Deep Learning for Anomaly Detection

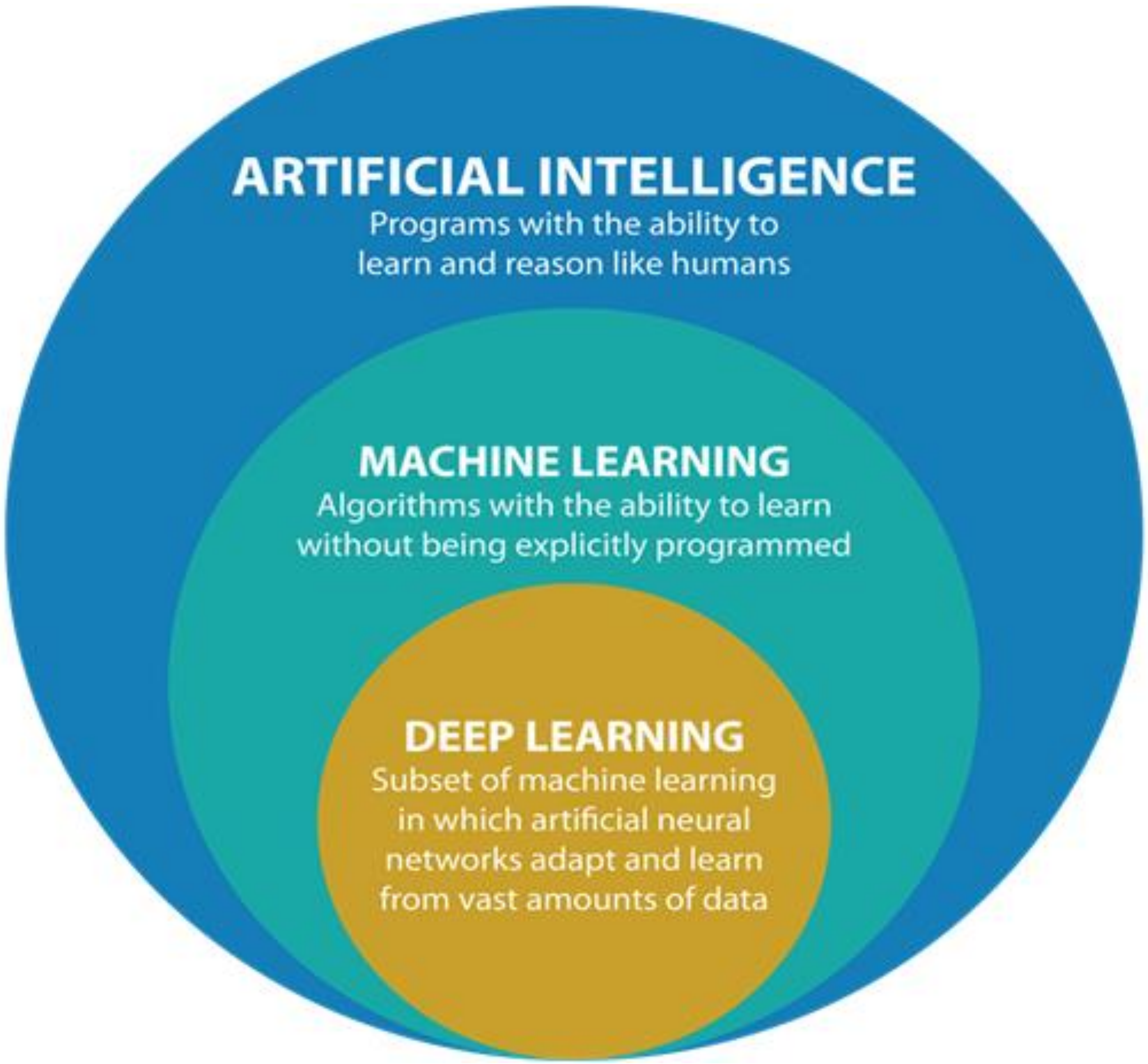
Meryem Janati Idrissi

Ismail Berrada

JDD'22

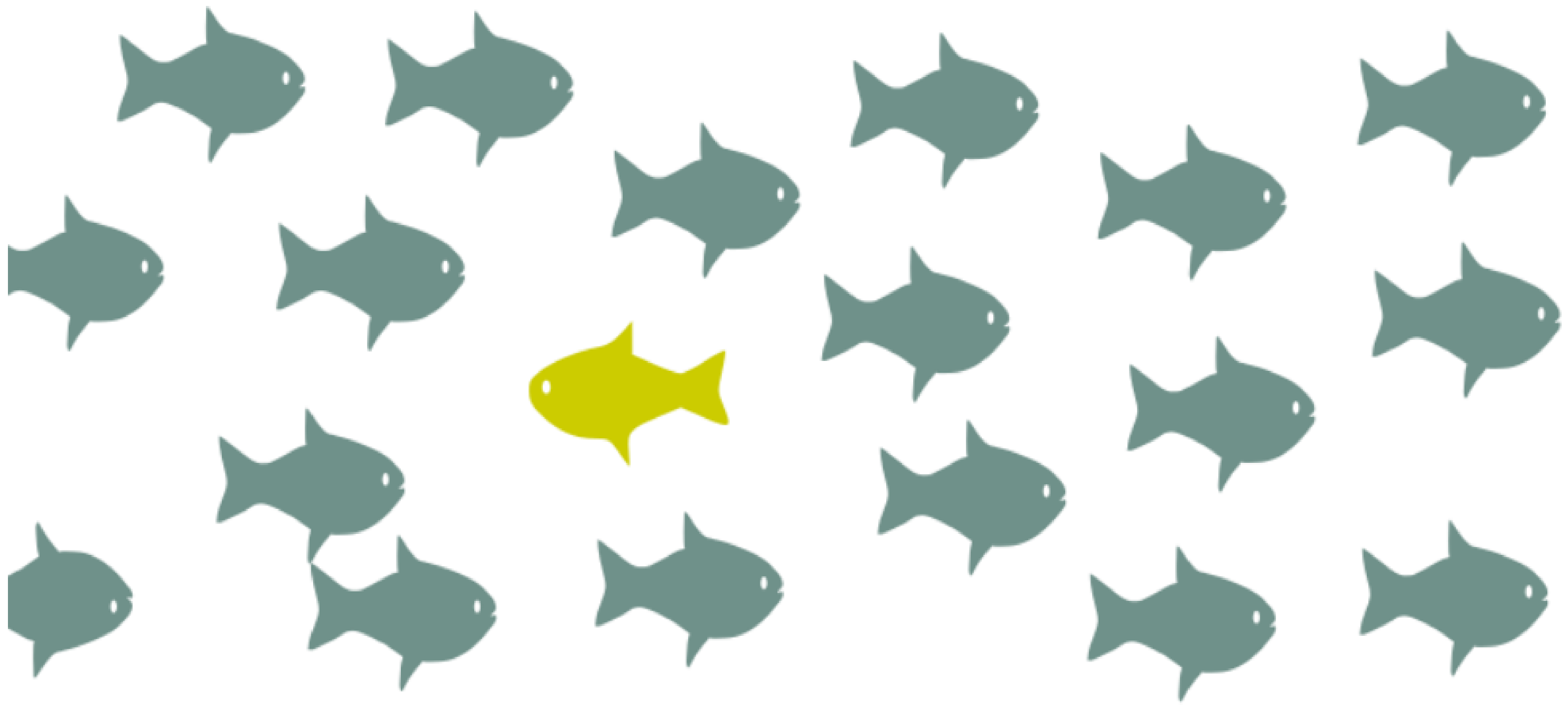
Table of Contents

Deep Learning



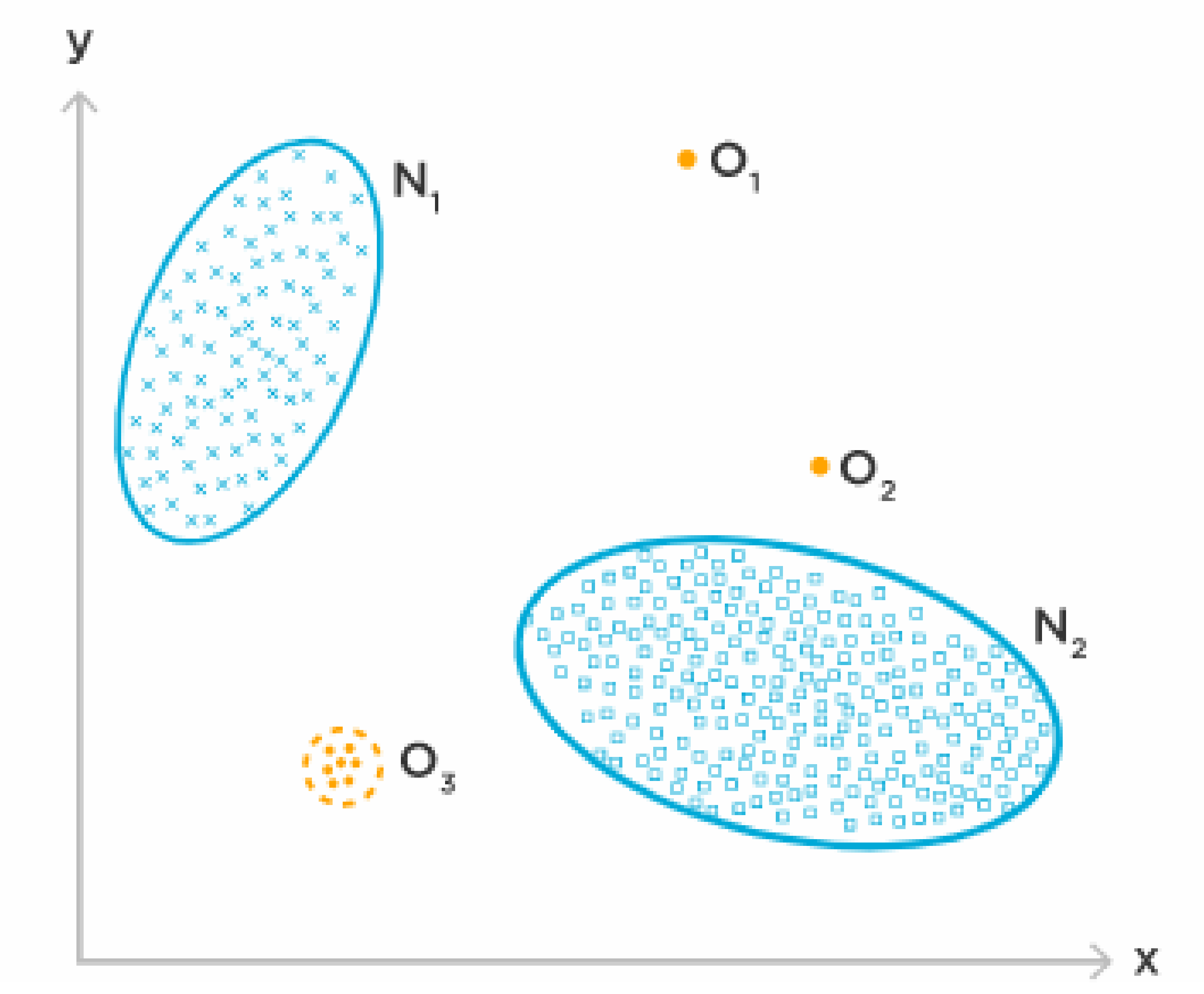
Anomaly Detection

- In recent years, anomaly detection has become increasingly important in a variety of domains in business, science and technology.
- Anomalous events occur relatively infrequently, however, when they do occur, their consequences can be quite dramatic.



What is an Anomaly?

- Anomalies are data points within the datasets that appear to deviate markedly from expected outputs.
- An anomaly is an observation which deviates so much from the other observations which raises suspicion of it being generated through a different mechanism.
- Anomaly Detection is to identify patterns in a given dataset that don’t confirm to expected behavior.

