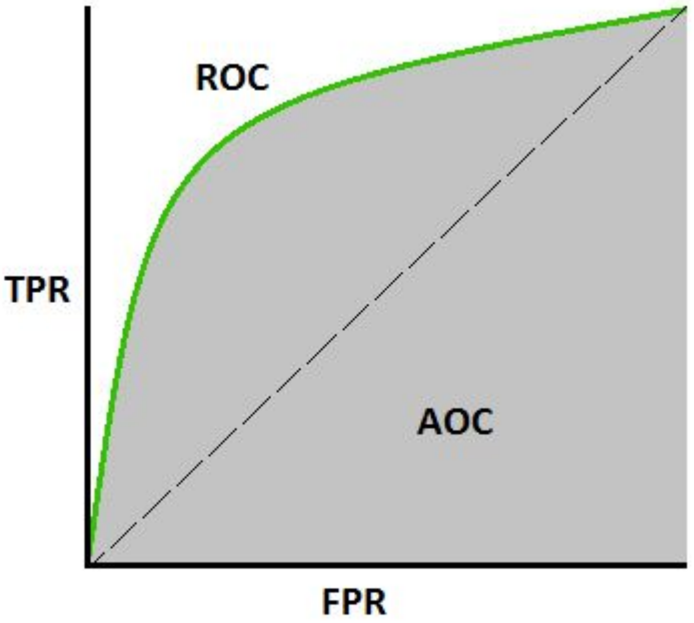
Evaluation Metric

- TPR (True Positive Rate)

