Robust Deep Learning Methods for Anomaly Detection

Prof Sanjay Chawla

Raghav Chalapathy

Nguyen Lu Dang Khoa

QCRI
معهد قطر لبحوث الحوسبة
Qatar Computing Research Institute
جامعة حمد بن خليفة
HAMAD BIN KHALIFA UNIVERSITY

CSIRO DATA 61

July 21, 2020

Overview

Auto-encoders

- Auto-encoder
- Variational Auto-Encoders (VAE)
- Adversarial Auto-Encoders (AAE)
- Wasserstein Auto-Encoders (WAE)

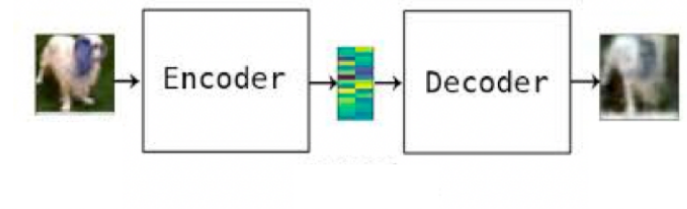
Extending Auto-encoders

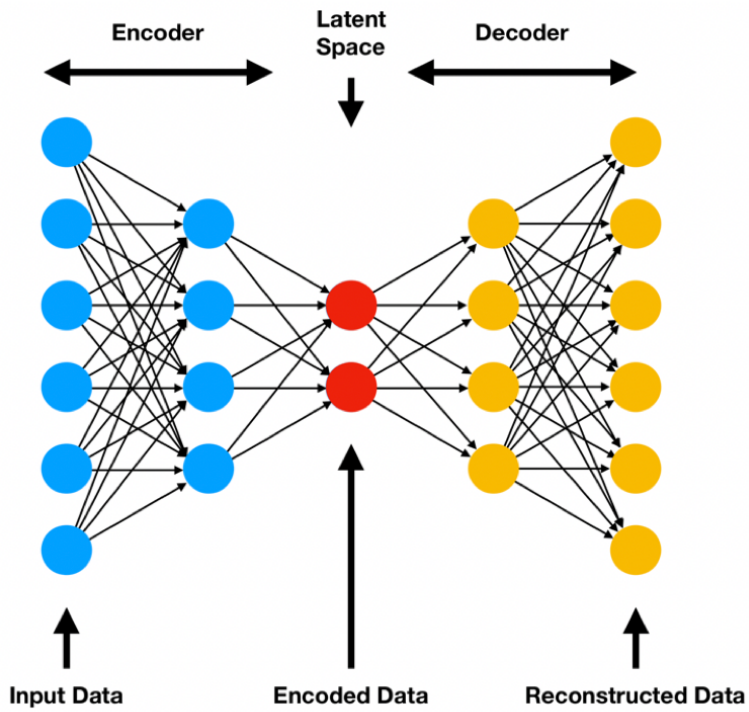
- One-Class Deep SVDD (Deep SVDD)
- One-Class Neural Networks (OCNN)

Modified Autoencoder Training and Scoring

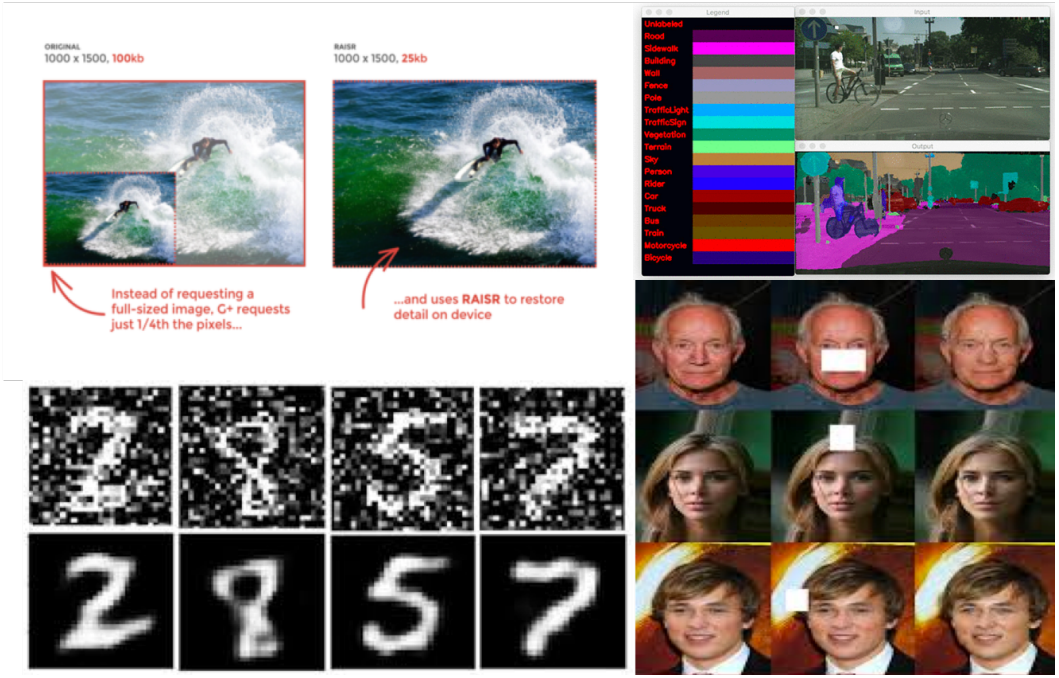
Basic Auto-encoder

- Data specific learned nonlinear compression.
- Compute the reconstruction loss with respect to input.