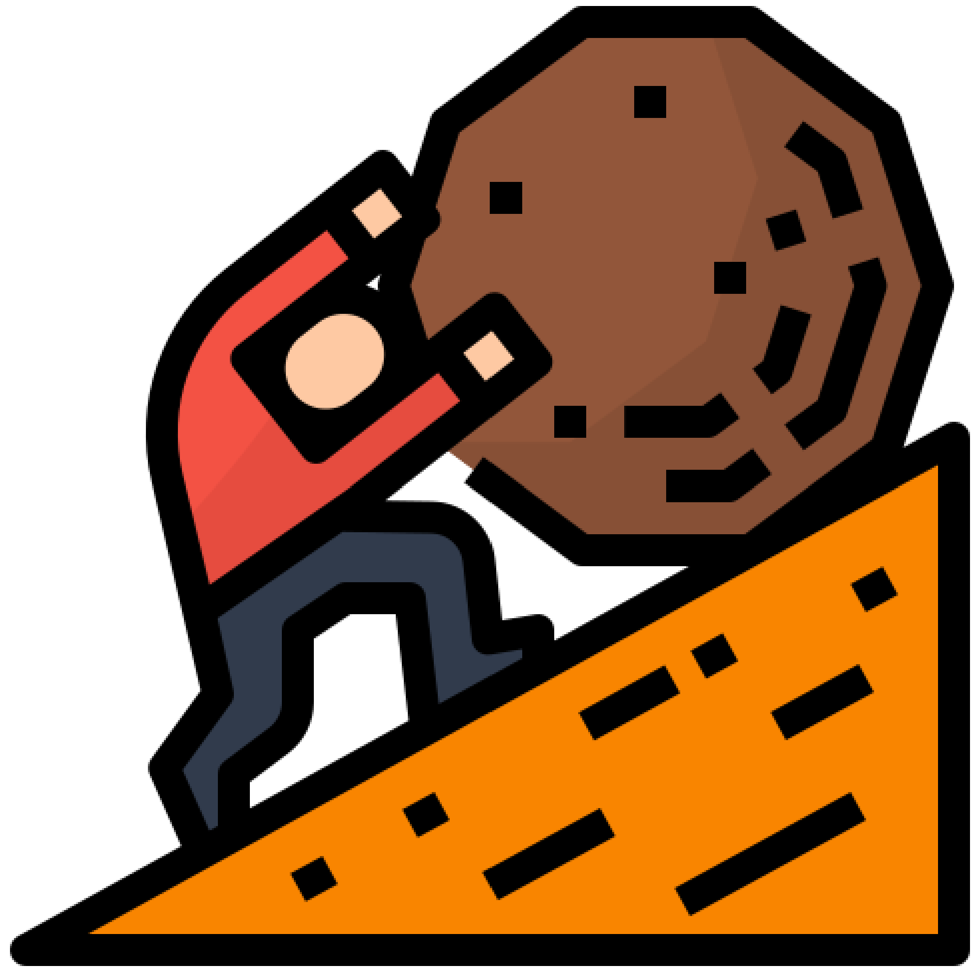


Applications of AD

Applications of anomaly detection include fraud detection in financial transactions, fault detection in manufacturing, intrusion detection in a computer network, monitoring sensor readings in an aircraft, spotting potential risk or medical problems in health data, and predictive maintenance.

What makes AD a hard problem?

- Lack of labelled data. Anomalies are hard to detect manually. As a result , labeling them is not feasible.
- Traditional methods don’t scale as the size of data increases.
- Traditional ML methods have limitations in extracting features from the high dimensional data. A lot of manual input is required to make the data ready for machine learning.



Why Deep learning for Anomaly Detection?

- Deep learning is highly scalable, especially because of modern distributed computing methods.
- Deep Learning methods have proven capabilities in image, text and speech recognition.
- They can efficiently learn information/features from data with minimum human intervention.

Generative Adversarial Network (GAN)

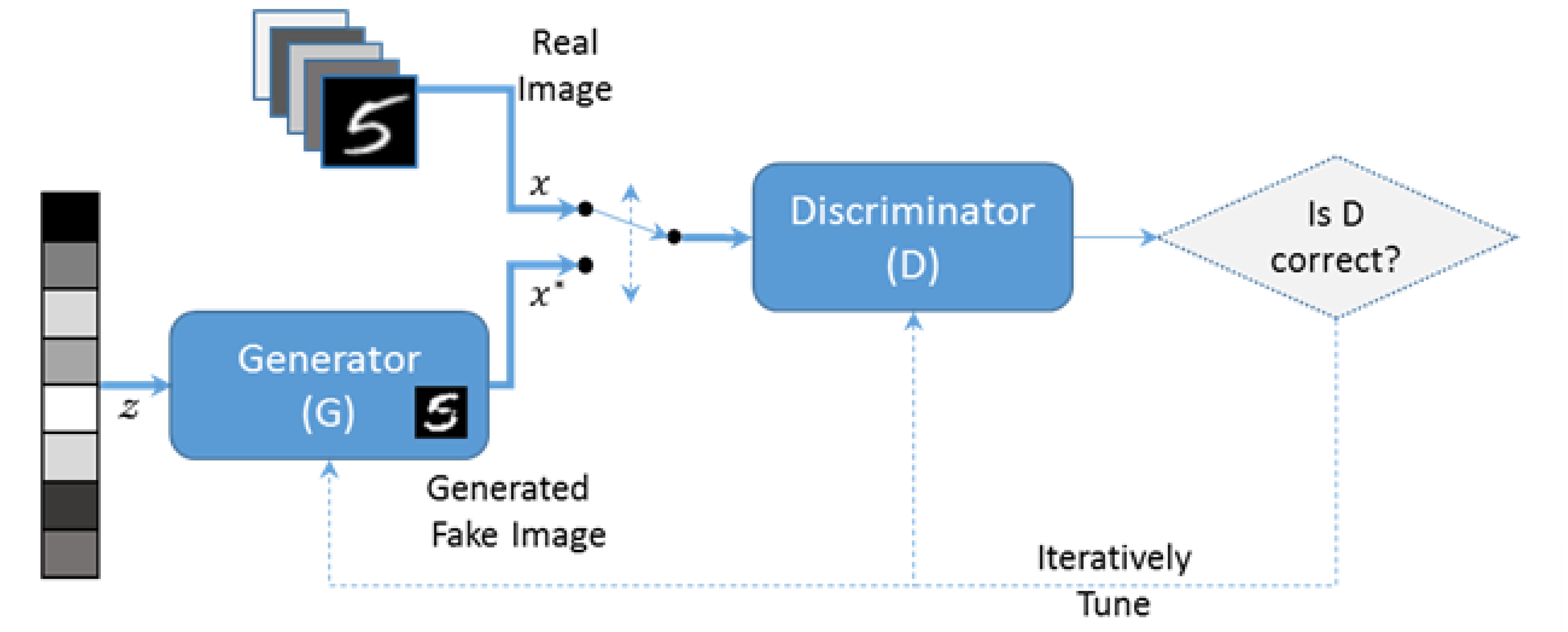
A Fraud (Generator)

Cop (Discriminator)

How does GAN works

Two components:

- **Generator (G):** Learns input data X representation.
- **Discriminator (D):** Distinguish input data and samples from generator.



D and G play a two-player minimax game with value function V(G,D).

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[1 - D(G(z))]$$

Source: Unsupervised and Semi-Supervised Learning with Categorical Generative Adversarial Networks | Sanghoon's Blog

Autoencoders (AE)

