UNIVERSIDADE DA CORUÑA

QGANS tecniques for anomaly detection

State of Art

Miranda Carou Laiño

Febrero 2025

Contents

1 Classic Generative Adversarial Networks

A classical generative adversarial Network (GAN) is a type of unsupervised machine learning initially developed by Ian Goodfellow. This generative model is composed by 2 neural networks (Generator G and Discriminator D), trained simultaneously, where the first one aim to capture the data distribution, while the other estimates the probability that a sample came from the training data rather than G. The generator can be thought of as analogous to a team of counterfeiters trying to produce fake currency and use it without detection while the discriminator is analogous to the police trying to detect the counterfeit currency. The original GAN framework (2014) poses this problem as a min-max game in which the two players compete against each other, playing the following zero-sum min-max game [5, 6].

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]. \tag{1}$$

To learn the generator's distribution p_g over data x, we define a prior on input noise variables $p_z(z)$, then represent a mapping to data space as $G(z;\theta_g)$, where G is a differentiable function represented by a multilayer perceptron with parameters θ_g . We also define a second multilayer perceptron $D(x;\theta_d)$ that outputs a single scalar. $D(x)$ represents the probability that x came from the data rather than p_g . We train D to maximize the probability of assigning the correct label to both training examples and samples from G. We simultaneously train G to minimize $\log(1 - D(G(z)))$.

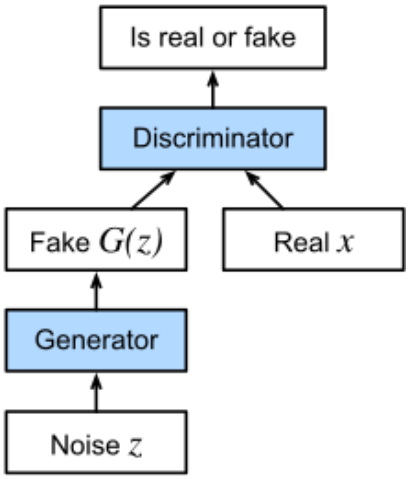


Figure 1: Generative Adversarial Networks basic workflow

The generative network generates candidates while the discriminative network evaluates them in terms of data distributions. Typically, the generative network learns to map from a latent space to a data distribution of interest, while the discriminative network distinguishes candidates produced by the generator from the true data distribution [6].

1.1 Conditional GANs

Conditional GANs are GANs extended to a conditional model proposed by Mirza and Osidero in 2014, conditioning either G or D on some extra information y, where y could be any auxiliary information. The conditioning can be performed by feeding y into both the discriminator and generator as an additional input layer. The generator combines the noise prior $p_z(z)$ and y in a joint hidden representation. In the discriminator x and y are presented as inputs to a discriminative function [7, 6].

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x|y)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z|y)))]. \tag{2}$$

1.2 BiGAN

The Bidirectional GANs extends the GAN framework including an encoder $E(x;\theta_E)$ that learns the inverse of the generator $E = G^{-1}$. This training process allows learning a mapping simultaneously from latent space to data and vice versa where the encoder E is a non-linear parametric function in the same way as G and D, and it can be trained using gradient descent. As in the conditional model, the discriminator must learn to classify not only real and fake samples, but pairs in the form $(G(z), z)$ or $(x, E(x))$ [7]. The training objective is:

$$\min_{G,E}\max_{D} V(D,G,E) = \mathbb{E}_{x\sim p_{\text{data}}(x)}\Big[\mathbb{E}_{z\sim p_E(z|x)}[\log D(x,z)]\Big] + \mathbb{E}_{z\sim p_z(z)}\Big[\mathbb{E}_{x\sim p_G(x|z)}[\log(1-D(x,z))]\Big]. \tag{3}$$

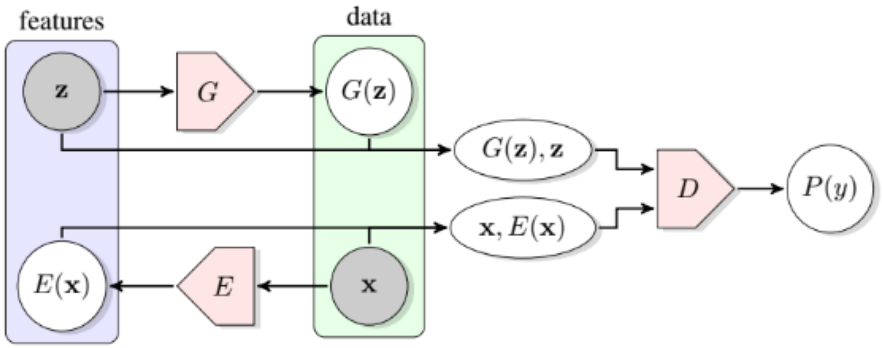


Figure 2: The structure of BiGAN proposed in (Donahue et al., 2016)

2 GANs for anomaly detection

Anomaly detection using GANs is an emerging research field where [8] was the first to proposed such a concept referred to as AnoGAN. Later related with the performance issues of AnoGAN, [9] propose a BiGAN-based approach of AnoGAN named as EGBAD (Efficient GAN Based Anomaly Detection) with a improvement in AnoGAN execution time. Recently, [10] propose a GAN + autoencoder based approach that exceed EGBAD performance from both evaluation metrics and execution speed.

2.1 AnoGAN

AnoGAN aim is to use a standard GAN, trained only on positive samples, to learn a mapping from the latent space representation z to the realistic sample $\hat{x} = G(z)$ and use this learned representation to map new, unseen, samples back to the latent space. Using this technique, training on normal samples only, makes the generator learn the the manifold χ of normal samples and how to generate it. When an anomalous image is encoded its reconstruction will be non-anomalous; hence the difference between the input and the reconstructed image will highlight the anomalies.

The mapping of input samples to a latent space is defined as an iterative process where the aim is to find a point z in the latent space that correspond to a generated value $G(z)$ that is similar to the query value x located on the manifold χ of the positive samples. The research process is defined as the minimization through $\gamma = 1, 2, \ldots, \Gamma$ back-propagation steps of the loss function defined as the weighted sum of the residual loss L_R and discriminator loss L_D.

The ***residual loss*** measures the dissimilarity between the query sample and the generated sample in the input domain space.

$$L_R(z_\gamma) = \|x - G(z_\gamma)\|_1. \tag{4}$$

The ***discriminator loss*** takes into account the discriminator response There are different ways to calculate the discriminator loss in a generative adversarial network. The first approach, inspired by a previous study, involves feeding the generated image to the discriminator and measuring the sigmoid cross-entropy based on the discriminator's confidence that the image came from the real data distribution. The second approach, inspired by another study, focuses on the feature matching loss, where features are compared from a layer of the discriminator to evaluate the similarity between the generated and input samples. The proposed loss function combines both approaches, with a weight parameter. This loss function is then used to calculate the anomaly score, which indicates the likelihood of a sample being anomalous. It is important to note that the minimization process is required for each input sample.

$$L_D(z_\gamma) = ||f(x) - f(G(z_\gamma)||_1 \tag{5}$$

so the proposed loss function is:

$$L(z_\gamma) = (1 - \lambda)L_R(z_\gamma) + \gamma L_D(z_\gamma) \tag{6}$$

Its value in step Γ matches the formulation of the anomaly score:

$$A(x) = L(z_\Gamma) \tag{7}$$

The value of $A(x)$ has no upper bound; high values correspond to a high probability that x is anomalous. It should be noted that the minimisation process is required for each input sample x.

PROS	**CONS**
GANs can be used for anomaly detection	Requires Γ optimization steps for every new input (poor time performance)
Introduced a new mapping scheme from latent space to input data space	The need for reverse mapping learning is not taken into account
Same mapping scheme to define anomaly score	The anomaly score is difficult to interpret, as it is not within the probability range

Table 1: Comparison of Pros and Cons of AnoGAN

2.2 EGBAD

Efficient GAN-Based Anomaly Detection brings the BiGAN architecture to the anomaly detection domain trying to solve the AnoGAN disadvantages with an encoder E able to map input samples to their latent representation during the adversarial training being strongly emphasized the importance of learning E jointly with G. The main contribution of the EGBAD is to allow computing the anomaly score without Γ optimization steps during the inference as it happens in AnoGAN [9].

2.3 GANomaly

Introduced by [10], this approach is inspired by AnoGAN, BiGAN and EGBAD. They train a generator network on normal samples to learn their manifold χ while at the same time an autoencoder is trained to learn how to encode the images in their latent representation efficiently. This approach only needs a generator and a discriminator as in a standard GAN architecture.