Application : Auto-encoder



- Helps save bandwidth : Data Transmission.
- Image Segmentation : Autonomous Driving.
- Image Denoising : Remove Noise
- Neural Inpainting : Fill in missing image portion, remove watermarks

Auto-encoders for anomaly detection.

- Auto encoder with single **hidden layer**

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X} - \mathbf{f}(\mathbf{XU})\mathbf{V}\|_{\mathbf{F}}^{2} \tag{1}$$

- $\hat{\mathbf{X}} = f(\mathbf{XU})\mathbf{V}$ is **reconstruction error measure.**

Auto-encoders for anomaly detection.

- Auto encoder with single **hidden layer**

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X} - \mathbf{f}(\mathbf{XU})\mathbf{V}\|_{\mathbf{F}}^{\mathbf{2}} \tag{1}$$

- $\hat{\mathbf{X}} = f(\mathbf{XU})\mathbf{V}$ is **reconstruction error measure.**
  - **U**: Weights (input $\rightarrow$ hidden),

Auto-encoders for anomaly detection.

- Auto encoder with single **hidden layer**

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X} - \mathbf{f}(\mathbf{XU})\mathbf{V}\|_{\mathbf{F}}^{\mathbf{2}} \tag{1}$$

- $\hat{\mathbf{X}} = f(\mathbf{XU})\mathbf{V}$ is **reconstruction error measure.**
  - **U**: Weights (input $\rightarrow$ hidden),
  - **V**: Weights (hidden $\rightarrow$ output),

- **XU** projects **X** into K dimensional space
  $\mathbf{U} \in \mathbf{R}^{\mathbf{D}\times\mathbf{K}}$, $\mathbf{V} \in \mathbf{R}^{\mathbf{K}\times\mathbf{D}}$.

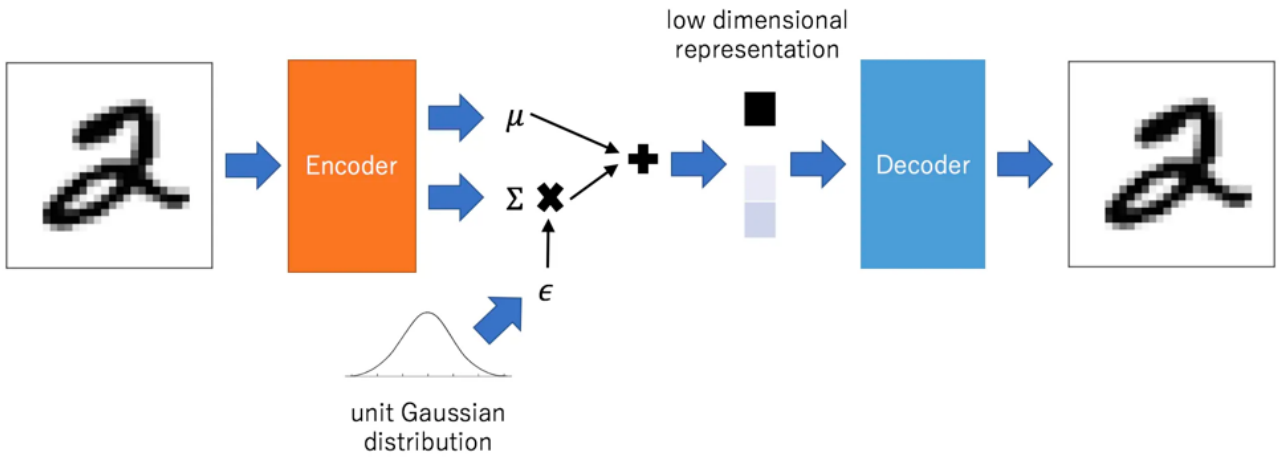
Auto-encoders for anomaly detection.

- Auto encoder with single **hidden layer**

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X} - \mathbf{f}(\mathbf{XU})\mathbf{V}\|_{\mathbf{F}}^{\mathbf{2}} \tag{1}$$

- $\hat{\mathbf{X}} = f(\mathbf{XU})\mathbf{V}$ is **reconstruction error measure.**
  - **U**: Weights (input $\rightarrow$ hidden),
  - **V**: Weights (hidden $\rightarrow$ output),
  - activation function: $f : \mathbb{R} \rightarrow \mathbb{R}$
- **XU** projects **X** into K dimensional space
  $\mathbf{U} \in \mathbf{R}^{\mathbf{D}\times\mathbf{K}}$, $\mathbf{V} \in \mathbf{R}^{\mathbf{K}\times\mathbf{D}}$.
- Non linear projection: **sigmoid** $f(\cdot) := (1 + \exp(-a))^{-1}$

Variational Auto-encoder (VAE)



- Map Input to distribution instead of a vector.
- Bottleneck embeddings consists of two separate vectors (mean, standard deviation) of distribution.
- Sampled vector is vector to fed into decoder for reconstruction.