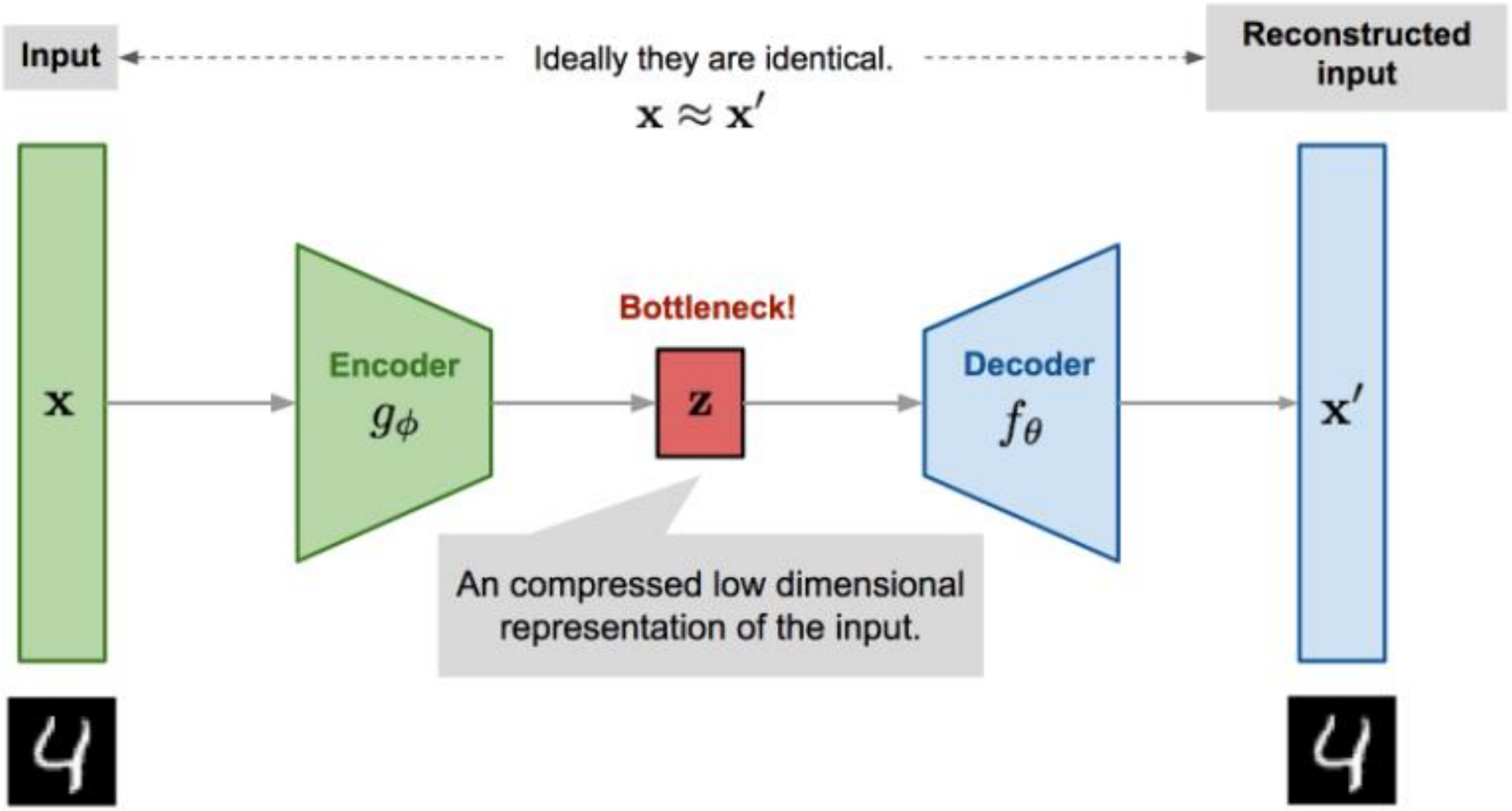
Two components:

- **Encoder:** Learns a representation or an encoding of a set of data in an unsupervised manner.
- **Decoder:** it recovers the data from the code, likely with larger and larger output layers.

The goal is to minimize the reconstruction loss L(X, X'), the differences between the original input and the reconstruction.

MSE loss:

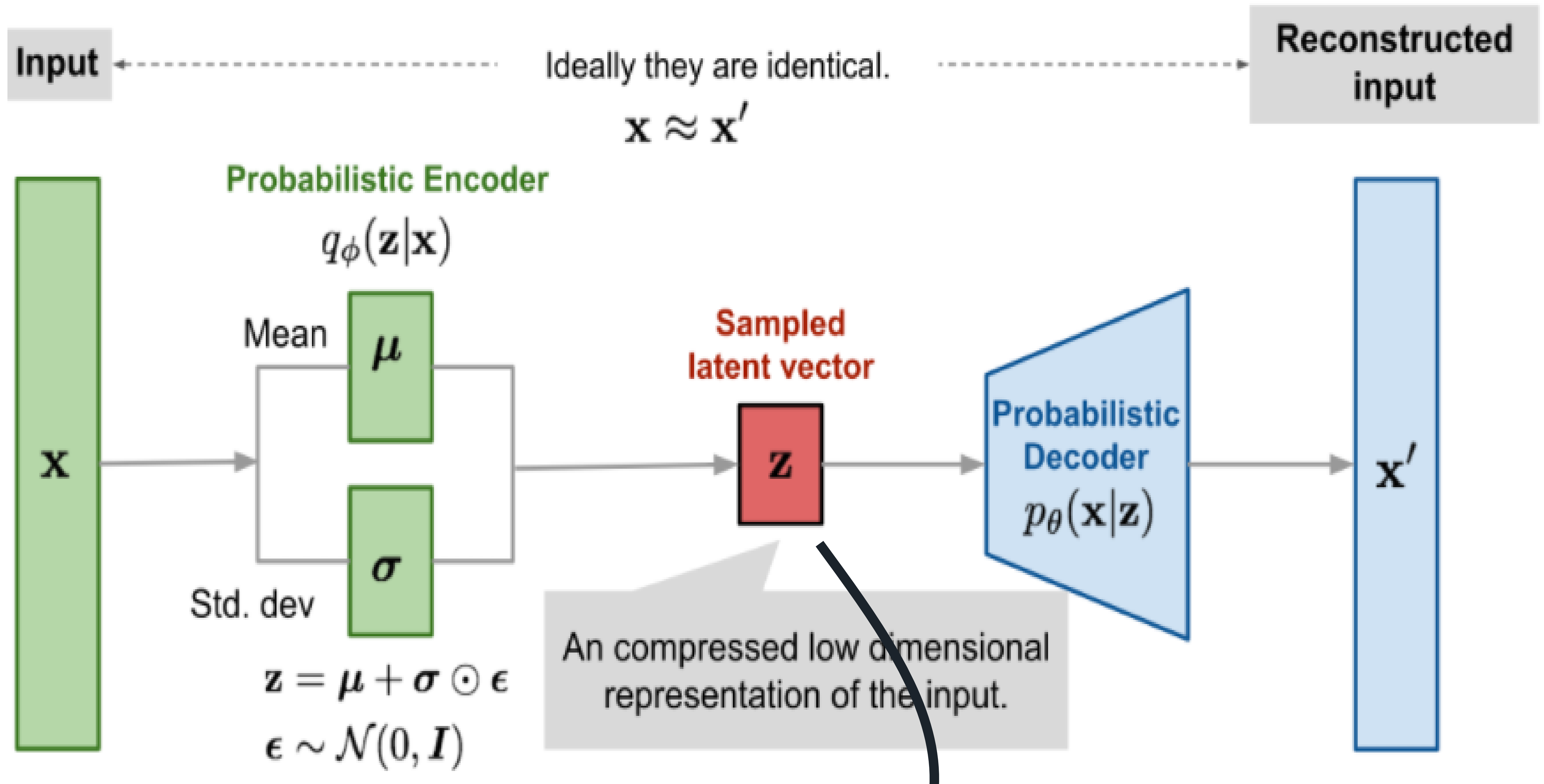
$$L_{AE}(\theta, \phi) = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)} - f_\theta(g_\phi(x^{(i)})\,))^2$$



Variational Autoencoders (VAE)

Two components:

- **Encoder:** Learns a representation or an encoding of a set of data in an unsupervised manner.
- **Decoder:** it recovers the data from the code, likely with larger and larger output layers.