2.3.1 Generator network

The generator network consist of three elements in series, a G_E encoder, a G_D decoder, and an E encoder. The G_E encoder takes an input picture x and outputs an encoded version z, which is

then used by the G_D decoder to produce a reconstructed version of the input x. The reconstructed version $\hat{x}$ is further given as input to the E encoder to produce a final representation $\hat{z}$. The main contributions of this architecture are twofold. Firstly, the anomaly detection is achieved through the structure of the autoencoder, where the visual difference between the input and the reconstructed version can highlight the location of anomalies. Secondly, the E-encoder helps in learning how to encode the images for the best possible representation, thus improving the reconstruction process. Overall, this architecture aims to effectively detect anomalies and generate accurate reconstructions $\hat{x}$ of the input data x [7].

2.3.2 Discriminator network

The discriminator, in the standard adversarial training, is trained to discern between real and generated data. When it is not able to discern among them, it means that the generator produces realistic images. The generator is continuously updated to fool the discriminator.

2.3.3 Generator loss

The objective function is formulated by combining three loss functions, each of which optimizes a different part of the whole architecture.

- **Adversarial loss**. The feature matching loss. Eq 8 [8, 9].

$$L_{\text{adv}} = \mathbb{E}_{x \sim p_X} \left\| f(x) - \mathbb{E}_{x \sim p_X} f(G(x)) \right\|_2 \tag{8}$$

  Where f is a layer of the discriminator D, used to extract a feature representation of the input. Alternatively, binary cross entropy can be used.

- **Contextual loss** This loss generator learns contextual information about the input data [11].

$$L_{\text{con}} = \mathbb{E}_{x \sim p_X} \left\| x - (G(x)) \right\|_1 \tag{9}$$

- **Encoder Loss** This loss is used to let the generator network learn how to best encode a normal (non-anomalous) image.

$$L_{\text{enc}} = \mathbb{E}_{x \sim p_X} \left\| G_E(x) - E(G(x)) \right\|_2 \tag{10}$$

The final generator loss will be the result of the (weighted) sum of the previously defined losses.

$$L = w_{adv} L_{adv} + w_{con} L_{con} + w_{enc} L_{enc} \tag{11}$$

where w_{adv}, w_{con} and w_{enc} are weighting parameters used to adjust the importance of the three losses.
The authors of these ideas [7] proposed to compute the anomaly score different way respect to AnoGAN, only using L_{enc}:

$$A(x) = \| \ G_E(x) - E(G(x)) \|_2 \tag{12}$$

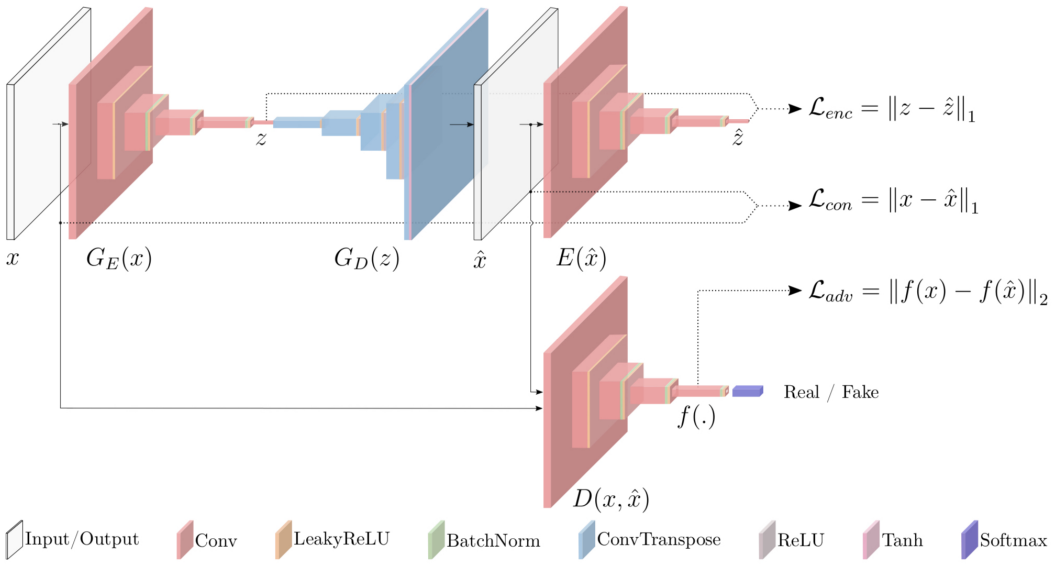


Figure 3: GANomaly architecture and loss function

PROS	**CONS**
An encoder is learned during the training process hence we don´t have the need for a research process as in AnoGAN	It allows detecting anomalies in both image space and latent space, but the results may not coincide: a higher anomaly score, which is computed only in latent space, may be associated with a sample generated with a low and thus very similar contextual loss value and vice versa.
Using a autoencoder like architecture (no use of noise prior) makes the entire learning process faster	Defines a new anomaly score
Anomaly score is easier to interpret	
The contextual loss can be used to localize the anomaly	

Table 2: Comparison of Pros and Cons of GANomaly

3 QGANS

We can applied the idea of GANs to quantum setting, named as QGANs and it can be formulated in several ways. One possibility could be both generator and discriminator as quantum systems where data takes the form of an assemble of quantum states, classic generator with a quantum discriminator or another possibility is a model with a quantum generator and a classical discriminator [12, 13].

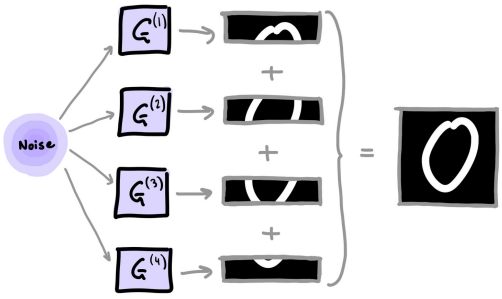


Figure 4: QGANs Generator lustration

3.1 Experimental QGANs for Learning and Loading Random Distributions proposed by IBM. [1]

The QGAN implementations proposed by IBM uses a quantum generator and a classical discriminator to capture the probability distributions of classical training samples. The aim of this approach is to train a data loading quantum channel for generic probability . In this setting, a parametrized quantum channel, the quantum generator, is trained to transform a given n-qubit input state $|\psi_{in}\rangle$ to an n-qubit output state.

$$G_\theta |\psi_{in}\rangle = |g_\theta\rangle = \sum_{j=0}^{2^n-1} \sqrt{p_\theta^j} |j\rangle \tag{13}$$

where p_θ^j describe the resulting occurrence probabilities of the basis states $|j\rangle$ The quantum generator is implemented by a variational form, a parametrized quantum circuit . They consider variational forms consisting of alternating layers of parametrized single-qubit rotations, the Pauli-Y-Rotations R_Y and blocks of two-qubit gates and controlled-Z-Gates, called entanglement blocks U_{ent}. The circuit consists of a first layer of R_Y gates and the k alternating repetitions of U_{ent} and further layers of R_Y gates. The rotation acting on the i^{th} qubit in the j^{th} layer is parametrized by $\theta^{i,j}$. Moreover, the parameter k is called the depth of the variational circuit. If such a variational circuit acts on n qubits it uses in total (k + 1)n parametrized single qubit gates and kn two-qubit gates. Increasing the depth k enables the circuit to represent more complex structures and increase the quantum generator´s ability to represent the number of parameters. Another possibility is adding ancilla qubits as this facilitates an isometric instead of a unitary mapping. The use of R_Y and CZ gates in contrast to other Pauli rotations and two qubit gates is for $\theta^{i,j} = 0$, so the variational form does not have any effect on the state amplitudes but only flips the phases, where the phase flips do not perturb the modelled probability distribution which solely depends on the state amplitudes.