Training Variational Auto-encoder (VAE)

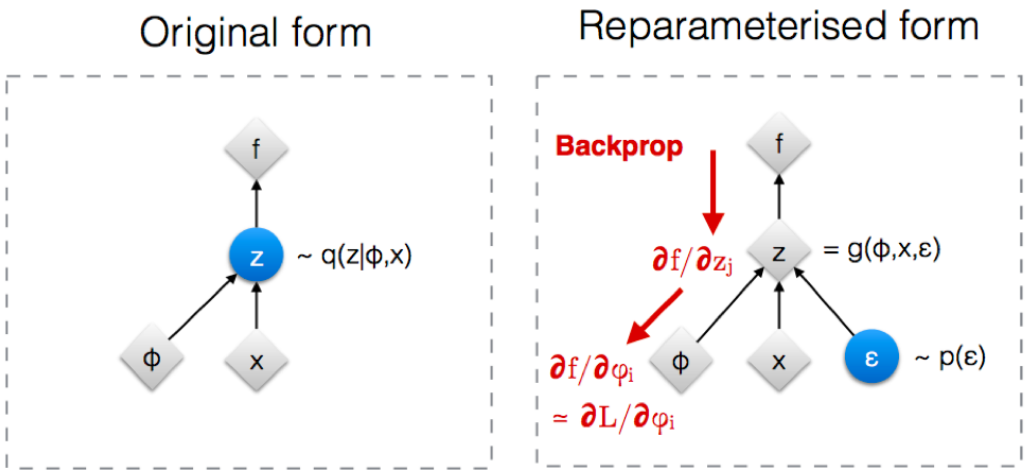
- Loss function consists of two terms (reconstruction loss, KL divergence term)

$$-\underbrace{\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z})\right]}_{\text{reconstruction error}}+\underbrace{\mathbf{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)}_{\text{regularization}}$$

- Reconstruction loss consists of expectation term since we are learning a distribution instead of latent vector.
- KL divergence enforces learned latent distribution to be relatively close to Normal distribution (mean, standard deviation: $\mu = 0, \sigma = 1$).
- Challenge: Backpropogation through sampling vector solved : Re-parameterization trick.

Training VAE: Re-parameterization trick

- ϵ introduced to separate the stochastic behaviour.



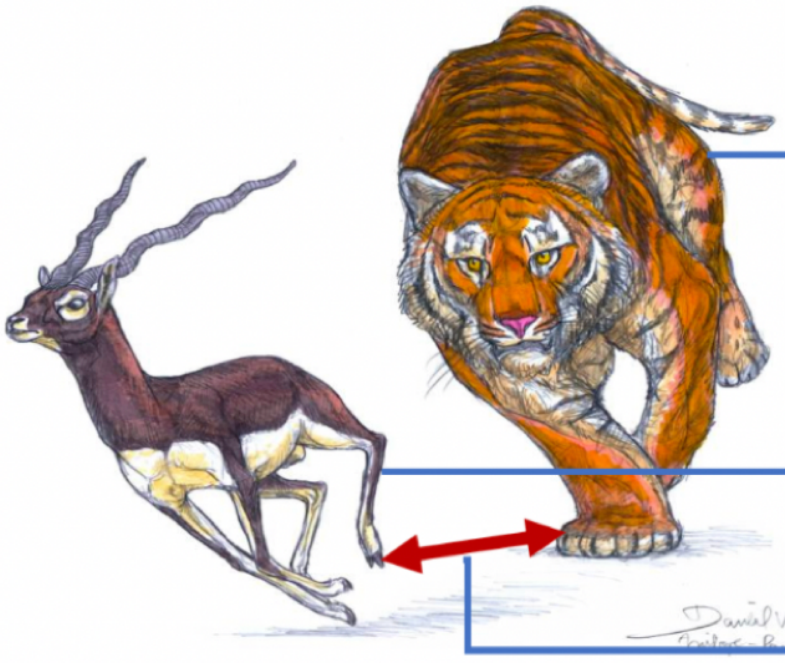
[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$$
$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$$

Limitations of VAE

- Challenges VAE:
  - KL divergence limits learning to Normal distributions (required for closed form solution).
  - Encoded latent distribution and Decoder may be separated far away.
  - Proposed Solution:
    - Use Jensen-Shannon Divergence (JSD) : **Adversarial Auto-encoders**
    - Wasserstein distance based loss function: **Wasserstein Auto-encoders**

Working Principle of Adversarial Networks



Generator
[Tries to generate distribution (minimize) closer to p(z)]