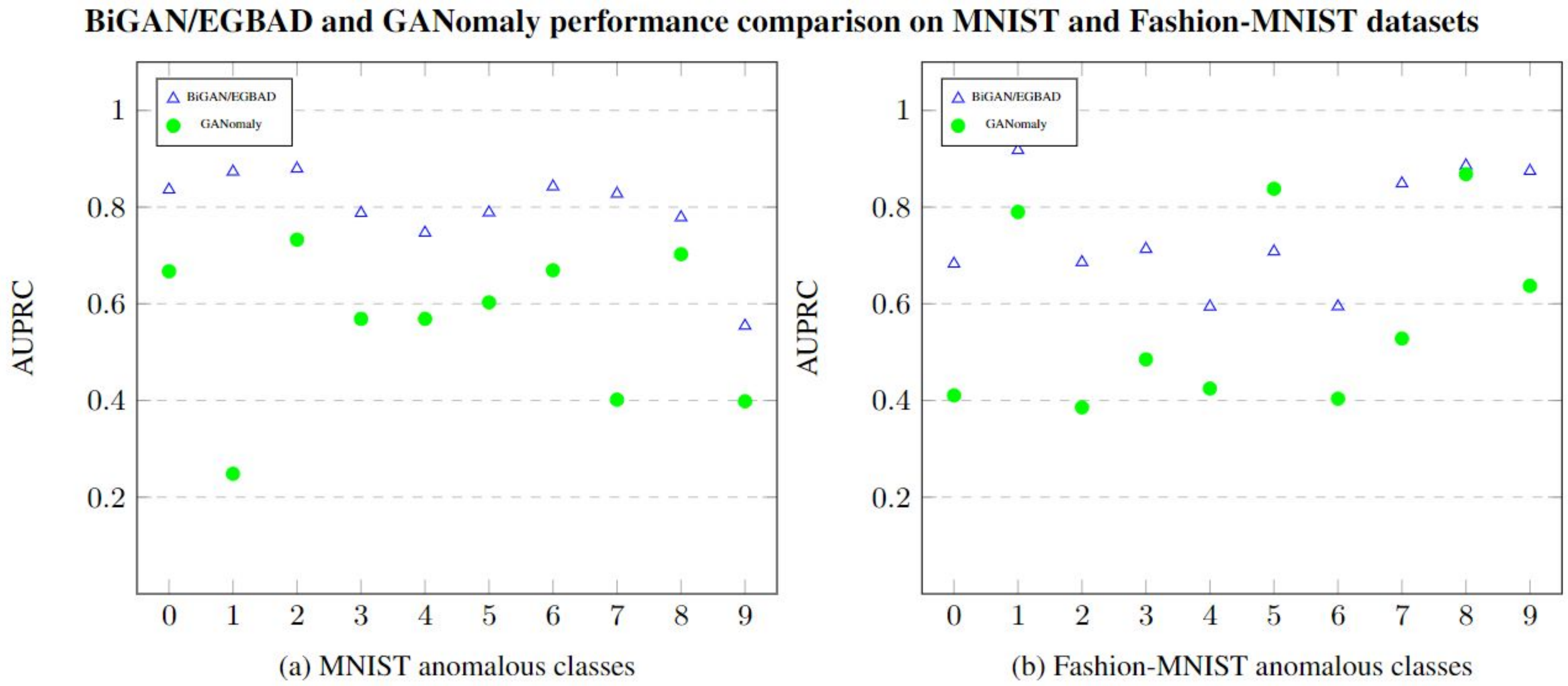
$$\frac{TP}{TP + FN}$$

- FPR (False Positive Rate)

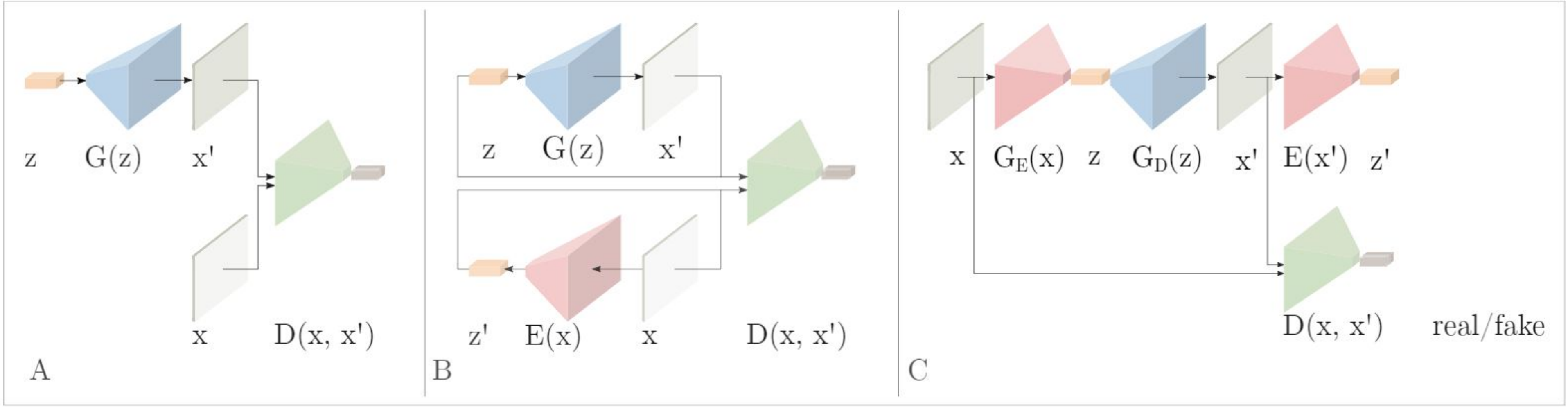
$$\frac{FP}{TN + FP}$$



Comparison


BiGAN/EGBAD and GANomaly performance comparison on MNIST and Fashion-MNIST datasets
BiGAN/EGBAD
GANomaly
AUPRC
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
(a) MNIST anomalous classes
BiGAN/EGBAD
GANomaly
AUPRC
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
(b) Fashion-MNIST anomalous classes