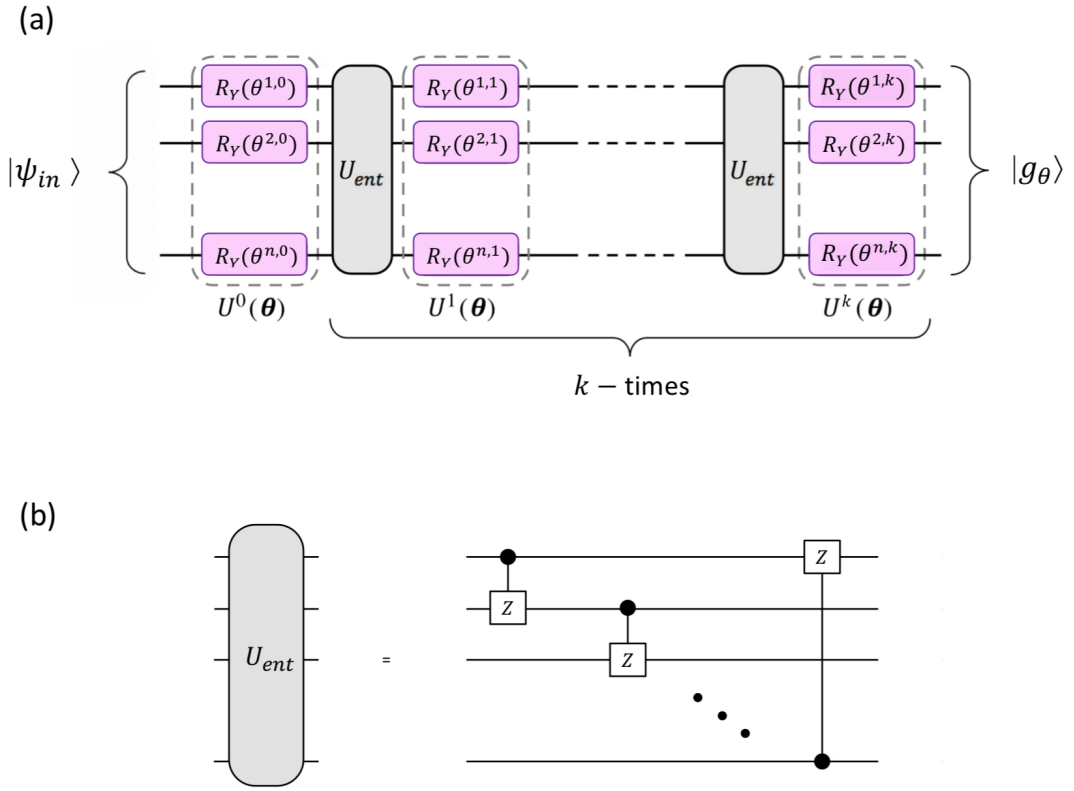


Figure 5: The variational form, depicted in (a), with depth k acts on n qubits. It is composed of $k+1$ layers of single-qubit Pauli-Y -rotations and k entangling blocks U_{ent}. As illustrated in (b), each entangling block applies CZ gates from qubit i to qubit $(i+1) \mod (n)$, $i \in \{0, ..., n-1\}$ to create entanglement between the different qubits.

3.1.1 The qGAN training

To train the qGAN, samples are obtained by measuring the output state $|g_\theta\rangle$ in the computational basis, where possible outcomes are $|j\rangle$ with $j \in \{0, ..., 2^n - 1\}$. Unlike classical methods, this sampling does not require stochastic input but relies on the inherent randomness of quantum measurements. The measurements yield classical information, with p_j representing the frequency of outcome $|j\rangle$.

This approach extends to d-dimensional distributions by using d qubit registers, each with n_i qubits, to form a multidimensional grid.

A well-chosen input state $|\psi_{\text{in}}\rangle$ can simplify the quantum generator, reduce training epochs, and avoid local optima. However, its preparation should not excessively increase gate complexity, meaning it must be loadable with $O(\text{poly}(n))$ gates. This is feasible for efficiently integrable probability distributions, such as log-concave distributions. In practice, statistical analysis of training data helps select a suitable $|\psi_{\text{in}}\rangle$ by matching expected values and variances.

3.1.2 The classical discriminator

The classical discriminator is a standard neural network consisting on several layers that apply non-linear activation functions, processes the data samples and labels them either as being real or generated. The number of nodes and layers, needs to be carefully chosen to ensure that the discriminator does not overpower the generator and vice versa Given m data samples g^l from the quantum generator and m randomly chosen training data samples x^l, where $l = 1, \ldots, m$, the loss

functions of the qGAN are

$$L_G(\phi, \theta) = -\frac{1}{m} \sum_{l=1}^{m} [\log D_\phi(g^l)] \tag{14}$$

for the generator, and

$$L_D(\phi, \theta) = -\frac{1}{m} \sum_{l=1}^{m} [\log D_\phi(x^l) + \log(1 - D_\phi(g^l))] \tag{15}$$

for the discriminator, respectively. As in the classical case, the loss functions are optimized alternatively with respect to the generators parameters θ and the discriminator´s parameters ϕ.

3.1.3 Simulation study by IBM

They present a broad simulation study on training qGANs with different settings for various target distributions. The quantum generator, implemented using Qiskit, operates on $n = 3$ qubits, allowing it to represent 8 values $\{0, ..., 7\}$. We applied the method to 20,000 samples of three target distributions: a log-normal distribution, a triangular distribution, and a bimodal Gaussian mixture. The generator's input state $|\psi_{\text{in}}\rangle$ was initialized using a uniform, truncated normal, or randomly chosen distribution.

The quantum generator depth k was varied across $\{1, 2, 3\}$, and the discriminator, a classical neural network implemented in PyTorch, was trained using AMSGRAD. The training process involved alternating updates of the generator and discriminator, with stability improvements via gradient penalties. Training convergence was evaluated using the Kolmogorov-Smirnov statistic and relative entropy.

Results from 10 training runs indicate that increasing k improves performance and that careful initialization enhances results, particularly for bimodal distributions. The approach is shown to be robust, as evidenced by small standard deviations and a high number of accepted results. Figure 6 visualizes the outcomes for each target distribution.

Figure 6: Benchmarking the qGAN training for log-normal, triangular and bimodal target distributions, uniform, normal and random initializations, and variational circuits with depth 1, 2 and 3. The tests were repeated 10 times using quantum simulation. The table shows the mean (μ) and the standard deviation (σ) of the Kolmogorov-Smirnov statistic (KS) as well as of the relative entropy (RE) between the generator output and the corresponding target distribution. Furthermore, the table shows the number of runs accepted according to the Kolmogorov Smirnov statistic ($n \leq b$) with confidence level 95%, i.e., with acceptance bound $b = 0.0859$.

data	init	k	μ_{KS}	σ_{KS}	$n_{\leq b}$	μ_{RE}	σ_{RE}
log-normal	uniform	1	0.0522	0.0214	9	0.0454	0.0856
		2	0.0699	0.0204	7	0.0739	0.0510
		3	0.0576	0.0206	9	0.0309	0.0206
	normal	1	0.1301	0.1016	5	0.1379	0.1449
		2	0.1380	0.0347	1	0.1283	0.0716
		3	0.0810	0.0491	7	0.0435	0.0560
	random	1	0.0821	0.0466	7	0.0916	0.0678
		2	0.0780	0.0337	6	0.0639	0.0463
		3	0.0541	0.0174	10	0.0436	0.0456
triangular	uniform	1	0.0880	0.0632	6	0.0624	0.0535
		2	0.0336	0.0174	10	0.0091	0.0042
		3	0.0695	0.1028	9	0.0760	0.1929
	normal	1	0.0288	0.0106	10	0.0038	0.0048
		2	0.0484	0.0424	9	0.0210	0.0315
		3	0.0251	0.0067	10	0.0033	0.0038
	random	1	0.0843	0.0635	7	0.1050	0.1387
		2	0.0538	0.0294	9	0.0387	0.0486
		3	0.0438	0.0163	10	0.0201	0.0194
bimodal	uniform	1	0.1288	0.0259	0	0.3254	0.0146
		2	0.0358	0.0206	10	0.0192	0.0252
		3	0.0278	0.0172	10	0.0127	0.0040
	normal	1	0.0509	0.0162	9	0.3417	0.0031
		2	0.0406	0.0135	10	0.0114	0.0094
		3	0.0374	0.0067	10	0.0018	0.0041
	random	1	0.2432	0.0537	0	0.5813	0.2541
		2	0.0279	0.0078	10	0.0088	0.0060
		3	0.0318	0.0133	10	0.0070	0.0069

3.1.4 IBM conclusions

An efficient, approximate probability distribution learning and loading scheme based on qGANs is demonstrated, requiring $O(\text{poly}(n))$ gates. In contrast, state-of-the-art methods for exact loading of generic distributions into an n-qubit state require $O(2^n)$ gates, significantly increasing the complexity of quantum algorithms. The quantum channel, implemented through a gate-based quantum algorithm, can be seamlessly integrated into other quantum algorithms. This is explicitly demonstrated by learning and loading a model for European call option pricing, which is evaluated using a QAE-based algorithm that provides a quadratic speed-up compared to classical Monte Carlo simulations.

The flexibility of the model allows it to adapt to the complexity of the data, with a trade-off between resolution and training complexity managed by adjusting the number of qubits n and circuit depth k. Additionally, qGANs support online or incremental learning, enabling model updates as new training data becomes available, potentially reducing training time in real-world applications.

Several open questions remain for future research, including the optimization of quantum generator and discriminator structures, as well as training strategies. Similar to classical machine learning, determining the most suitable model structure and training method for a given problem is not straightforward. Furthermore, although barren plateaus were not observed in the experiments, their potential occurrence and possible mitigation strategies should be investigated. Classical machine learning techniques, such as adding noise, incorporating momentum-based optimization, or computing higher-order gradients, may offer solutions. Additionally, methods developed for VQE algorithms, such as adaptive initialization, could help address this issue.

Another promising research direction involves exploring the representation capabilities of qGANs with different data types. Encoding data into qubit basis states inherently results in a discrete and uniformly distributed set of values. However, investigating the compatibility of qGANs with continuous or non-uniformly distributed data could provide valuable insights.