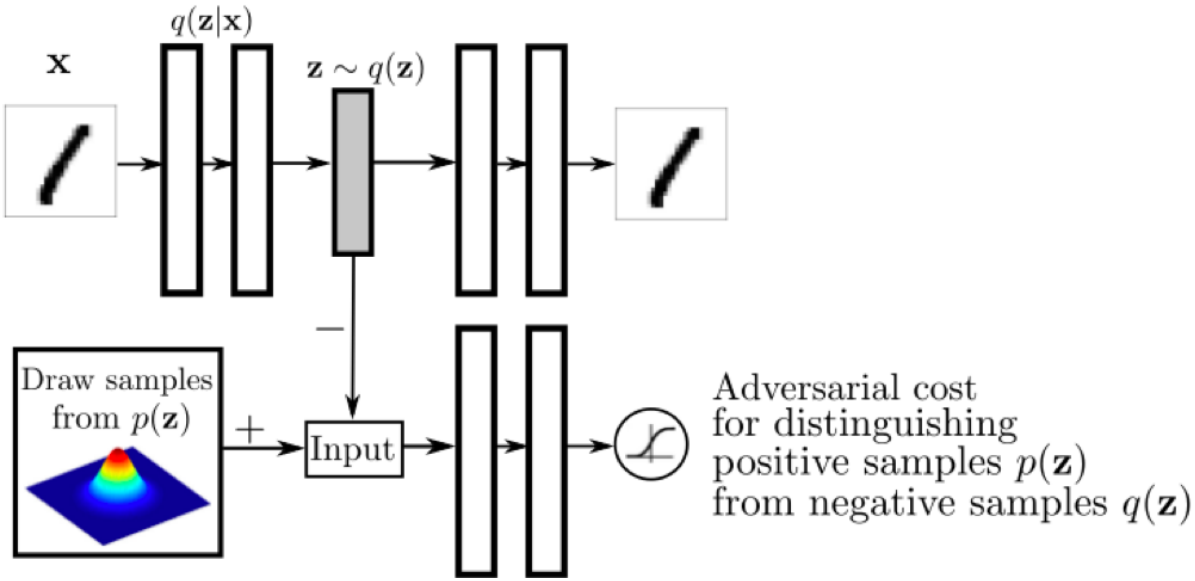
Discriminator
[Tries to maximize separation between generated and input distribution p(z)]

Distance of Separation

$$\min_{\text{Tiger}} \max_{\text{Deer}} \{ \text{Distance of separation} \}$$

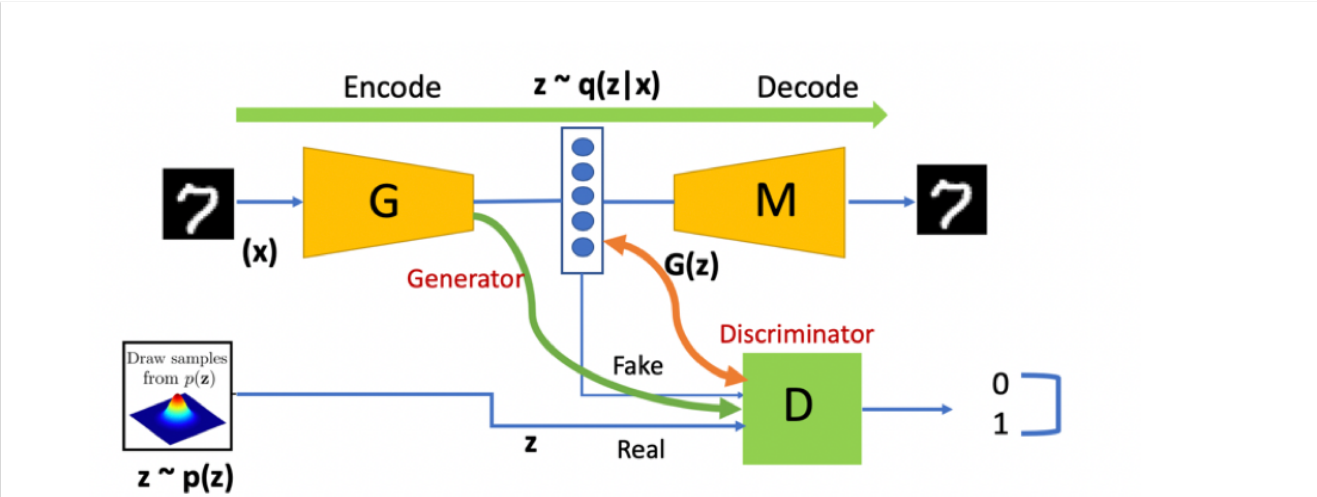
Source: Gredinia from deviantart.com

Adversarial Auto-Encoders (AAE)



- Leverage generator and discriminator networks
- Train VAE and minimize reconstruction loss
- Tune Encoder(Generator) to produce latent embeddings following input p(z) .
- Discriminator aids to tune encoder weights to learn p(z) through back-propagation feedback.

Training AAE



$$\min_G \max_D \mathrm{E}_{\mathrm{x}\sim p_{\mathrm{data}}}[\log D(\mathbf{x})] + \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))]$$

Reconstruction error

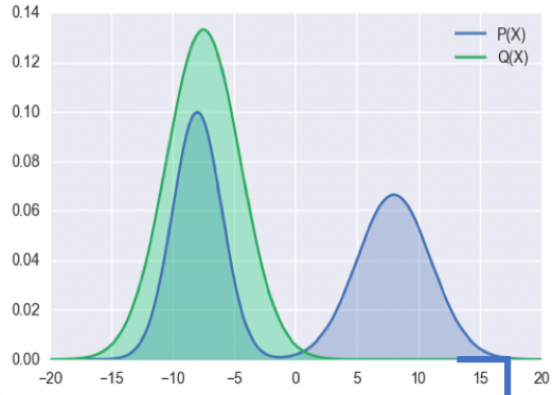
Regularization

- Training is composed of
  - Reconstruction : Autoencoder
  - Regularization : JSD Divergence