The goal is to maximize the marginal likelihood:

$$\log p(x) \geq \log p(x) - KL\left(q_\phi(\mathrm{z}|\mathrm{x}) \,||\, \mathrm{p}(\mathrm{z}|\mathrm{x})\right)$$
$$= E_{z\sim q_\phi(z|x)} \log p_\theta(x|z) - KL(q_\phi(\mathrm{z}|\mathrm{x})||p(z))$$

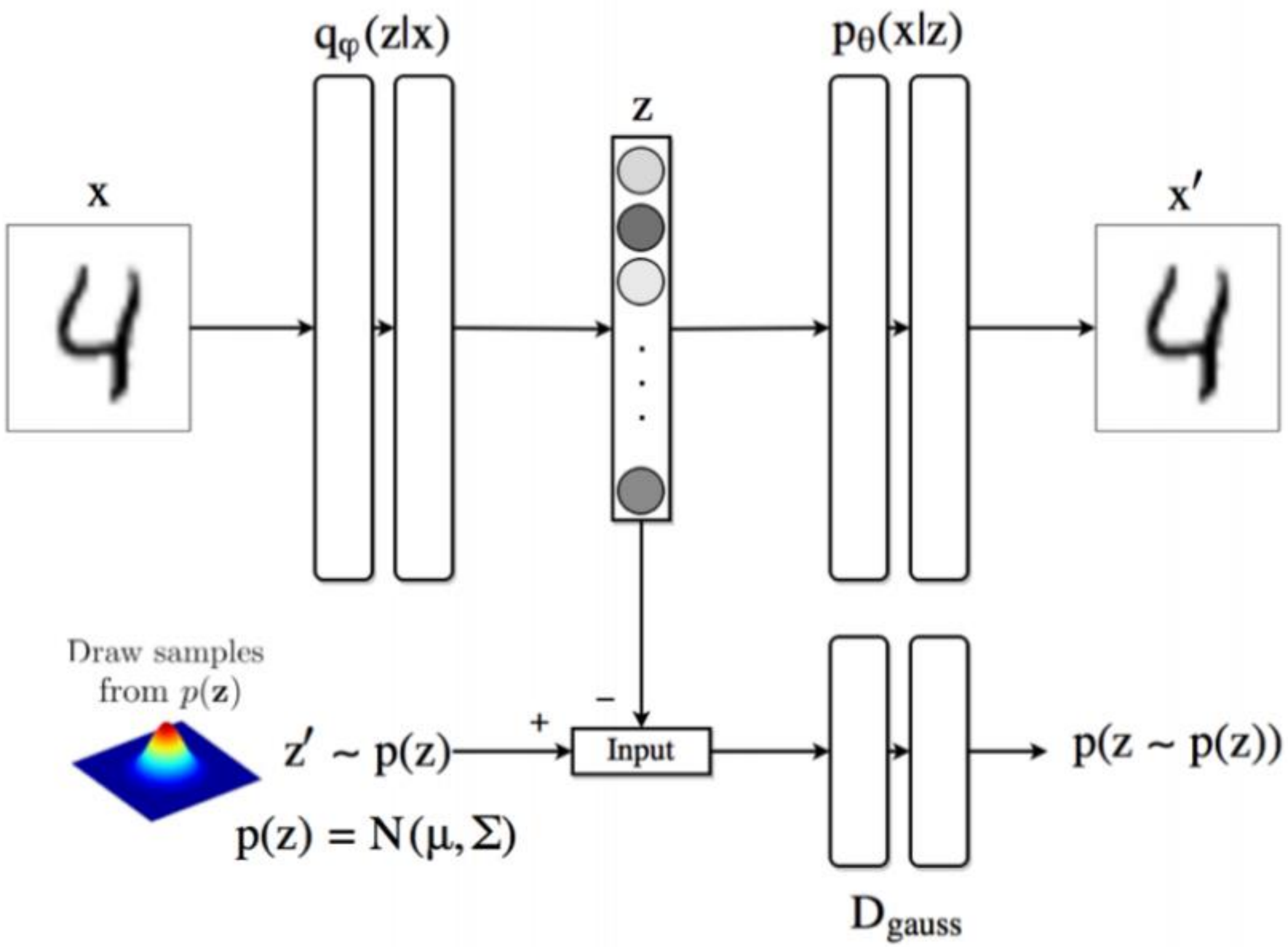
$\boldsymbol{z = q_\phi(z/x)}$ = N(μ, σ²)

Source: From Autoencoder to Beta-VAE | Lil'Log (lilianweng.github.io)

Adversarial Autoencoders (AAE)

Three components:

- **Encoder(Generator):** the encoder will take the input and transform it into a lower dimension (latent code z).
- **Decoder:** the decoder will take the latent code z and transform it into the generated image.
- **Discriminator:** the discriminator takes random vector z sampled from the chosen distribution (real) and also the encoded latent code z (fake) from the autoencoder as the input. It will check whether the input is real or not.