3.2 qGANs for anomaly detection for HEP. Study by IBM, ETH Zurich and CERN [2]

The Standard Model (SM) of particle physics provides a theoretical framework for describing fundamental forces. Despite its broad applicability, the SM is unable to account for all possible physical events. The identification of anomalous events—those not explained by the SM—and the potential discovery of exotic physical phenomena pose significant challenges. These challenges are expected to increase with next-generation colliders, which will generate a higher number of events with greater complexity. This motivates the development of unsupervised anomaly detection methods that do not rely on prior knowledge of the underlying models.

In this study, a quantum generative adversarial network (qGAN) is utilized to identify anomalous events. The method is designed to learn the background distribution from SM data and subsequently assess whether a given event aligns with the learned background. The proposed quantum anomaly detection strategy is evaluated through numerical simulations and IBM Quantum processors. The results indicate that quantum generative techniques, utilizing ten times fewer training samples, achieve accuracy comparable to classical approaches in detecting Graviton and Higgs particles. Additionally, an empirical analysis of the quantum model's capacity reveals enhanced expressivity compared to classical methods.

3.2.1 Quantum approach for anomaly detection

In this study they introduce their A-qGAN algorithm for the detection of anomalies, considering a qGAN architecture with a quantum generator and a quantum discriminator. First, each classical data point x_j is mapped to a quantum state $|x_j\rangle$. During the training, the qGAN learns a representation for the distribution underlying the generation process of the classical $\{x_j\}_{J=1}^{M}$. The anomaly score is the evaluated for data points $|x\rangle$ of the data set. The loss for the anomaly score is defined

as:

$$SQ(x;\alpha) = (1-\alpha)|\langle x|G\rangle|^2 + \alpha\frac{1+\langle Z\rangle_{D|x\rangle}\langle Z\rangle_{D|G\rangle}}{2} \tag{16}$$

with $\langle Z\rangle|u\rangle = \langle u|1\otimes 1\otimes\cdots\otimes Z|u\rangle$. The first term is a *residual score* which measures the fidelity between the embedded input point $|x_j\rangle$ and the trained generator state $|G\rangle$. The second term is a *discriminator score* based on the classification of the embedded input point $|x_j\rangle$ and the trained generator state $|G\rangle$. The parameter α weight the importance of both circuits for the anomaly score. An event is labelled as anomalous if

$$SQ(x;\alpha) = \delta \tag{17}$$

for reasonably chosen δ and α. The quantum anomaly scores does not require an additional optimization over the random prior z unlike its classical counterpart.

3.2.2 The studied dataset

This study utilizes an artificial dataset based on HEP experiments, combining Standard Model (SM) processes and anomalous Beyond the Standard Model (BSM) processes. The SM dataset contains 3,450,279 training events and 3,450,277 testing events, while datasets for the Higgs boson and Graviton contain 139,991 and 6,910 events, respectively. Each event is characterized by 23 features. For the quantum approach, 100 events from each dataset are used for training and testing. For the classical benchmark, 1,000 events per dataset are used.

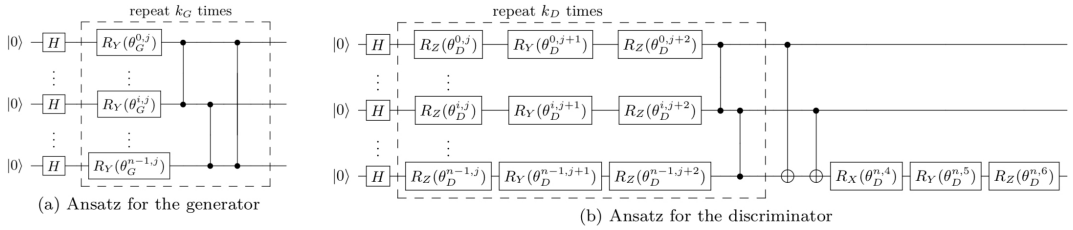
Classical preprocessing involves Principal Component Analysis (PCA) and normalization of data to the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$. The quantum data is encoded into an n-qubit quantum state $|x\rangle$ using angle encoding:

$$|x\rangle = \bigotimes_{i=1}^{n} R_y(x_i)|0\rangle_i. \tag{18}$$

where x_i corresponds to the i-th feature of the data point x, and $|0\rangle_i$ denotes the ground state of the i-th qubit.

3.2.3 Quantum implementation

They implement the quantum anomaly detection algorithm in Qiskit. The quantum generator circuit $G(\theta_G)$ consist of an initialization layer and k_G alternating layers of parametrized Pauli-Y single-qubit rotations and blocks of controlled-Z-gates. The initialization layer consist of Hadamard gates H on all qubits. In total, the circuit has $(k_G+1)n$ parametrized Pauli-Y rotations and $k_G n$ controlled-Z gates.



(a) Ansatz for the generator

(b) Ansatz for the discriminator

Figure 7: Ansatz for the quantum architecture. The quantum generator (a) consists of R_y rotation gates and CZ gates with variational parameters $\theta_G^{i,j}$ which corresponds to the parameter of the R_Y rotation acting on the i qubit on the j layer. The quantum discriminator (b) consists of R_X , R_Y and R_Z rotation gates, CZ and $CNOT$ gates with variational parameters $\theta_D^{i,j}$

The quantum discriminator is also a parametrized quantum circuit. We use an ansatz containing approximately as many parameters as the generator. The variational form consists in different parts.

1. They applied a layer of Hadamard gates on all qubits followed by k_D alternating layers of parametrized single-qubit rotations, Pauli-Y (R_Y) and Pauli-Z (R_Z) rotations on all qubits followed by controlled-Z gates (CZ).

2. They applied a series of controlled-X ($CNOT$) on the last qubit.

3. R_X, R_Y and R_Z single qubit rotations are applied on the last qubit.

In total, the circuit has $3(k_D n + 1)$ parametrized single-qubit rotations. At each optimization step, the training data is shuffled and split into batches of size 10. The discriminator is trained with samples from the batch while the generator is trained to output a single quantum state. They improve the stability of the algorithm with more optimization iterations for the discriminator than for the generator.

3.2.4 Classification scores

They use ROC metrics, accuracy, F1 score and precision as follow:

$$\text{Accuracy} = \frac{\text{True Anomalies}}{\text{Total number of events}} \tag{19}$$

$$\text{Precision} = \frac{\text{True Anomalies}}{\text{True Anomalies} + \text{False Anomalies}} \tag{20}$$

$$F1 = \frac{2}{\text{Recall}^{-1} + \text{Precision}^{-1}} \tag{21}$$

with

$$\text{Recall} = \frac{\text{True Anomalies}}{\text{True Anomalies} + \text{False Non-Anomalies}}. \tag{22}$$

3.2.5 Anomaly detection algorithm used in the approach

They compute the anomaly score for each sample and the mean of the anomaly score define thresholds which in turn help to identify whether a test sample is an anomaly or not. Grid Search was used over the α parameter, to find the one which results in the best AUC (Area Under Curve) on the test dataset. The algorithm for anomaly detection was the following one.

Algorithm 1 Anomaly detection algorithm

Data: Training data set $\mathbb{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$ (or $\{|\mathbf{x}_1\rangle, \ldots, |\mathbf{x}_M\rangle\}$), testing data set $\mathcal{T}$, generator optimizer Opt_G, discriminator optimizer Opt_D.

for number of training steps **do**

Train generative model $(\mathcal{G}, \mathcal{D})$ with Opt_G and Opt_D.

end for

for $\alpha \in [0, 1]$ **do**

for $\mathbf{x} \in \mathbb{D} \cup \mathcal{T}$ **do**

if classical **then**

$S_C(\mathbf{x}) \leftarrow \min_z[(1-\alpha)\|\mathbf{x} - \mathcal{G}(z)\| + \alpha\|\mathcal{D}(\mathbf{x}) - \mathcal{D}(\mathcal{G}(z))\|]$;

end if

if quantum **then**

$S_Q(\mathbf{x}) \leftarrow (1-\alpha)|\langle \mathbf{x}|\mathcal{G}\rangle|^2 + \alpha\frac{1+\langle Z\rangle_{\mathcal{D}|\mathbf{x}}\langle Z\rangle_{\mathcal{D}|\mathcal{G}}}{2}$;

end if

end for

Compute ROC Curve and AUC(α) score;

Compute $F_1(\alpha)$, Accuracy(α), and Precision(α) scores;

end for

Compute $\text{AUC}(\alpha_{\max}) = \max_\alpha \text{AUC}(\alpha)$;

return $F_1(\alpha_{\max})$, Accuracy($\alpha_{\max}$), and Precision($\alpha_{\max}$).

3.2.6 Results and discussion

Data results for the training of the (q)GANs on up to 8 features have been presented and then also demonstrate its feasibility on actual quantum using 3 features. Experiments at larger scale are currently infeasible due to the noise present in today´s quantum devices.