Research Path VAE to WAE

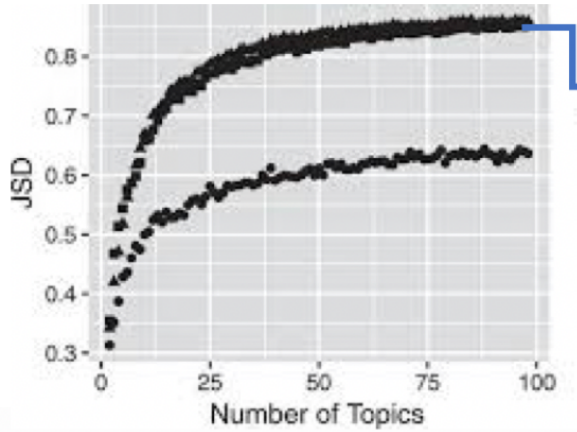
- Kullback–Leibler (KL) Divergence: VAE
- Jensen-Shannon Divergence (JSD): AAE | GAN
- Wasserstein Distance : WAE
- **VAE $<$ AAE $<$ WAE**

Motivation 1: Wasserstein Auto-Encoders (WAE)



$$D_{KL}(p\|q) = \int_x p(x) \log \frac{p(x)}{q(x)} dx \quad \rightarrow \infty$$

(a) KL Divergence



$$D_{JS}(p\|q) = \frac{1}{2} D_{KL}(p\|\frac{p+q}{2}) + \frac{1}{2} D_{KL}(q\|\frac{p+q}{2})$$

(b) Jensen Shannon Divergence (JSD)

- Need stable metric for probability distribution comparision.

Motivation 2: Wasserstein Auto-Encoders (WAE)

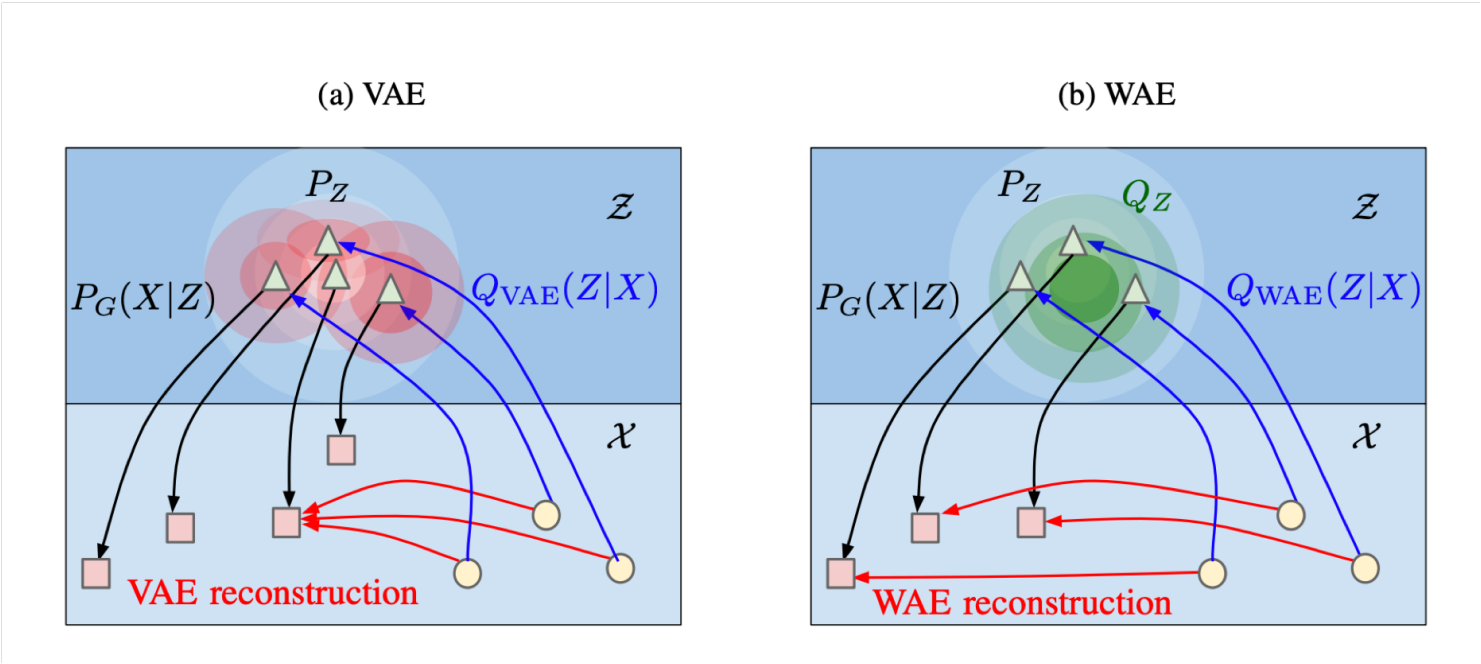
$$\min_G \max_D \mathrm{E}_{\mathrm{x}\sim p_{\mathrm{data}}}[\log D(\mathbf{x})] + \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))]$$