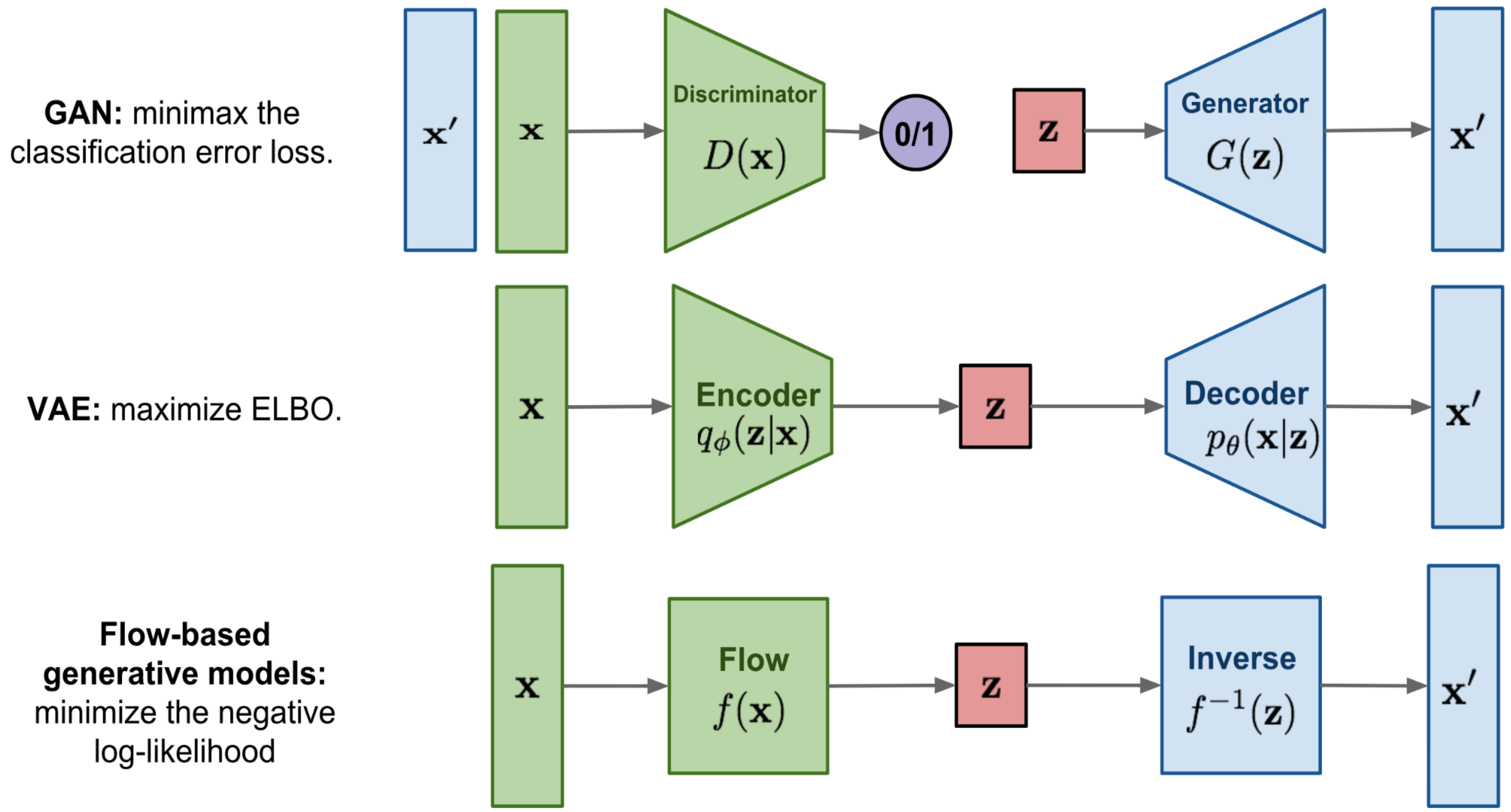
The goal :

$$\min_E \max_D V(E, D) = E_{z \sim p(z)}[\log D(z)] + E_{x \sim p_{data}(x)}[1 - D(E(x))]$$

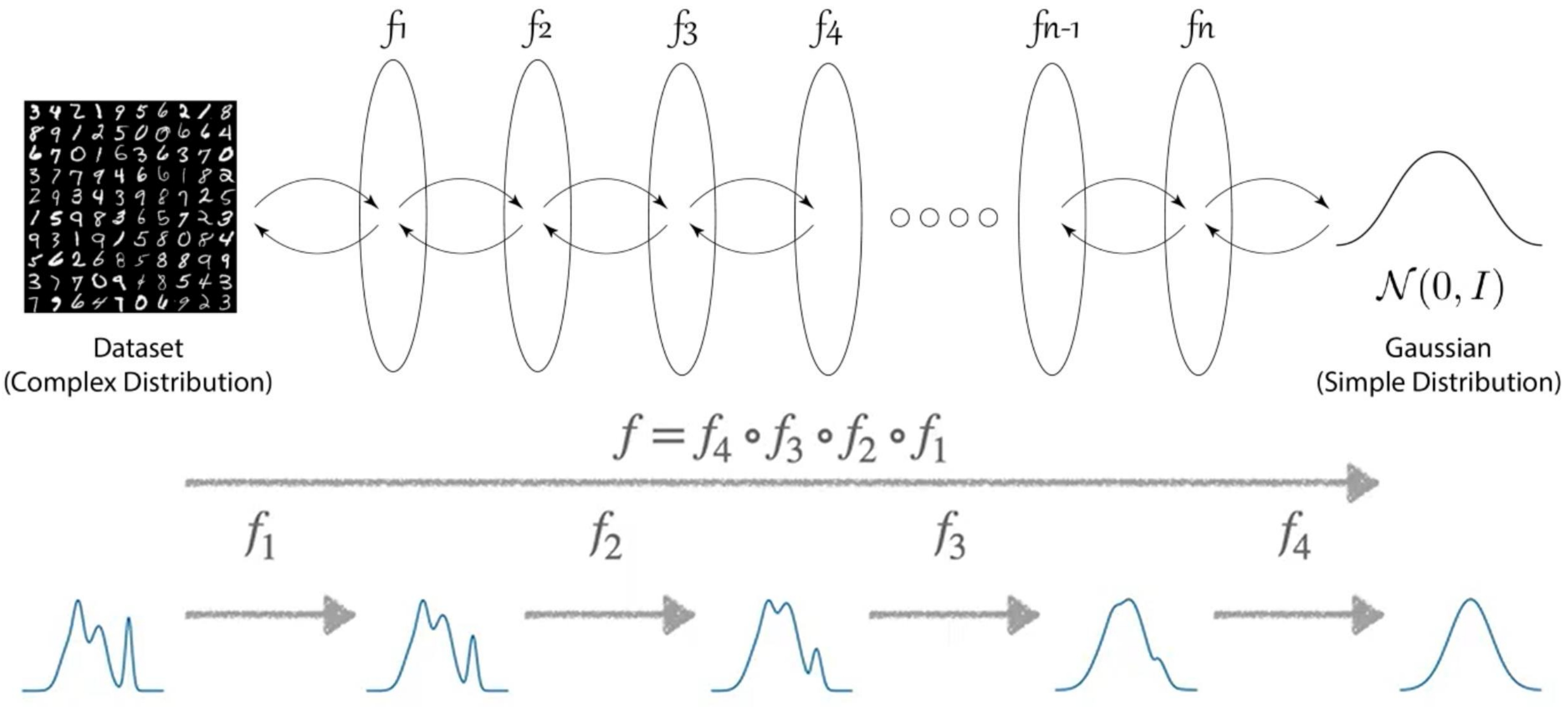
Source: [1511.05644] Adversarial Autoencoders (arxiv.org)

Normalizing Flows (NF)

GANs and VAEs have shown awe-inspiring results for learning complex data distributions and having simple inference methods. However, Neither of them explicitly learns the probability density function of real data p(x) because it is really hard!