The results shown that quantum simulations optimize qGANs on a noiseless simulator for 500 epochs using the AMSGRAD optimizer with a learning rate of 10^{-3}. After training, anomaly scores are computed for SM, Higgs, and Graviton data sets, followed by classification. The procedure is repeated on a noisy simulator mimicking IBM Quantum's **ibmq belem**, using a higher learning rate (10^{-2}) and fewer training epochs.

Quantum models were benchmarked against classical neural networks with comparable trainable parameters. Anomaly detection is performed using 3 to 8 PCA-compressed features. Both quantum and classical methods yield similar classification metrics. The AUC score improved with more features until reaching a plateau at six features, where both models perform similarly. The classical model slightly outperforms the quantum one for fewer than seven features, but quantum simulations match classical performance with seven or more features.

Quantum models were trained with ten times fewer data samples, still achieve comparable accuracy, AUC, precision, and F1 scores. Noiseless and noisy quantum simulations yield similar results, indicating noise robustness. The observed standard deviations arise from GAN training volatility and dataset reshuffling.

Anomaly	Features	Method	Best α value	F1 score	Accuracy	Precision	ROC-AUC
Graviton	3	**GAN**	**0.25**	$\mathbf{0.83 \pm 0.127}$	$\mathbf{0.80 \pm 0.162}$	$\mathbf{0.91 \pm 0.183}$	$\mathbf{0.82 \pm 0.205}$
		qGAN	0	0.77 ± 0.084	0.71 ± 0.126	0.90 ± 0.163	0.70 ± 0.169
	7	**GAN**	**0.75**	$\mathbf{0.96 \pm 0.059}$	$\mathbf{0.95 \pm 0.072}$	$\mathbf{1.0 \pm 0.001}$	$\mathbf{0.96 \pm 0.086}$
		qGAN	0.75	0.92 ± 0.035	0.92 ± 0.042	1.0 ± 0.002	0.96 ± 0.038
Higgs	3	**GAN**	**0.5**	$\mathbf{0.83 \pm 0.088}$	$\mathbf{0.83 \pm 0.093}$	$\mathbf{0.98 \pm 0.036}$	$\mathbf{0.89 \pm 0.087}$
		qGAN	0.25	0.75 ± 0.081	0.70 ± 0.125	0.87 ± 0.162	0.71 ± 0.164
	7	GAN	1	0.82 ± 0.089	0.81 ± 0.098	1.0 ± 0.003	0.87 ± 0.091
		qGAN	**0.5**	$\mathbf{0.89 \pm 0.063}$	$\mathbf{0.88 \pm 0.080}$	$\mathbf{1.0 \pm 0.001}$	$\mathbf{0.92 \pm 0.084}$

Figure 8: Classification scores for the detection of BSM anomalies using three and seven principal components, given for the best value in the anomaly score. The bold characters correspond to the best results between the classical and quantum methods. In all cases, the quantum calculations are done with 100 data samples while the classical one contains 1000 data samples containing these 100 data samples. For 3 features, the classical model yields better classification scores than for the quantum one for both the Graviton and Higgs detections. For 7 features, the quantum and classical models yield similar classification scores for detection of both the Graviton and Higgs events.

Experimental results indicate that the quantum model achieves comparable accuracy to classical approaches while requiring ten times fewer training data points. Although data scarcity is not a major issue in HEP, this advantage could be valuable in other fields, such as medicine. The quantum model also demonstrates greater expressive power, suggesting its potential for detecting Beyond Standard Model (BSM) anomalies in more complex scenarios.

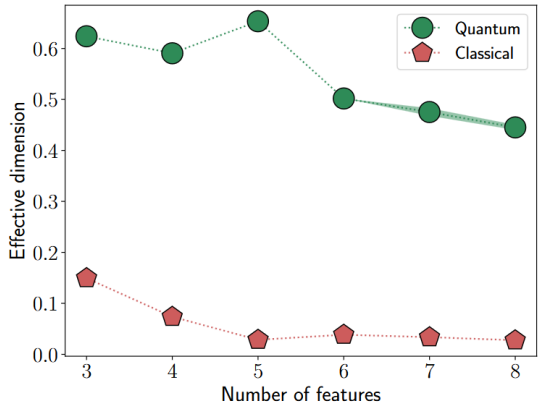


Figure 9: Effective dimension of the classical (red pentagons) and quantum (green circles) generator for increasing number of features. For each number of features, the classical and quantum generator have the same number of trainable parameters. Therefore, the higher effective dimension of the quantum generator indicates a potential advantage of the quantum approach over the classical one.

4 Qudit Quantum Machine Learning for classification task [3]

4.1 Learning with qudits

Most of the current research in this field are based on quantum variational algorithm (VQA) for addressing supervised learning problems. The data consists of input vector containing the features values ant their corresponding labels $\{\vec{x_i}, y_i\}_{i=1}^{N}$ with $dim(\vec{x_i}) = D_x$ and $y_i = 1, \ldots, K$. We can assume in this case that the data can be classified in K-classes. Typically VQSs can be divided into three steps:

1. **Encoding** the data onto a wave function $\vec{x_i} \to |\psi(\vec{x_i})\rangle$
2. **Evolving** it to a final state $U(\vec{\varphi})\,|\psi(\vec{x_i})\rangle$

$$|\psi(\vec{x_i}, \vec{\varphi})\rangle = \hat{U}(\vec{x_i}; \vec{\varphi})\,|0\rangle$$

3. Performing a **measurement** $o_i \equiv \langle 0|\, U_\theta^\dagger(\vec{x_i}, \vec{\varphi}) O U_\theta(\vec{x_i}, \vec{\varphi})\,|0\rangle$

Optimizing specific parameters ($\vec{\varphi}$) we minimize a predetermined objective function that depends on the output o_i

4.2 Qudit unitaries

Some studies take in consideration a d-level system with a fixed (time-independent) Hamiltonian H_0, which has eigenvalues $|l\rangle$, where $l = 0, \ldots, d-1$. Furthermore, they assume that a series of two-level rotations can be applied. Importantly, these operations are *universal* for a single qudit, enabling the generation of any wave function within the d-dimensional Hilbert Space.

4.3 Metric learning

Metric learning is well-suited for classifiers, particularly in image classification and recognition. This type of learning aims to map the data point onto a feature space equipped with a metric such the distance between points belonging to the same class are minimized while ones belonging to different classes are maximally separated. The algorithm aims to learn how to classify the input vector into one of these classes. The literature discusses two main approaches: **implicit** and **explicit** algorithms

4.3.1 Implicit approach

The training data defines K ensembles, one for each class. Using the encoded input vectors , K ensembles can be defined as follow:

$$\rho_k := \frac{1}{N_k} \sum_k tr(\rho_k^2) + \frac{2}{K} \sum_{k<l} tr(\rho_k \rho_l) \tag{23}$$

N_k represent the number of training points belonging to class k. The cost functions to be minimized reads:

$$L_I = 1 - \frac{1}{N_k} \sum_k tr(\rho_k^2) + \frac{2}{K} \sum_{k<l} tr(\rho_k \rho_l) \tag{24}$$

This approach aims to maximize the purity of the ensemble of states belonging to the same class while moving them away from ensembles of other classes by minimizing the trace of the crossed ensembles.

4.3.2 Explicit approach

In this approach, the algorithm utilizes predefined references states (referred to as centres) one for each class, denoted as $\{|\psi_k^R\rangle\}_{k=1}^K$. Introducing the density matrix,

$$\sigma_k := 1 - \frac{1}{K} \sum_k tr(\rho_k \sigma_k) \tag{25}$$

Here, it aims to minimize the distance between the training data and the references states. The last term into the above equation is nothing but the fidelity, $tr(\rho_k \sigma_k) = F(\rho_k, \sigma_k) := \langle \psi_k^R|\, \rho_k\, |\psi_k^R\rangle$. The performance of the algorithm depends on the selection of the centres. Ideally they should be as separate as possible. In terms of quantum states, orthogonal states are maximally separated. Its natural to choose $\langle \psi_k^R|\, |\psi_{k'}^R\rangle = \delta_{k,k'}$. However, it is possible that $d < K$.