$$D^* = \frac{p_{data}}{p_{gen} + p_{data}}$$

$$L(G, D^*) = JSD(p_{data}||p_{gen}) - 2log(2)$$
$$\nabla(L(G, D^*) = 0$$

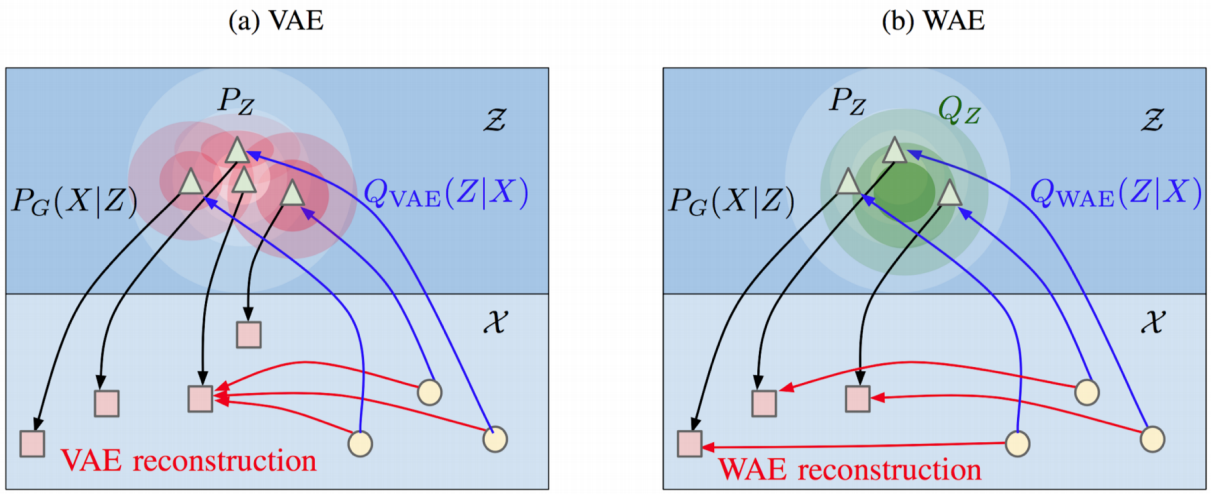
- No learning taking place in initial stages of backpropogation (Min, Max Game).

Wasserstein Auto-Encoders (WAE)



- VAE, $Q(Z|X)$ matches the prior P_z for each point in input.
- WAE, matches the prior in an aggregated global sense to $Q(Z|X)$ with penalized form of the Wasserstein distance.
- WAE latent code for different examples are mapped to be well separately.

Training (WAE)



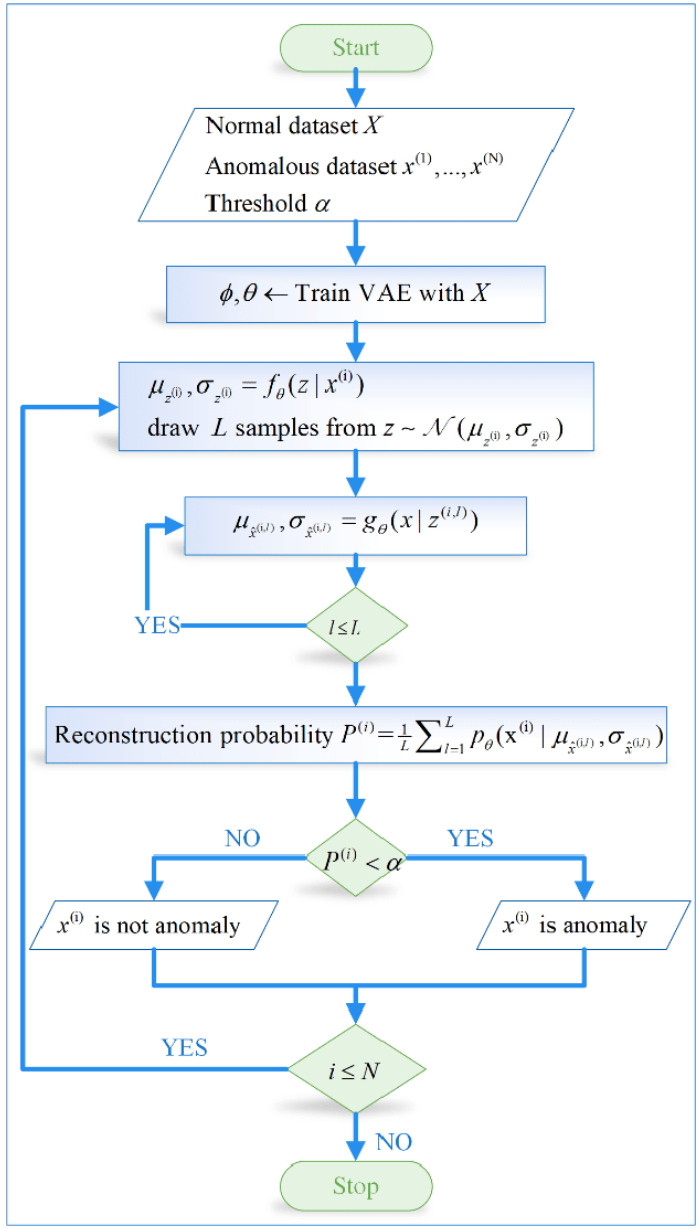
VAE: $\inf_{P_G} \inf_{Q \in \mathcal{Q}} \mathbb{E}_{P_X} \mathbb{E}_{Q(Z|X)}[-\log p_G(X|Z)] + \mathbb{E}_{P_X}[\mathrm{KL}(Q(Z|X), P_Z)]$