Summary of GANs



AnoGAN EGBAD GANomaly

Reference

https://arxiv.org/pdf/1711.09325.pdf

https://arxiv.org/pdf/1809.10816.pdf

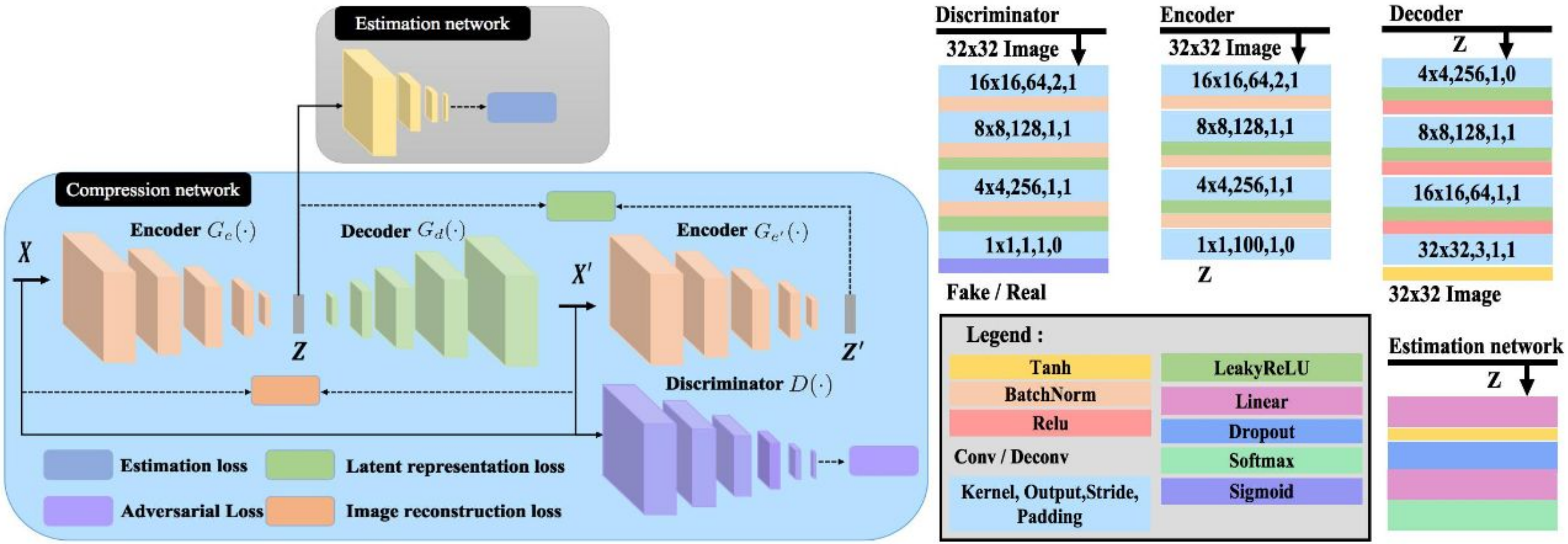
https://arxiv.org/pdf/1812.02288.pdf

https://arxiv.org/pdf/1901.08954.pdf

https://arxiv.org/pdf/1905.13147.pdf

Anomaly detection on Audio

GMGAN (Gaussian Mixture GAN)



https://arxiv.org/pdf/2002.01107.pdf

GMGAN (Gaussian Mixture GAN)

- Adversarial loss

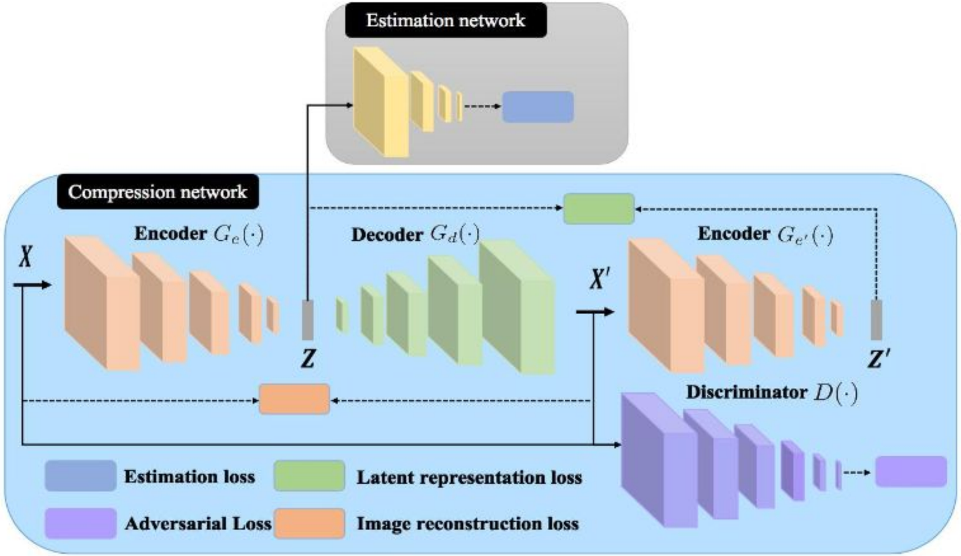
$$\mathcal{L}_{adv} = \min_{G} \max_{D} (E_{\boldsymbol{x} \sim p_{\mathbf{x}}}[\log(D(\mathbf{x}))] + E_{\boldsymbol{x} \sim p_{\mathbf{x}}}[\log(1 - D(G(\mathbf{x})))]).$$