4.4 Encoding strategies

The mentioned paper [3] differ with other approaches in that they do not apply independent ansatz for the encoding. The model learns a metric, so the encoding itself already maximizes the distance between different classes. Only a single ansatz was considered, where the classical data and the parameters to be optimized are tied together. We need to discuss the chosen function to map the input vectors and variational parameters to rotation angles.

$$g : (\vec{x}, \vec{\varphi}) \to (\vec{\theta}, \vec{\phi}) \tag{26}$$

This, in turn, will give rise to the embedding

$$\varepsilon : \mathbb{R}^{D_x} \to \mathcal{H}^d \tag{27}$$

$$\vec{x} \to |\psi(\vec{x})\rangle \tag{28}$$

through the ansatz. They choose g similar to those used in classical neural networks. Specifically, our optimization parameters, $\vec{\varphi}$, are divided into a weight matrix, $\overline{w}$, and a bias vector, $\vec{b}$, such that $g : \vec{x} \to \vec{x'} = \overline{w}\vec{x} + \vec{b}$. This transformed vector, denoted as $\vec{x'}$, is then used to determine the rotation angles $\vec{\theta}$ and $\vec{\phi}$. Relevant comments here are that the dimension of $\vec{x'}$, $D_{x'}$, does not have to be equal to the dimension of $\vec{x}$, $D_{x'} \neq D_x$ Even though there are $2(d-1)$ parameters in a layer of our ansatz, they can handle a higher dimension by employing sublayers. They split the vector $\vec{x'}$ into tuples of dimension $2(d-1)$. Furthermore, the assignment of rotation angles $(\vec{\theta}, \vec{\phi})$ from the transformed data $\vec{x'}$ is arbitrary and will impact the model´s performance.

Taking a 3-level system (a qutrit) with $d = 3$ for classifying data into three classes ($K = 3$). The reference states are the lowest-energy states from the orthonormal basis of the qutrit: $|k\rangle$ for class k with $k = 0, 1, 2$. Supposing the data dimension is $D_x = 4$. W could start with the following transformation: $x'_i = \overline{w}_{ii}x_i + b_i$. In other words, the matrix $\overline{w}$ is chosen to be diagonal, and the angles are assigned as $(\theta_i, \phi_i) = (x'_{2i}, x'_{2i+1})$, so $\theta_i = \theta_i(x_{2i})$ and $\phi_i = \phi_i(x_{2i})$. Since it is a qutrit, we have two available transitions and, thus four rotation angles in our ansatz. Applying this transformation to the ground state $|0\rangle$ yields the following expression:

$$|\psi(\vec{x})\rangle = \hat{U}(\vec{x}, \overline{w}, \vec{b}) |0\rangle = \sum_k c_k |k\rangle \tag{29}$$

where the coefficients c_k are as follows:

$$\begin{aligned} c_0 &= \cos\frac{\theta_0}{2}, \\ c_1 &= -i \sin\frac{\theta_0}{2} \cos\frac{\theta_1}{2} e^{i\phi_0}, \\ c_2 &= -\sin\frac{\theta_0}{2} \sin\frac{\theta_1}{2} e^{i(\phi_0+\phi_1)} \end{aligned}$$

As we can observe, not all parameters play a role in the decision boundaries. This issue can be address in various ways:

1. Initialize the qutrit in the state $|+\rangle = \frac{1}{\sqrt{k}} \sum_k |k\rangle$, becoming coefficients c_k a more intricate combination of rotation angles.

2. Change the basis in which the reference states are tailored. Instead of using the orthonormal qubit case $\{|0\rangle, |1\rangle\}$ we could use $\{|+\rangle, |-\rangle\}$.

3. Apply a different transformation g such that $g : \mathbb{R}^{D_x} \to \mathbb{R}^{2(d-1)}$, considering $\overline{w}$ as a general $(D_{x'}, D_x)$ matrix with $D_{x'} = 2(d-1)$, ensuring that the output always fit into one sublayer. Again we need to initialize the ground state to $|+\rangle$ instead of $|0\rangle$ to face the issue

The following table summarize the previous ways.

Encodings		
g	Transformation	Initialization
g_1	$x'_i = \bar{\omega}_{ii} \cdot x_i + b_i$	$\|0\rangle$
g_2	$x'_i = \bar{\omega}_{ii} \cdot x_i + b_i$	$\|+\rangle$
g_3	$x'_i = \sum_j \bar{\omega}_{ij} \cdot x_j + b_i$	$\|+\rangle$

Figure 10: Different encodings used in this section

Have seen different ways of choosing the function that maps our classical data into the control parameters of our physical system, its also important to consider other ways of defining the control itself. The control function in quantum models can be redefined using maximally orthogonal states (MOS) instead of standard basis states, enabling efficient operations when the number of classes exceeds available levels. By leveraging MOS, quantum state generation may require less optimization effort. Additionally, for high-dimensional input data ($D_x \gg d$), classical dimensionality reduction techniques like Principal Component Analysis (PCA) and Convolutional Neural Networks (CNNs) can be explored to improve encoding efficiency.

4.4.1 Iris Dataset

The review paper [3] applied the previous information to Iris Dataset, a Fisher´s database of different species of the Iris plant which is on of the most used dataset for classification problems. It contains 3 different species (classes) and for each one, 50 data points. Each data point is composed by 4 features which are the length and width of both the sepal and petal of each plant. It is a perfect dataset for an initial benchmark due to the small size of the dataset, the low dimensionality of the data and the small number of classes. Here, we used a training set of $N_{training} = 30$ data points, 10 for each class, and the rest is left for the test set, $N_{test} = 120$. To gain insight, they leverage the Iris dataset and comparisons are conducted using the explicit method, where the centres are predetermined as described. The chosen performance metric was:

$$\text{Test Accuracy} = \frac{\text{Number of Correct Predictions}}{N_{test}}. \tag{30}$$

The following figure illustrate the investigation conducted using various ansatz designs.

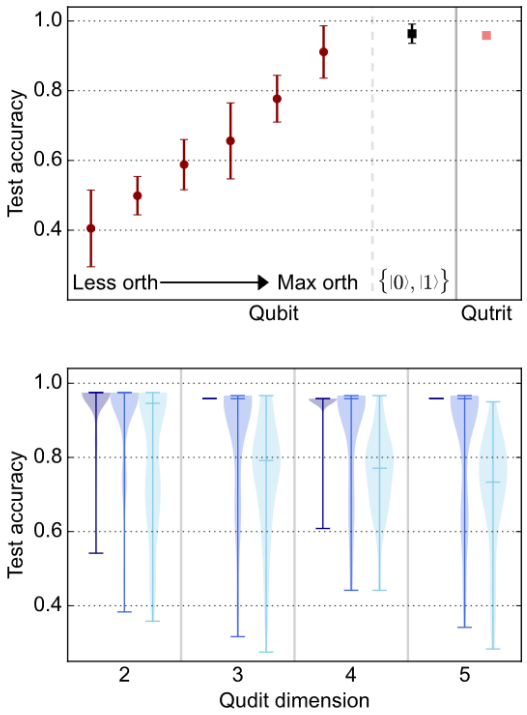


Figure 11: Comparison between ansatz in the explicit framework. Top: Comparison between different ansatz structures (design of the basic pulses). For the red rounded points, a non-orthogonal set of states is used as basis. The black square point represents the results when the original orthonormal basis is used and the last salmon square point marks the result for a qutrit (also with the orthonormal basis). Bottom: Comparison between different embedding functions of the data (how the original data is transformed into the parameters of the pulses). From darker to lighter colours: g_1, g_2 and g_3 from Figure 10

The qutrit ($d = 3$) serves as an ideal unit of information for this problem. It has two transitions, enabling the embedding of complete data dimensionality within our ansatz, without necessitating sublayers. Surprisingly, the performance of the qutrit falls short compared to the performance achieved using a qubit with sublayers. As we will delve into, this is due to the choice of the encoding function, denoted as g_1, which, as demonstrated in the previous section, fails to fully exploit the information provided by each data point.

As we can observe, for the darkest colour g_1 is negligible statistical variance when $d > 2$, with a single exception at $d = 4$. The encoding strategy g_1 is suboptimal as it discards half of the original data $\vec{x}$, treating it as relative phases that do not influence projections onto the orthonormal basis. While it performs well for this dataset, it may fail in scenarios where ignored features are crucial. Among the alternatives, g_2 and g_3, the latter, despite using more parameters, underperforms compared to g_2. Thus, encoding with $g_2 : x_i' \mapsto \bar{\omega}_{ii} \cdot x_i + b_i$, with the qudit initialized in $|+\rangle$, was chosen for all simulations.

4.4.2 Quantum metric learning

They now proceed to compare the performance of both explicit and implicit methods. The explicit method employs the g_2 encoding , whereas the implicit method follows g_1, initializing the state to $|0\rangle$. This choice is justified by the absence of fixed centres in the implicit approach, eliminating concerns about unused relative phases. For simplicity, the system is initialized in the ground state, though simulations with initialization in $|+\rangle$ produced similar results. The dimensionality of the qudit does not appear to be a critical factor, except for the Iris dataset, where $d > 2$ yields slightly better results than $d = 2$. This suggests that when the number of classes does not fully utilize the capacity of the MOS ($K \approx d$), the Hilbert space size has minimal impact. The implicit method demonstrates slightly superior performance over the explicit method due to its ability to determine optimal cluster centres for the data structure. Figure 12 compares the mapping results of both methods using the test set of the Iris dataset with a single qubit and a single layer. The reference states, indicated by arrows, and the colour plot representing their overlap reveal that while the explicit method generates perfectly homogeneous states, the implicit method produces slightly less homogeneous ones. This distinction, along with differences in cost function definitions, is reflected in the distribution of points

on the Bloch sphere: the explicit method aligns points along the circumference connecting the three reference states, whereas the implicit method shows greater dispersion on the surface.