WAE: $\inf_{P_G} \inf_{Q \in \mathcal{Q}} \mathbb{E}_{P_X} \mathbb{E}_{Q(Z|X)}[c(X, G(Z))] + \lambda \cdot \mathcal{D}_Z(\mathbb{E}_{P_X}[Q(Z|X)], P_Z)$

- WAEs can be used with any encoders and decoders. .
- Used with any prior distributions P_Z with D_Z = arbitrary divergence (P_Z, Q_Z)
- WAEs can be readily used with any reconstruction cost function.

Anomaly Detection using VAE || AAE || WAE

- Flow chart for anomaly detection [Sultan Zavrak et.al]
- Thesis: Anomalies have higher reconstruction error.



Anomaly Detection using VAE || AAE || WAE

Input : Set of points $\mathbf{X} = \{x_1, ..., x_N\}$, **input** $\mathcal{G} = \left(\mathbf{G}(m)\right)_{m \in \{1,2,...,M\}}$ normal reference images.

Output: Anomaly scores **S** for input points **X**

$f_\phi, g_\psi \leftarrow$ train a VAE || AAE using the input points **X**

for *(m = 1 to M)* **do**

  for *(n = 1 to N)* **do**

    $\mu_z(i), \sigma_z(i) = f_\phi(z|x^{(i)})$

  end

  $(\mu_z,\ \sigma_z) = \sum_{n=1}^{N}(\mu_z(i), \sigma_z(i))$

  draw a sample from $z \sim \mathcal{N}(\mu_z,\ \sigma_z)$

  reconstruct sample using decoder $X_m^{(ref)} = g_\psi(x|z)$

  `/* compute distance this reference group instance` $X_m^{(ref)}$ `*/`

  for *(j = 1 to N)* **do**

    compute the score $s_j = d(X_m^{(ref)}, x_j)$

  end

  sort the images by descending order of score

  S= $(s_j >, s_{j-1} > ..., s_N)$

  points which are farthest from $X_m^{(ref)}$ **constitute group anomalies.**

end

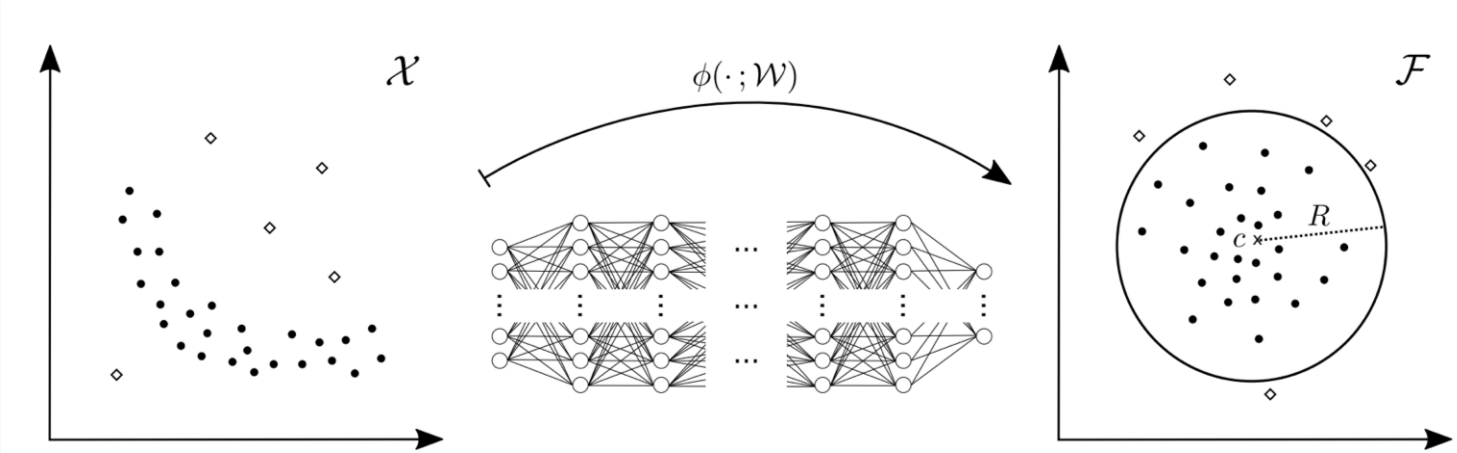
return S

Algorithm: Anomaly detection using generative models VAE || AAE

Notebooks

- Notebook : AE || VAE || AAE || WAE.

Deep Support Vector Data Description (Deep SVDD)



- Deep SVDD learns a neural network transformation into a hypersphere.
- Mappings of normal examples fall within, whereas mappings of anomalies fall outside the hypersphere.