- Image reconstruction loss

$$\mathcal{L}_{irec} = \mathbb{E}_{x \sim p_{\mathbf{x}}} \|x - G(x)\|_1$$

- Latent representation loss

$$\mathcal{L}_{zrec} = \mathbb{E}_{x \sim p_{\mathbf{X}}} \|G_e(x) - G_{e'}(x')\|_2$$



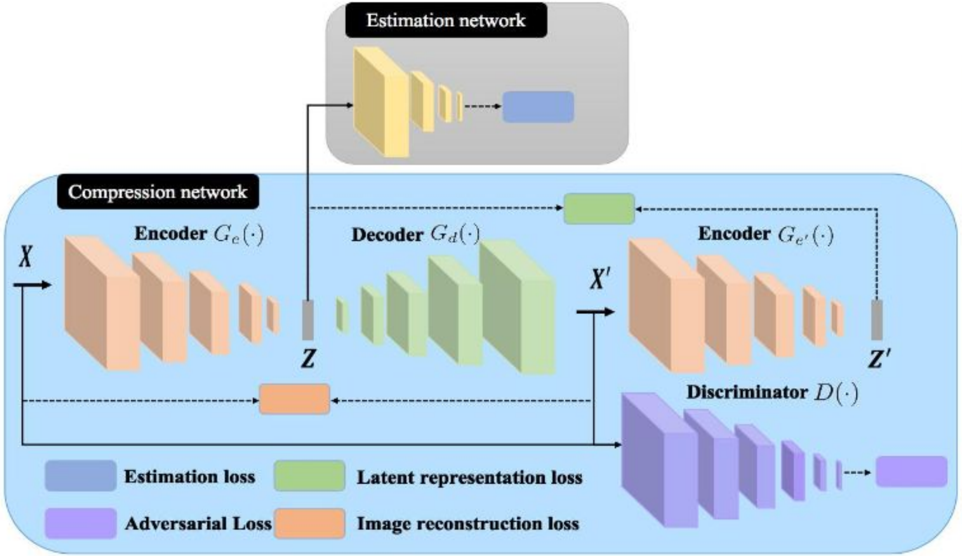
https://arxiv.org/pdf/2002.01107.pdf

GMGAN (Gaussian Mixture GAN)

- Estimation loss
- $\hat{\gamma} = \mathrm{softmax}(MLN\,(\mathbf{z};\theta_m))$
- $\hat{\gamma}$ is a K-dimensional vector and.
  $\hat{\gamma}_k$ denotes the input belongs to
  the k^th distribution.
- Calculate the component of k^th mixture.

$$\hat{\alpha}_k = \frac{1}{n}\sum_{i=1}^{n}\hat{\gamma}_{ik};\ \hat{u}_k = \frac{\sum_{i=1}^{n}\hat{\gamma}_{ik}z_i}{\sum_{i=1}^{n}\hat{\gamma}_{ik}},$$

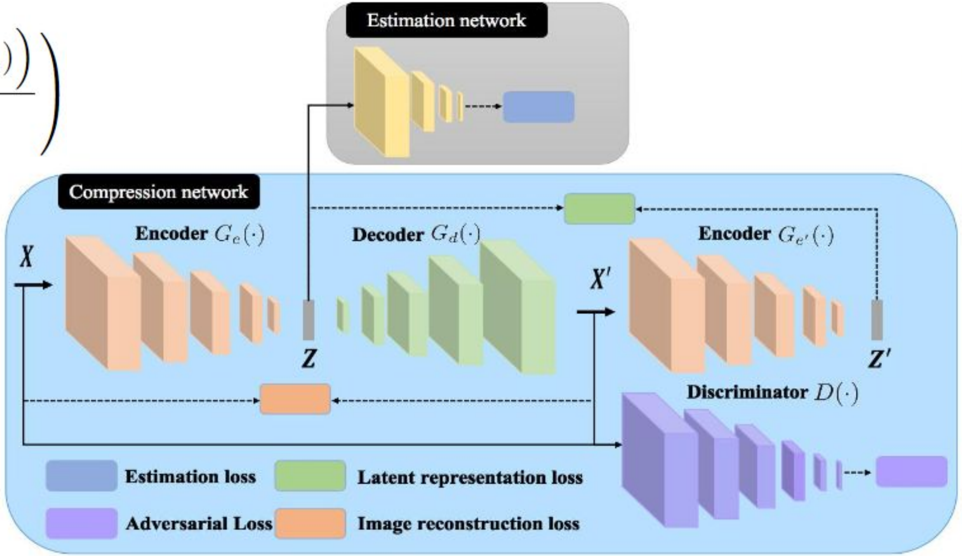
$$\hat{\Sigma}_k = \frac{\sum_{i=1}^{n}\hat{\gamma}_{ik}\left(z_i-\hat{u}_k\right)\left(z_i-\hat{u}_k\right)^T}{\sum_{i=1}^{n}\hat{\gamma}_{ik}},$$