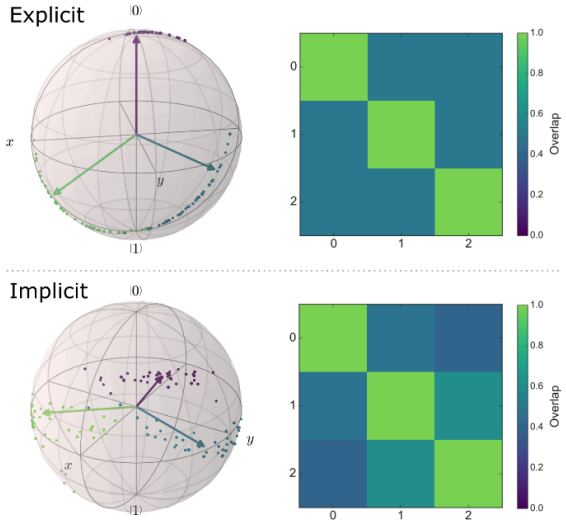
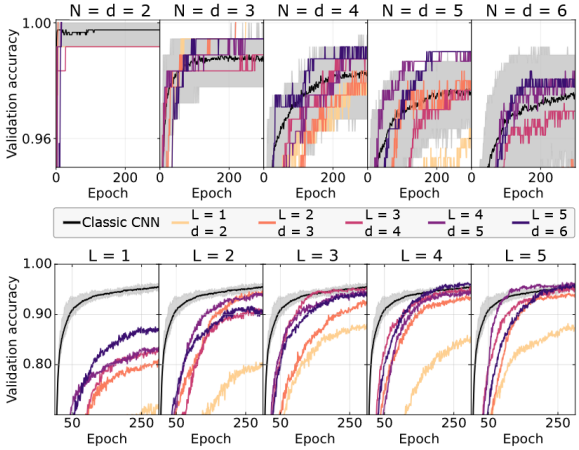


Figure 12: Different mappings produced by each method in the Iris data set. On the left side of the figure we represent each wave function produced by the circuit after training. The colour of each point represents the class at which it belongs. Same for the arrows (reference states). On the right side we plot the overlap between these reference states. Although not perfectly, the implicit method manages to find states that are practically homogeneous and maximally orthogonal in this case.

4.4.3 High-Dimensional Data

To investigate the problem of handling high-dimensional data, the MNIST dataset were used [3]. The standard data re-uploading technique becomes impractical due to the dataset's large dimensionality ($D_x \gg d$). Thus, dimensionality reduction is necessary. In this cases, Principal Component Analysis (PCA) is commonly used, transforming data into uncorrelated components while preserving relevant variance. However, flattening images into vectors removes spatial correlations crucial for digit classification. To solve this challenge, convolutional neural networks (CNNs) were employed, keeping spatial correlations by extracting features through convolutional layers. Also, a hybrid convolutional neural network (HCNN) were designed, integrating classical convolutional layers for preprocessing and a quantum layer for classification. The CNN reduces data dimensions to match the qudit's capacity, ensuring compatibility with the quantum layer.

Figure 13: Results obtained for the HCNN model. Top: Results for the partial uses of the data set. We use the same number of digits as levels available in the qudit. Bottom: Results for the full data set (10 digits). Line colours represent the number of layers used (top image) and the number of levels in the qudit (bottom image). In both cases we plot the validation accuracy as it is the one of which we keep track along all the epochs.



Results indicate that when the number of qudit levels matches the digit classes ($N = d$), the HCNN achieves high accuracy, even surpassing the classical model in some cases. However, for the full dataset, the quantum layer becomes a bottleneck, limiting performance. It is important to notice

that increasing the number of quantum layers improves results, demonstrating the importance of non-linearity in the quantum ansatz. Despite these improvements, the classical network maintains superior accuracy on the test dataset.

Test accuracy (%)					
d	L = 1	L = 2	L = 3	L = 4	L = 5
2	68.8	76.8	84.5	83.1	85.3
3	78.1	91.9	90.3	90.6	91.2
4	80.2	89.3	92.1	92.5	92.8
5	83.3	93.6	94.0	94.7	94.5
6	82.7	89.9	93.7	94.1	93.1
Classic	95.26 ± 0.14				

Figure 14: Test accuracy obtained from the best models. Each best model is defined as the one with the highest validation accuracy throughout the training.

4.4.4 Computational Complexity

The implicit method requires a higher training load due to the absence of fixed centres, necessitating evaluation of additional cross terms. Unlike the implicit method, the explicit method does not require full state tomography, simplifying cost function evaluation. The ansatz's universality allows direct unitary transformations to generate reference states, enabling efficient fidelity measurements. Training complexity depends on the number of training samples, circuit evaluations, and optimizer iterations. The test phase involves comparing each test point with reference states, making computational efficiency crucial for scalability.

4.4.5 Conclusion of the study

Evaluating quantum processors in machine learning is both timely and relevant. Simple quantum systems allow for easier implementation and deeper theoretical insights. This study investigates a d-level quantum system where transitions between levels are optimized for classification tasks. Key findings include:

- Different encoding strategies yield advantages for specific problems. The explicit method benefits from g_2 encoding, while the implicit method aligns with g_1. In the hybrid model, g_3 encoding proves most suitable.
- Both explicit and implicit metric learning methods demonstrate strong performance, sometimes rivalling classical algorithms. However, the explicit method is more practical for immediate experimental implementation.
- Simulations incorporating real-world noise sources validate their approach. A genetic algorithm aids in generating optimal measurement operator sets (MOS), ensuring adaptability for experimental implementations.
- The study extends to real-world datasets, including MNIST. A hybrid quantum-classical approach achieves reasonable performance, though the limited capacity of current quantum hardware remains a bottleneck compared to classical methods.

These insights highlight the promise of qudits in quantum machine learning while acknowledging current hardware limitations.

5 Qutrits approach for quantum machine learning based in [4]

Qutrits in machine learning appealing due to their potential to reduce the complexity and cost of quantum computing hardware, developing more efficient and scalable QML Algorithms. Qutrit

parametric circuits are used as a type of quantum circuits where qutrits are the fundamental building blocks, designed to perform machine learning classification tasks. The several advantages offered by qutrits including:

- **Reduced complexity**. Qutrits can encode more information per qubit.
- **Increased efficiently**. Qutrit parametric circuits can achieve high classification accuracy using fewer circuit elements than qubit systems.
- **Scalability**. Qutrits has the potential to reduce the complexity and cost of quantum computing hardware.

Quantum gate	Matrix Repr.	Quantum gate	Matrix Repr.	Quantum gate	Matrix Repr.
$Rx_{(3)}^{(01)}(\theta)$	$\begin{bmatrix} \cos\frac{\theta}{2} & i\sin\frac{\theta}{2} & 0 \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$Ry_{(3)}^{(01)}(\theta)$	$\begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} & 0 \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$Rz_{(3)}^{(01)}(\theta)$	$\begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$Rx_{(3)}^{(02)}(\theta)$	$\begin{bmatrix} \cos\frac{\theta}{2} & 0 & i\sin\frac{\theta}{2} \\ 0 & 1 & 0 \\ -i\sin\frac{\theta}{2} & 0 & \cos\frac{\theta}{2} \end{bmatrix}$	$Ry_{(3)}^{(02)}(\theta)$	$\begin{bmatrix} \cos\frac{\theta}{2} & 0 & -\sin\frac{\theta}{2} \\ 0 & 1 & 0 \\ \sin\frac{\theta}{2} & 0 & \cos\frac{\theta}{2} \end{bmatrix}$	$Rz_{(3)}^{(02)}(\theta)$	$\begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$
$Rx_{(3)}^{(12)}(\theta)$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\frac{\theta}{2} & i\sin\frac{\theta}{2} \\ 0 & -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$	$Ry_{(3)}^{(12)}(\theta)$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ 0 & \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$	$Rz_{(3)}^{(12)}(\theta)$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & e^{-i\frac{\theta}{2}} & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$
$CX_{(3)}$	$\begin{bmatrix} 1&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0&0 \end{bmatrix}$				

Figure 15: 3-level quantum gates used in this study and their matrix representation

5.1 Qudits Quantum data embedding

Quantum data embedding allows classical data to be represented in its quantum state in Hilbert space, mapping from the data space D to the Hilbert space H. This mapping takes a classical data point x and translate it into a set of gate parameters in a quantum circuit, creating a quantum state $|\psi_x\rangle$

5.1.1 Angle encoding

Angle encoding encoded in qudits using quantum rotation gates with the angles if the gates determined by the classical data.

$$|\psi_x\rangle = \otimes_l^n R_d(x_j)\,|0^i\rangle \tag{31}$$

where $R_{(d)}$ can be one or more $R_{X\,(d)}$, $R_{Y\,(d)}$ and $R_{Z\,(d)}$. The number of computational resources used for encoding equals the size of the vector.