Training Deep SVDD

- Soft Boundary Deep SVDD: minimizing the volume of a data-enclosing hypersphere in output.

$$\min_{\mathcal{W}} \quad \frac{1}{n}\sum_{i=1}^{n}\|\phi(\boldsymbol{x}_i;\mathcal{W})-\boldsymbol{c}\|^2+\frac{\lambda}{2}\sum_{\ell=1}^{L}\|\boldsymbol{W}^\ell\|_F^2.$$

- Deep SVDD.

$$\begin{aligned}\min_{R,\mathcal{W}} \quad & R^2+\frac{1}{\nu n}\sum_{i=1}^{n}\max\{0,\|\phi(\boldsymbol{x}_i;\mathcal{W})-\boldsymbol{c}\|^2-R^2\}\\ & +\frac{\lambda}{2}\sum_{\ell=1}^{L}\|\boldsymbol{W}^\ell\|_F^2.\end{aligned}$$

- Anomaly Score:

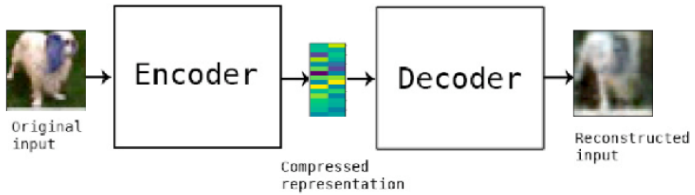
$$s(\boldsymbol{x})=\|\phi(\boldsymbol{x};\mathcal{W}^*)-\boldsymbol{c}\|^2$$

- $\phi(x_i, W^*)$: its associated feature mapping; λ is tuning parameter c, R center and radius of hypersphere,

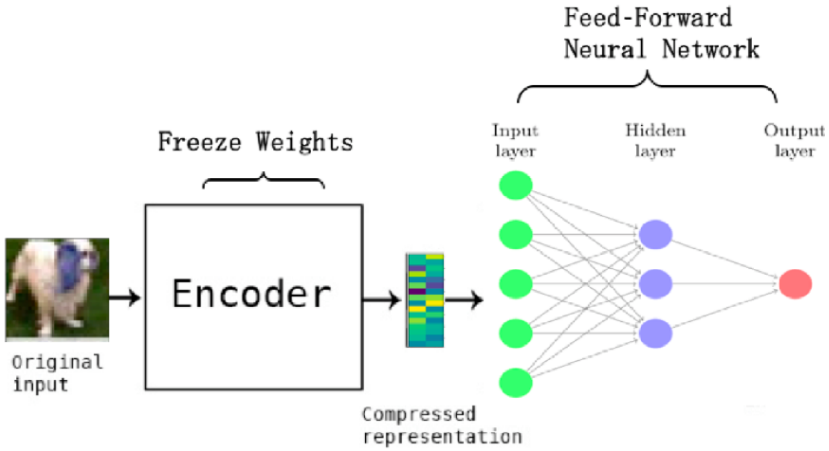
Notebook

- Notebook : Deep SVDD.

One-Class Neural Networks (OCNN)



(a) Autoencoder.



(b) One-class neural networks.

- OC-NN objective is customized for anomaly detection.
- OC-NN uses one-class SVM (OC-SVM) like loss function to train a neural network.

Training OCNN

- One-class Neural Network (OCNN).

$$\min_{w,V,r} \frac{1}{2}\|w\|_2^2 + \frac{1}{2}\|V\|_F^2 + \frac{1}{\nu} \cdot \frac{1}{N} \sum_{n=1}^{N} \max(0, r - \langle w, g(V\mathbf{X}_{n:})\rangle) - r$$

- OCNN Algorithm for anomaly detection:

Algorithm 1 one-class neural network (OC-NN) algorithm

Input: Set of points $\mathbf{X}_{n:}$, $n : 1, ..., N$
Output: A Set of decision scores $S_{n:} = \hat{y}_{n:}$, n: 1,...,N for X
Initialise $r^{(0)}$
$t \leftarrow 0$
while (no convergence achieved) **do**
    Find $(w^{(t+1)}, V^{(t+1)})$ ▷ Optimize Equation 4 using BP.
    $r^{t+1} \leftarrow \nu^{\text{th}}$ quantile of $\{\hat{y}_n^{t+1}\}_{n=1}^N$
    $t \leftarrow t + 1$
end
Compute decision score $S_{n:} = \hat{y}_n - r$ for each $\mathbf{X}_{n:}$
if $(S_{n:} \geq 0)$ **then**
    $\mathbf{X}_{n:}$ is normal point
else
    $\mathbf{X}_{n:}$ is anomalous
return $\{S_n\}$

- Key insight is to replace scalar dot product with feature representations within neural network

Notebook

- Notebook : One-class Neural Network (OCNN).

References

- Borghesi, Andrea, et al. "Anomaly detection using autoencoders in high performance computing systems." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. 2019.
- Anomaly scores for generative models *Vaclavet.al* *https://arxiv.org/pdf/1905.11890.pdf*
- Ruff, Lukas, et al. "Deep one-class classification." International conference on machine learning. 2018.
- Chalapathy, Raghavendra, Aditya Krishna Menon, and Sanjay Chawla. "Anomaly detection using one-class neural networks." arXiv preprint arXiv:1802.06360 (2018).
- Merrill, Nicholas, and Azim Eskandarian. "Modified Autoencoder Training and Scoring for Robust Unsupervised Anomaly Detection in Deep Learning." IEEE Access (2020).