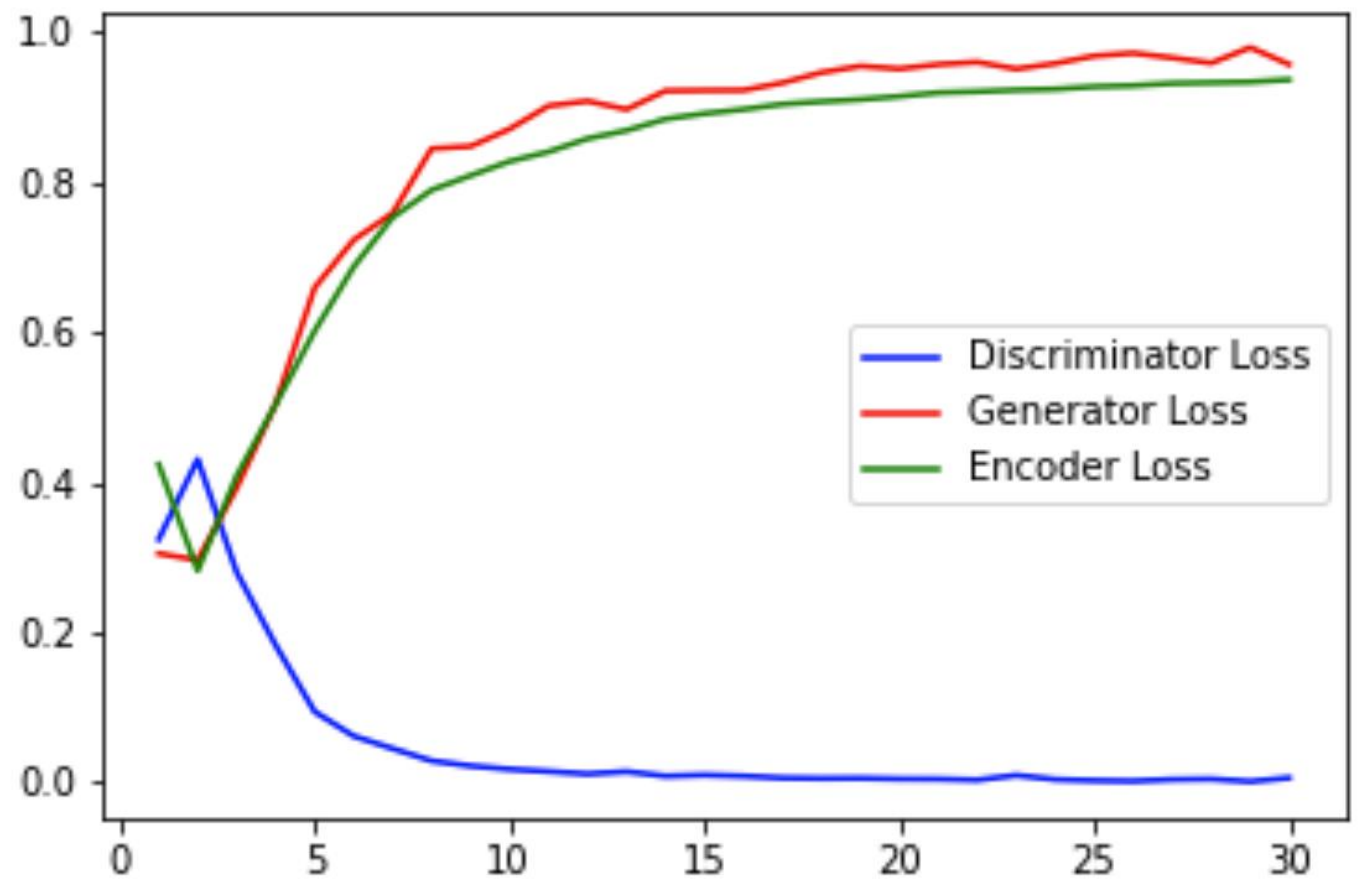
Normalizing Flows (NF)



A normalizing flow transforms a complex distribution into a simple one (Gaussian distribution) by applying a sequence of invertible transformation functions. Flowing through a chain of transformations, eventually we obtain a probability distribution of the final target variable.

Evaluation - BIGAN

BIGAN



	Precision	Recall	F1-Score
Benign	0.514	0.9744	0.673
Attack	0.937	0.293	0.447

USTC-TF16 dataset

Evaluation - Autoencoder

Dataset	Epochs	Train Loss	Validation Loss	Test Loss
CICIDS2017	100	0.01632	0.01631	0.01988
USTC-TF	100	0.01295	0.012977	0.01767

Training, validation and Test losses

	Precision	Recall	F1-Score
Benign	0.8437	0.8635	0.8535
Attack	0.3839	0.347	0.3645

CICIDS-2017 dataset

	Precision	Recall	F1-Score
Benign	0.7654	0.8761	0.817
Attack	0.8008	0.6498	0.7175

USTC-TF16 dataset

Evaluation - Variational Autoencoder

Dataset	Epochs	Train Loss	Validation Loss	Test Loss
CICIDS2017	100	0.03338	0.03345	0.03521
USTC-TF	100	0.03591	0.03583	0.03989

Training, validation and Test losses

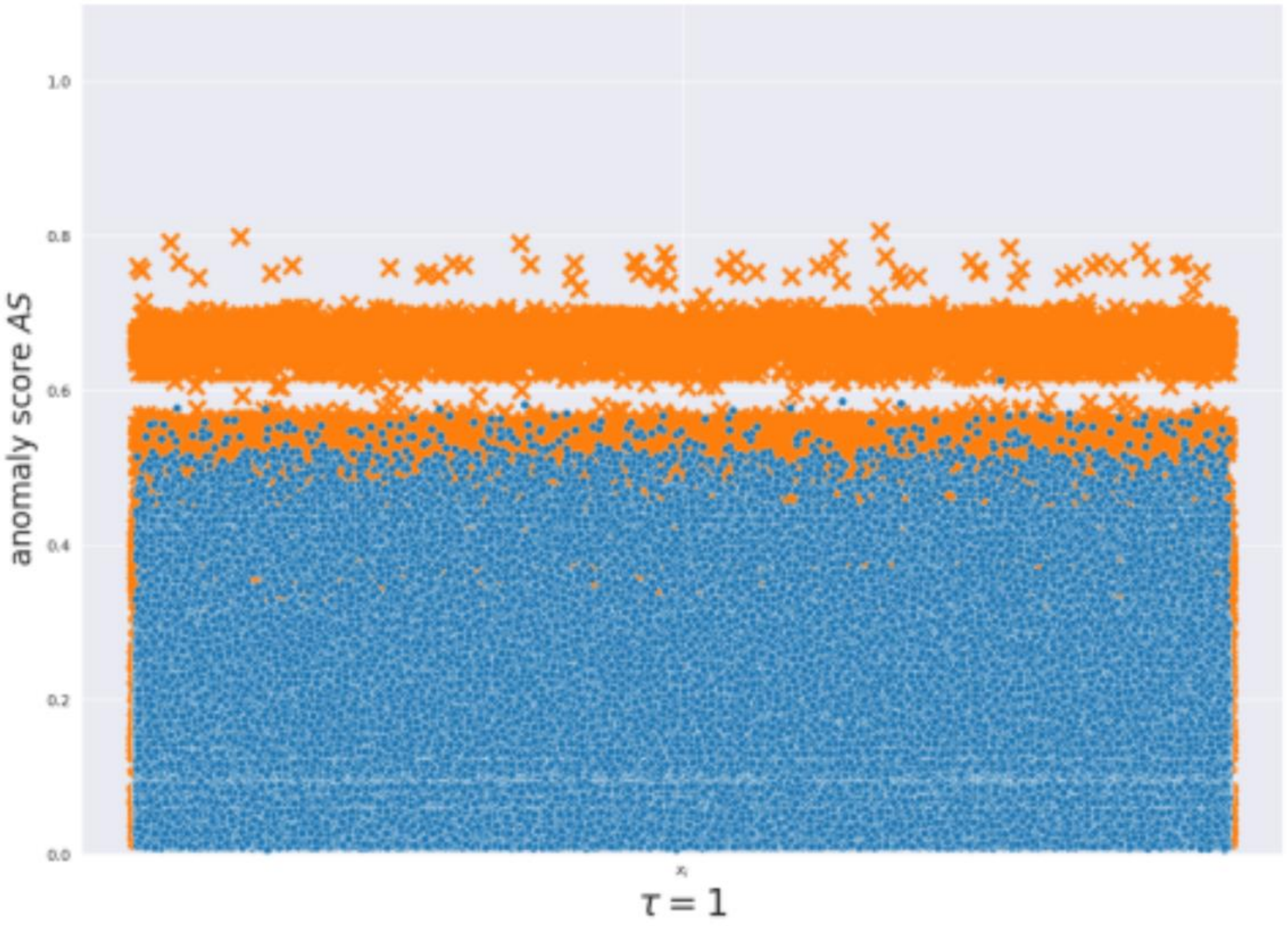
	Precision	Recall	F1-Score
Benign	0.8437	0.8635	0.8535
Attack	0.3839	0.347	0.3645