5.1.2 Amplitude encoding

Amplitude encoding is a coding method where the amplitudes of the quantum system are used to represent data values. This method encodes a vector x of length N into the amplitudes of an $\lceil \log_d N \rceil$ computational source:

$$|\psi_x\rangle = \sum_{i=0}^{N-1} x_i\,|i\rangle \tag{32}$$

where $\{|i\rangle\}$ is the computational basis for the Hilbert space H_d. Since classical data constitute the amplitudes of a quantum state, the input must satisfy the $\sum_i |x_i|^2 = 1$ normalization condition.

5.2 Variational quantum neural networks

Variational Quantum Neural Networks (VQNNs) are hybrid quantum-classical machine learning models that integrate quantum circuits into neural networks. These networks encode input data into quantum states, process them through parameterized quantum circuits, and measure the results to obtain classical outputs. The parameters, acting as learnable weights, are optimized using gradient-based techniques such as automatic differentiation. VQNNs leverage quantum entanglement and interference to identify complex patterns, making them well-suited for classification, regression, and optimization tasks. Their fault-tolerant nature allows execution on noisy quantum computers, positioning them as promising tools for future quantum machine learning applications as quantum hardware advances.

5.2.1 Preprocessing (Applied to Iris, Wine and Breast Cancer datasets)

The datasets used in this study went through the following preprocessing stages.

1. **Dimensionality Reduction**. To make a precise comparison between qubit and qutrit systems, the current developments ensure that the same number of computational resources and parameters are utilized. They use PCA for dimensionality reduction only in with higher dimensional features to reduce their dimensionality.

2. **Feature Scaling**. Used to reduce these differences when different.

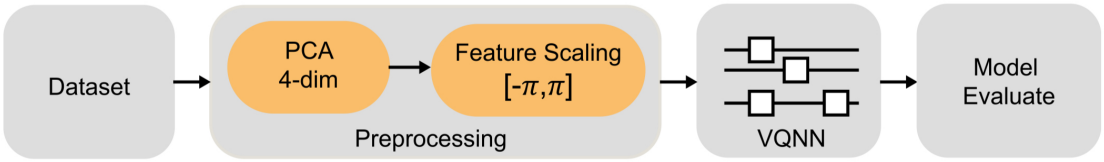


Figure 16: The essential elements of experimental methodology

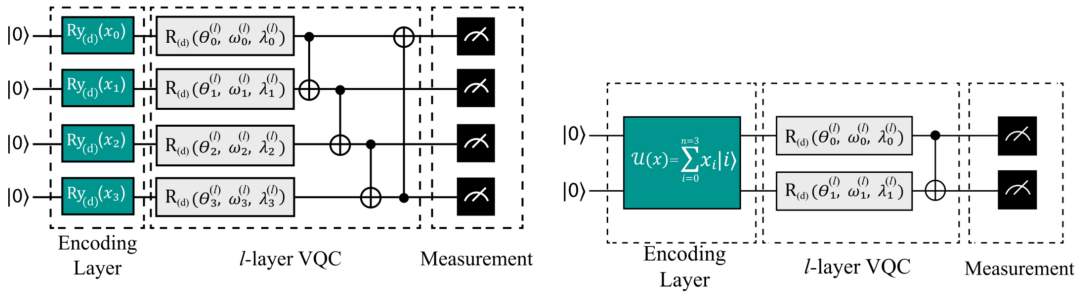


Figure 17: Circuit diagrams of VQNN with entanglement utilizing angle encoding (left) and amplitude encoding (right) methods

6 Comparison of classical, qubit, and qudit Approaches in QML and QGANs

Quantum Machine Learning (QML) and Quantum Generative Adversarial Networks (QGANs) can be implemented using classical, qubit-based, or qudit-based architectures, each offering distinct advantages and challenges. Classical models, while well-established, lack quantum advantages such

as entanglement and superposition. Qubit-based models leverage these quantum properties, potentially outperforming classical counterparts, though they are limited by hardware noise and circuit depth. Qudit-based approaches extend beyond qubits, allowing for more efficient information encoding and reduced circuit complexity, but are constrained by the current lack of hardware support and increased gate complexity. The following table summarizes key aspects of QGAN implementations.

Type	**PROS**	**CONS**	**Complexity**
Classical	Established methods, stable hardware	No quantum advantages	$\mathcal{O}(n)$
Qubit-based	Quantum speed-up, entanglement	Noise, limited hardware	$\mathcal{O}(\text{poly}(n))$
Qudit-based	Higher information density, reduced gates	Limited hardware, complex operations	$\mathcal{O}(\text{poly}(n))$

Table 3: Comparison of QGAN implementations using classical, qubit, and qudit approaches.

References

[1] Arrazola, J. M. *et al.* Quantum machine learning for quantum anomaly detection. *arXiv preprint arXiv:1904.00043* (2019). URL https://arxiv.org/pdf/1904.00043.

[2] Author, A. & Author, B. Quantum anomaly detection with qubits. *arXiv preprint arXiv:2304.14439* (2023). URL https://arxiv.org/pdf/2304.14439.

[3] Roca-Jerat, S., Román-Roche, J. & Zueco, D. Qudit machine learning. *Machine Learning: Science and Technology* **5**, 015057 (2024). URL http://dx.doi.org/10.1088/2632-2153/ad360d.

[4] Acar, E. & İhsan Yılmaz. Unlocking the high dimensional' potential: Comparative analysis of qubits and qutrits in variational quantum neural networks. *Neurocomputing* **623**, 129404 (2025). URL https://www.sciencedirect.com/science/article/pii/S0925231225000761.

[5] Wikipedia. Generative adversarial network. https://en.wikipedia.org/wiki/Generative$_a dversarial_n etwork$(2023). *Accessed* : 2023 − 10 − 10.

[6] Goodfellow, I. *et al.* Generative adversarial nets. In *Advances in Neural Information Processing Systems*, vol. 27 (2014). URL https://proceedings.neurips.cc/paper$_f iles/paper/$2014/$file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf$.

[7] Ruff, L. *et al.* A unifying review of deep and shallow anomaly detection. *arXiv preprint arXiv:1906.11632* (2019). URL https://arxiv.org/pdf/1906.11632.

[8] Dumoulin, V. *et al.* Adversarially learned inference (2017). URL https://arxiv.org/abs/1606.00704. 1606.00704.

[9] Zenati, H., Foo, C. S., Lecouat, B., Manek, G. & Chandrasekhar, V. R. Efficient gan-based anomaly detection. *CoRR* **abs/1802.06222** (2018). URL http://arxiv.org/abs/1802.06222. 1802.06222.

[10] Akcay, S., Abarghouei, A. A. & Breckon, T. P. Ganomaly: Semi-supervised anomaly detection via adversarial training. *CoRR* **abs/1805.06725** (2018). URL http://arxiv.org/abs/1805.06725. 1805.06725.

[11] Isola, P., Zhu, J., Zhou, T. & Efros, A. A. Image-to-image translation with conditional adversarial networks. *CoRR* **abs/1611.07004** (2016). URL http://arxiv.org/abs/1611.07004. 1611.07004.

[12] Schuld, M. & Killoran, N. Quantum machine learning models are kernel methods. *arXiv preprint arXiv:2010.06201* (2020). URL https://arxiv.org/abs/2010.06201.

[13] PennyLane. Quantum generative adversarial networks. https://pennylane.ai/qml/demos/tutorial$_q uantum_g ans$(2023). *Accessed* : 2023 − 10 − 10.

[14] Arrazola, J. M. *et al.* Quantum machine learning for quantum anomaly detection. *arXiv preprint arXiv:1904.00043* (2019). URL https://arxiv.org/pdf/1904.00043.

[15] PennyLane. Qutrits for quantum gans. https://discuss.pennylane.ai/t/qutrits-for-quantum-gans/4409 (2023). Accessed: 2023-10-10.

[16] Author, A. & Author, B. Unknown title. *arXiv preprint arXiv:2309.13036* (2023). URL https://arxiv.org/pdf/2309.13036.

[17] Author, A. & Author, B. Unknown title. *arXiv preprint arXiv:2311.12003* (2023). URL https://arxiv.org/pdf/2311.12003.

[18] Researcher, A. & Researcher, B. Quantum machine learning for high-energy physics. *Physical Review Research* **3**, 033056 (2021). URL https://journals.aps.org/prresearch/pdf/10.1103/PhysRevResearch.3.033056.