GMGAN (Gaussian Mixture GAN)

- Estimation loss
- $$E(\mathbf{z}) = -\log\left(\sum_{k=1}^{K}\hat{\alpha}_k \frac{\exp\left(-\frac{1}{2}\left(\mathbf{z}-\hat{\mu}_k\right)^T\hat{\boldsymbol{\Sigma}}_k^{-1}\left(\mathbf{z}-\hat{\mu}_k\right)\right)}{\sqrt{|2\pi\Sigma_k|}}\right)$$

$$\mathcal{L}_{es} = \lambda_1\sum_{i=1}^{N}E\left(z_i\right) + \lambda_2\sum_{k=1}^{K}\sum_{j=1}^{d}\frac{1}{\hat{\Sigma}_{jj}^{k}},$$



GMGAN (Gaussian Mixture GAN)

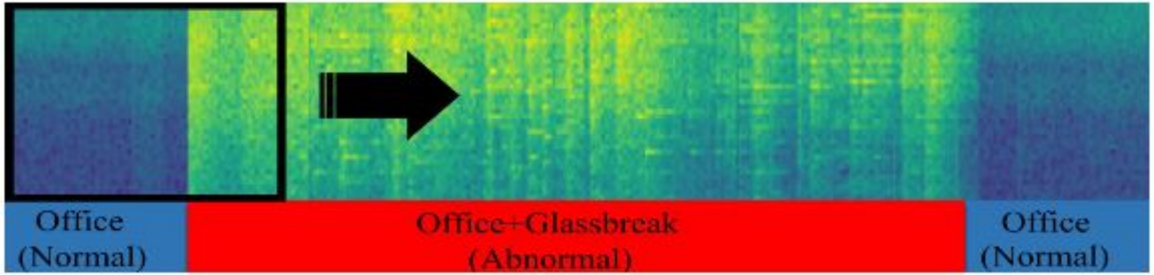


Fig. 2. Spectrogram of one second sequence containing office background audio and glass break audio.

https://arxiv.org/pdf/2002.01107.pdf

GMGAN (Gaussian Mixture GAN)

Scene	CAE [9]	WaveNet [19]	Proposed method
beach	0.69	0.72	**0.80**
bus	0.79	0.83	**0.89**
cafe/restaurant	0.69	**0.76**	**0.76**
car	0.79	0.82	**0.93**
city center	0.75	0.82	**0.83**
forest path	0.65	0.72	**0.77**
grocery store	0.71	0.77	**0.83**
home	**0.69**	**0.69**	**0.69**
library	0.59	0.67	**0.85**
metro station	0.74	0.79	**0.81**
office	0.78	0.78	**0.80**
park	0.70	0.80	**0.89**
residential area	0.73	**0.78**	**0.78**
train	0.82	0.84	**0.92**
tram	0.80	0.87	**0.94**

Outline

- What is Anomaly Detection
- Classic Method
  - With Classifier
  - GMM (Gaussian Mixture Model)
  - Auto-Encoder
  - PCA
  - Isolation Forest
  - Summary
- Anomaly Detection with GAN
  - AnoGAN
  - EGBAD
  - GANomaly
  - Summary
- Anomaly Detection on Audio
  - GMGAN

More Reference

Anomaly detection on self-driving

http://taoxie.cs.illinois.edu/publications/tii17-safedrive.pdf

https://arxiv.org/pdf/2004.09496.pdf

https://arxiv.org/pdf/2004.12581.pdf