CICIDS-2017 dataset

	Precision	Recall	F1-Score
Benign	0.7506	0.8431	0.7942
Attack	0.7562	0.6347	0.6901

USTC-TF16 dataset

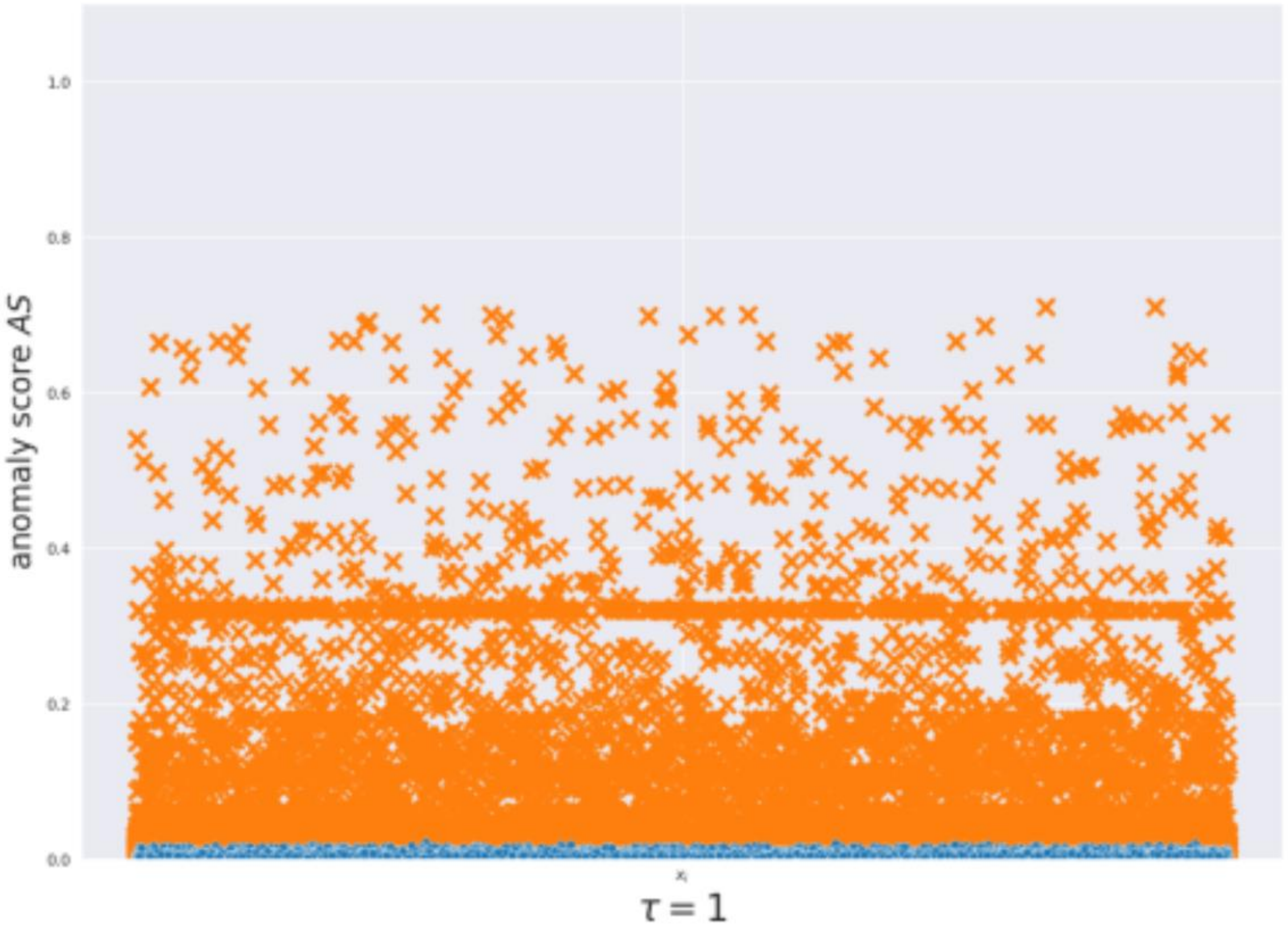
Evaluation– Adversarial Autoencoder



	Precision	Recall	F1-Score
Benign	0.862	0.848	0.855
Attack	0.421	0.45	0.43

CICIDS-2017 dataset

Evaluation – Adversarial Autoencoder



	Precision	Recall	F1-Score
Benign	0.89	0.99	0.938
Attack	0.86	0.84	0.908

USTC-TF16 dataset

Evaluation - Normalizing Flows

	Precision	Recall	F1-Score
Benign	0.8	0.89	0.84
Attack	0.97	0.94	0.96

CICIDS-2017 dataset

	Precision	Recall	F1-Score
Benign	1.00	0.92	0.96
Attack	0.94	1.00	0.97

USTC-TF16 dataset

Training NSF(Neural Spline Flows) on 5% of the datasets:

Conclusion & Perspectives

Short term perspectives:

- Add other datasets and compare the behavior of each model on all datasets.
- Try to generalize our models to make them work for different datasets.

Long term perspectives:

- Transfer the current work to a Federated setting and investigate the collaboration side of FL.

THANK YOU FOR YOUR ATTENTION!

Deep Learning for Anomaly Detection

Meryem Janati Idrissi

MOHAMMED VI POLYTECHNIC UNIVERSITY

Ismail Berrada

MOHAMMED VI POLYTECHNIC UNIVERSITY

JDD'22